# Evaluating elliptic curve based KEMs in the light of pairings

David Galindo, Sebastià Martín and Jorge L. Villar

Dep. Matemàtica Aplicada IV. Universitat Politècnica de Catalunya
Campus Nord, c/Jordi Girona, 1-3, 08034 Barcelona
e-mail: {dgalindo,sebasm,jvillar}@mat.upc.es

**Abstract**

Several efforts have been made recently to put forward a set of cryptographic primitives for public key encryption, suitable to be standardized. In two of them (in the first place the NESSIE european evaluation project, already finished, and in the second place the standardisation bodies ISO/IEC), the methodology by Victor Shoup for hybrid encryption, known as *Key Encapsulation Method-Data Encapsulation Mechanism* (KEM-DEM), has been accepted. In this work we re-evaluate the elliptic curve based KEMs studied to become standards, which are called ACE-KEM, ECIES-KEM and PSEC-KEM. Their security is based on different assumptions related to the elliptic curve discrete logarithm (ECDL) problem on a random elliptic curve. First of all, we fix some inexact results claimed in the previous literature. As a consequence, the performance features of PSEC-KEM are dramatically affected. In second place, we analyse both their security properties and performance when elliptic curves with computable bilinear maps (*pairing curves* for short) are used. It turns out that these KEMs present a very tight security reduction to the same problem, namely the ECDH problem on such curves; moreover, one can even relate their security to the ECDL problem in certain curves with a small security loss. It is also argued that ECIES-KEM arises as the best option among these KEMs when pairing curves are used. This is remarkable, since NESSIE did not include ECIES-KEM over a random curve in its portfolio of recommended cryptographic primitives. It is concluded that for medium security level applications, which is likely the case for many embedded systems (e.g. smart cards), implementing these KEMs over pairing curves should be considered a very reasonable option.

**Keywords:** public-key cryptography, key encapsulation mechanisms, pairings, standardization, smart cards.

## 1 Introduction

A key encapsulation mechanism (KEM) is a probabilistic algorithm that produces a random symmetric key and an asymmetric encryption of that key. Using this random key in a suitable encryption scheme (referred to as a data encapsulation mechanism-DEM), a secure hybrid encryption of arbitrary long messages is obtained. The problem of designing secure DEMs in the standard model is efficiently solved

using well-known cryptographic techniques (cf. [CS]). Therefore, designing secure hybrid encryption schemes within the KEM-DEM methodology is reduced to designing secure KEMs.

As far as we know, there are three elliptic curve based KEMs that have been considered for standardisation so far (in particular in ISO/IEC 18033 [Sho04] and NESSIE[1] [NES03a]), namely, ACE-KEM, ECIES-KEM and PSEC-KEM. Their security relies on different problems related to the elliptic curve discrete logarithm (ECDL). PSEC-KEM and ECIES-KEM use the Random Oracle (RO) heuristic [BR93] in their security proofs, while ACE-KEM is proven secure in the standard model but based on a decisional assumption. They were first proposed as KEMs in [Sho01], the ISO standard draft for public key encryption by Victor Shoup, while in their original form they were submitted by IBM, Certicom and NTT corporations, respectively.

The hardness of these ECDL problems closely depends on the elliptic curve generation method used. Indeed, special families of elliptic curves with easy point-counting, such as supersingular curves or anomalous curves, turned out to be insecure (as shown, among others works, in [MOV93] and [Sma99]). How these negative results must be interpreted is a quite debatated question. The conservative approach is to avoid special families of curves: future developments may show inherent weaknesses in particular curves. The proposal is then to generate curves at random. A more efficient approach is to build curves with known order using complex multiplication techniques [AM93, LZ94], but then some randomness is lost in the way. Finally, the most appealing approach from a practical point of view is to use any curve which has not been proven insecure.

In [Jou00] a special family of curves, namely, elliptic curves with an efficiently computable non-trivial bilinear map (which will be hereafter referred to as *pairing curves*), were found a positive application in cryptography: the design of a one-round tripartite Diffie-Hellmann protocol. A breakthrough in this constructive direction was made in [BF01], presenting the most complete and practical identity-based encryption scheme to the date. Since then, pairing curves have found a lot of applications in cryptography (see [DBS04] for a comprehensive account).

But in [Jou00] was also pointed out that in such curves the Decisional Diffie-Hellman (ECDDH) problem becomes easy. Again, this negative result allows different interpretations. The conservative choice is to intepret this result as an inherent weakness of these curves, and therefore one should avoid using them in cryptography. However, in [JN03] pairing curves were presented for which the ECDL is believed to be hard, and the Computational Diffie-Hellman (ECDH) problem is equivalent to the ECDL. Since there are not known attacks against ECDL on these curves if appropiately generated, hardness assumptions related to pairing curves are being given more and more confidence by the cryptographic community.

**Our contribution.** We aim at getting an insight into the use of elliptic curve cryptography in the context of KEMs. We revisit the security proofs of the elliptic curve based KEMs when they are performed over pairing curves. As a result, *all these* KEMs can be proven secure in the RO heuristic with respect to the ECDH *assumption on a pairing curve*. Thus, the different security nature of these KEMs

[1] We point out that NESSIE is a not standardisation body, that is, it does not produce standards, although its results are helpful for standardisation bodies.

on a random curve is made uniform in the set of pairing curves, allowing a more natural comparison among them. The security reduction obtained turns out to be *very tight*, improving the concrete security claimed over a random curve. This enables the schemes to use smaller keys, and therefore make these KEMs suitable for implementation in constrained memory devices. We rigorously derive secure key sizes for each of these KEMs on pairing curves. It is worth pointing out that although the schemes are implemented over a pairing curve, and we use efficient pairing computations to obtain the concrete security results, *no pairing computations* are involved in a real implementation. The crucial point is that ECDDH problem is solvable in these groups.

We also find out that the key size for PSEC-KEM claimed in NESSIE is inexact. More precisely, there is no evidence supporting that the 160 bits key size used up to now for PSEC-KEM over a random curve results in a secure implementation. Following our security analysis, a larger key size is needed. As a consequence, the efficiency of PSEC-KEM is negatively affected.

On the other hand, using [Mau94] there are elliptic curves where ECDL can be reduced to ECDH. Then, it is possible to give an exact security result *relating the* IND-CCA *security of these* KEMs *to the* ECDL *problem*. The good news is that they are closely related, due to small security losses in the reduction. From a theoretical point of view, this gives more confidence on the security of these KEMs over pairing curves. In particular, we show that for the current security level (that is, $2^{80}$), breaking ECIES-KEM *is equivalent* to solving the ECDH or ECDL problem on a pairing curve with a prime order subgroup with a $2^{80}$ security of the ECDH or a $2^{102}$ security of the ECDL respectively. We point out that such a concrete estimation with respect to the ECDL is rarely found in the literature (to the best of our knowledge, only in [GJ03, MSV04] appear similar results).

Since ECIES-KEM has the best perfomance, it is concluded that ECIES-KEM *is preferable* among the others if pairing curves are used. This is interesting, since when using a randomly generated curve a different result is obtained. In fact, ECIES-KEM has not been selected in the evaluation carried out by NESSIE, while ACE-KEM and PSEC-KEM have been positively evaluated. We argue that in environments where a high security level is not important but efficiency is critical (which is likely the case for smart cards applications), *implementing these KEMs over pairing curves should be considered a very reasonable option*.

## 2 Security properties of existing elliptic curve based KEMs

We first summarize some notation. If $p$ is a positive integer, then $|p|$ denotes the length of its binary representation. If $A$ is a non-empty set, then $x, y \leftarrow A$ denotes that $x, y$ have been uniformly and independently chosen from $A$. On the other hand, if $\mathcal{A}$ is a probabilistic polynomial time (PPT) algorithm, then $x \leftarrow \mathcal{A}$ denotes that $x$ is the output of $\mathcal{A}$. $Hash$ and $KDF$ denote a hash function and a key derivation function, respectively (cf. [CS]).

**IND-CCA security of a KEM.** A KEM consists of three algorithms:

– A *key generation* algorithm $\mathcal{K}$, a probabilistic algorithm which takes as input a security parameter $1^\ell$ and outputs a public/secret-key pair $(\mathsf{pk}, \mathsf{sk})$.

- A *encapsulation* algorithm $\mathcal{E}$, a probabilistic algorithm taking as inputs a security parameter $1^\ell$ and a public key $\mathsf{pk}$ and returning an encapsulated key-pair $(K, C)$, with $K \in \{0,1\}^{p(\ell)}$, $C \in \{0,1\}^{q(\ell)}$, for some polynomials $p, q \in \mathbb{Z}[\ell]$.
- A *decapsulation* algorithm $\mathcal{D}$, a deterministic algorithm that, on inputs a security parameter $1^\ell$, an encapsulation $C$ and a secret key $\mathsf{sk}$; outputs a key $K$ or a special symbol $\mathsf{reject}$ meaning there was a failure in the execution of the algorithm.

It is required to be sound, that is, for almost all $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^\ell)$, and almost all $(K, C) \leftarrow \mathcal{E}(1^\ell, \mathsf{pk})$ we have that $K = \mathcal{D}(1^\ell, C, \mathsf{sk})$.

Here follows the description of the attack game used to define the IND-CCA security of a KEM:

- The adversary queries a *key generation oracle,* which computes $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^\ell)$ and returns $\mathsf{pk}$.
- The adversary makes a sequence of calls to a *decryption oracle*, submitting encapsulations $C$ of its choice, for which the decryption oracle responds with $\mathcal{D}(1^\ell, C, \mathsf{sk})$.
- The adversary queries an *encryption oracle*, which computes:

$$(K_0, C^*) \leftarrow \mathcal{E}(1^\ell, \mathsf{pk}); \quad K_1 \leftarrow \{0,1\}^{p(\ell)}; \quad b \leftarrow \{0,1\}$$

  and returns the pair $(K_b, C^*)$.
- The adversary issues new calls to the decryption oracle, subject only to the restriction that a submitted ciphertext $C \neq C^*$.
- The adversary outputs $b' \in \{0,1\}$.

For a PPT adversary $\mathcal{A}$ we define

$$\mathsf{Adv}_{\mathrm{KEM},\mathcal{A}}(\ell) := \left| \mathsf{Pr}\left[\mathcal{A}(1^\ell) = 1 \,|\, b = 0\right] - \mathsf{Pr}\left[\mathcal{A}(1^\ell) = 1 \,|\, b = 1\right] \right|.$$

We say that a KEM is IND-CCA secure if for all PPT adversaries $\mathcal{A}$ the function $\mathsf{Adv}_{\mathrm{KEM},\mathcal{A}}(\ell)$ grows negliglibly in $\ell$. A quantity $\epsilon(\ell)$ is negligible if for any polynomial $p \in \mathbb{R}[\ell]$, there exist $M_p \in \mathbb{R}^+$ such that $\epsilon(\ell) < \frac{M_p}{p(\ell)}$, for all $\ell \in \mathbb{Z}^+$.

**Elliptic curve discrete logarithm problems.** Let $E_{a,b}(\mathbb{F}_q)$ denote the group of points of the elliptic curve

$$E_{a,b} \; : \; y^2 = x^3 + ax + b$$

over the prime finite field $\mathbb{F}_q$, $q > 3$. For finite fields with characteristic 2 or 3, the equation defining an elliptic curve takes different forms [Men93]. Let $G_p = \langle P \rangle$ be a cyclic group of prime order $p$, where $P \in E_{a,b}(\mathbb{F}_q)$. Then:

- The *discrete logarithm* (ECDL) is the problem of finding $u$ when given $(P, uP)$.
- The *computational Diffie-Hellman* problem (ECDH) is the problem of finding $uvP$ when given $(P, uP, vP)$.
- The *decisional Diffie-Hellman* problem (ECDDH) is the problem of distinguishing $(P, uP, vP, uvP)$ from $(P, uP, vP, wP)$.
- The *gap Diffie-Hellman* problem (gap-ECDH) is the problem of finding $uvP$ when given $(P, uP, vP)$ and an oracle $\mathcal{O}$ that correctly solves the decisional Diffie-Hellman problem.

It is assumed that $u, v, w \leftarrow \mathbb{F}_q$. Notice that all three KEMs are intended to be performed on random elliptic curves, so all these problems are assumed to be intractable. All of them are well established, except for the gap-ECDH problem, which was formally introduced in [OP01]. It is an open problem to establish all the relations between them. In fact, we rigorously know little more than the obvious reductions, which are ECDDH infeasible $\Rightarrow$ ECDH infeasible $\Rightarrow$ ECDL infeasible; and gap-ECDH infeasible $\Rightarrow$ ECDH infeasible. Thus, the better way known to attack these problems in a general elliptic curve is to solve ECDL. The fastest method for solving ECDL on a random elliptic curve is the Pollar $\rho$ method [Pol78], which runs in exponential time $\sqrt{\pi q/2}$ for a group with $q$ elements. It is unknown whether there exist groups for which the ECDH problem is substantially easier than the ECDL problem, while the ECDDH problem appears to be easier than the ECDH problem in general. We refer the reader interested in the state of the art to [MW00].

**Concrete security.** The efficiency of a reduction is the relationship between an *attacker* who breaks the cryptosystem with probability at least $\epsilon$ in time $t$, doing less than $q_D$ calls to a decryption oracle, and less than $q_K$ calls to an oracle for a hash or a $KDF$ function; and the implied $(t', \epsilon')$ *solver* against the corresponding trusted cryptographic assumption. Such an attacker is referred as a $(t, \epsilon, q_D, q_{\mathcal{O}_i})$ attacker for short. Following the usual terminology, the security reduction is *tight* if $\frac{t'}{\epsilon'} \approx \frac{t}{\epsilon}$, and *not tight* if $\frac{t'}{\epsilon'} > q_D \frac{t}{\epsilon}$. It is also stated that a scheme is *very tight* if $\epsilon \approx \epsilon'$ and $t'$ is equal to $t$ plus a linear quantity in the number of oracle calls. The tighter is the reduction, the smaller is the gap between the computational efforts needed to break the scheme and to solve the underlying problem. The optimal tightness is achieved with *very tight* reductions.

To be consistent with the time units commonly used in the literature, we use the sentence *a problem $\mathcal{P}$ has a $2^t$ security level* to say that, an attacker against $\mathcal{P}$, running in time less than $2^t$ 3-DES encryptions (cf. [LV01]), has a negligible success probability.

**Known results about elliptic curve based KEMs.** The first step of the key generation algorithm in the three schemes studied is to build a suitable curve $E$, together with a point $P$ that generates a secure cyclic subgroup $G_p$ of $E$, with prime order $p$. Moreover, $p$ is of size $\ell$, where $\ell$ is the security parameter. So we will assume that the key generation algorithm takes the group parameters $(E, P, p)$ as input.

A so-called *key derivation function $KDF$* has been used in these KEMs. This function can be considered as a hash function for our purposes (for further details see [Sho04]). In Table 1 we summarize the exact security results known for the KEMs we are interested in, along with the reference where these results come from. In these expressions, $q_K$ denotes the number of queries made to the KDF oracle, $L_G$ is the time needed to check a Diffie-Hellman triple in $G$, and $SR_q$ is the time needed to compute a square root modulo $q$. We point out that in the ECIES-KEM security reduction claimed in [Den02], the authors do not take into account the time to compute a square root in $\mathbb{F}_q$, which is needed in order to obtain the two points in $E(\mathbb{F}_q)$ that have a given $x$-coordinate.

As we can see, ACE-KEM offers several security reductions, depending on which problem its security is based. In the case of the NESSIE evaluation, the emphasis

Schematic description of the elliptic curve KEMs

| $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathcal{K}(E,P,p,\ell)$ | $(K,C) \leftarrow \mathcal{E}(\mathsf{pk})$ | $K \leftarrow \mathcal{D}(C,\mathsf{sk})$ |
|---|---|---|
| 1. $w,x,y,z \leftarrow \mathbb{Z}_p^*$ | 1. $r \leftarrow \mathbb{Z}_p^*$ | 1. Parse $C$ as $(C_1,C_2,C_3)$ |
| 2. $W := wP$, $X := xP$, | 2. $C_1 := rP$ | 2. $\alpha := Hash(C_1 \| C_2)$ |
| $Y := yP$, $Z := zP$ | 3. $C_2 := rW$ | 3. $t := x + y\alpha$ |
| 3. $\mathsf{pk} := (E,P,p,W,X,Y,Z,\ell)$ | 4. $Q := rZ$ | 4. If $C_2 \neq wC_1$, |
| 4. $\mathsf{sk} := (w,x,y,z,\mathsf{pk})$ | 5. $\alpha := Hash(C_1 \| C_2)$ | output reject and halt |
| 5. Output $(\mathsf{pk},\mathsf{sk})$ | 6. $C_3 := rX + \alpha rY$ | 5. If $C_3 \neq tC_1$, |
| | 7. $C := (C_1,C_2,C_3)$ | output reject and halt |
| | 8. $K := KDF(C_1 \| Q)$ | 6. $Q := zC_1$ |
| | 9. Output $(K,C)$ | 7. $K := KDF(C_1 \| Q)$ |
| | | 8. Output $K$ |

Description of ACE-KEM

| $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathcal{K}(E,P,p,\ell)$ | $(K,C) \leftarrow \mathcal{E}(\mathsf{pk})$ | $K \leftarrow \mathcal{D}(C,\mathsf{sk})$ |
|---|---|---|
| 1. $s \leftarrow \mathbb{Z}_p^*$ | 1. $r \leftarrow \mathbb{Z}_p^*$ | 1. $Q := sC$ |
| 2. $W := sP$ | 2. $C := rP$ | 2. If $Q = \mathcal{O}$ |
| 3. $\mathsf{pk} := (E,P,p,W,\ell)$ | 3. Set $x$ the | output reject and halt |
| 4. $\mathsf{sk} := (s,\mathsf{pk})$ | x-coordinate of $rW$ | 3. Set $x$ x-coord. of $rW$ |
| 5. Output $(\mathsf{pk},\mathsf{sk})$ | 4. $K = KDF(C \| x)$ | 4. $K = KDF(C \| x)$ |
| | 5. Output $(K,C)$ | Output $K$ |

Description of ECIES-KEM

| $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathcal{K}(E,P,p,\ell)$ | $(K,C) \leftarrow \mathcal{E}(\mathsf{pk})$ | $K \leftarrow \mathcal{D}(C,\mathsf{sk})$ |
|---|---|---|
| 1. $s \leftarrow \mathbb{Z}_p^*$ | 1. $r \leftarrow \{0,1\}^\ell$ | 1. Parse $C$ as $(C_1,C_2)$ |
| 2. $W := sP$ | 2. $H := KDF(0_{32} \| r)$ | 2. $Q := sC_1$ |
| 3. $\mathsf{pk} := (E,P,p,W,\ell)$ | 3. Parse $H$ as $t \| K$ | 3. $r := C_2 \oplus KDF(1 \| C_1 \| Q)$ |
| 4. $\mathsf{sk} := (s,\mathsf{pk})$ | 4. $\alpha := t \bmod p$ | 4. $H := KDF(0 \| r)$ |
| 5. Output $(\mathsf{pk},\mathsf{sk})$ | 5. $Q := \alpha W$ | 5. Parse $H$ as $t \| K$ |
| | 6. $C_1 := \alpha P$ | 6. $\alpha := t \bmod p$ |
| | 7. $C_2 := r \oplus KDF(1_{32} \| C_1 \| Q)$ | 7. If $C_1 \neq \alpha P$, |
| | 8. $C := (C_1,C_2)$ | output reject and halt |
| | 9. Output $(K,C)$ | 8. Output $K$ |

Description of PSEC-KEM

is put on the ECDDH problem, since in this case the security is achieved in the standard model. On the other hand, ECIES-KEM presents a very tight reduction to the gap-ECDH problem, while PSEC-KEM has a not tight reduction to the ECDH problem. Both schemes are analysed within the RO heuristic. In Table 2 we have the key length in bytes for a $2^{80}$ IND-CCA security bound in each scheme. To compute them, the usual way is to set that in a random curve ECDDH, gap-ECDH or ECDH problems have complexity similar to the ECDL problem. Although this is widely believed, these computational equivalences are far from being proved. $KDF$'s output has been set to be 16 bytes. In computing the expected number of elliptic curve additions involved in encapsulation/decapsulation, the binary exponentiation algorithm has been used, that is, $3/2 \log p$ additions are expected to compute a random multiple $rP$ in $G_p$. Finally, note that from the values in Table 2, both ACE-KEM and ECIES-KEM need a group $G_p$ with a $p \approx 2^{160}$ cardinality, while PSEC needs $p \approx 2^{270}$ (although the values appearing in the table are referred to the definition field $\mathbb{F}_q$, is usually desired that $|p| \approx |q|$). This important difference between PSEC-KEM

| Scheme | Assumption | Reduction | Random Oracle | Reference |
|---|---|---|---|---|
| ACE-KEM | ECDDH | very tight | No | [CS] |
| | gap-ECDH | $\epsilon' \approx \epsilon$<br>$t' \approx t + q_K(2L_G + SR_q)$ | Yes | [Sho01]<br>[Den02] |
| | ECDH | not tight | Yes | [Sho00] |
| ECIES-KEM | gap-ECDH | $\epsilon' \approx \epsilon$<br>$t' \approx t + q_K(2L_G + SR_q)$ | Yes | [Den02] |
| PSEC-KEM | ECDH | $\epsilon' \approx \frac{1}{q_D + q_K}\epsilon$<br>$t' \approx t$ | Yes | [Sho01] |

Table 1: IND-CCA KEMs concrete security over a random curve

and the others KEMs is due to the fact that the security proof of PSEC-KEM is not tight. We notice that NESSIE parameter length estimation for PSEC-KEM is not exact (it is stated that a 160-bit prime is enough), and we argue it in Section 4.

| KEM | Problem | Exponentiations in Enc/Dec | $(K, C)$ length 16-Byte Keys | Public/Secret key length | EC additions in Enc/Dec |
|---|---|---|---|---|---|
| ACE | ECDDH | 5/3 | 76 | 80/80 | 1200/720 |
| ECIES | gap-ECDH | **2/1** | **36** | **20/20** | 480/240 |
| PSEC | ECDH | 2/2 | 66 | 34/34 | 810/810 |

Table 2: Performance features over random curves (byte lengths using a point compression technique)

Comparing these results is somewhat contrived, since different assumptions are involved. On the one hand we can compare them using their performance and, on the other hand, by using the hard problems they are based on. In terms of performance, ECIES-KEM is clearly the best option. Not only does it present the smallest computation time, but also the smallest parameter length. For instance, ECIES-KEM is roughly 2.5 and 3 times faster in encapsulation and decapsulation respectively than ACE-KEM. When compared to PSEC-KEM, we must take into account that they use different field sizes. To make the comparison possible, it is used than an elliptic curve addition is equal to a constant number of multiplications in the corresponding field $\mathbb{F}_q$; and that a multiplication in a field of 270 bits length takes $\left(\frac{270}{160}\right)^2$ times a multiplication in a field of 160 bits length. Therefore, ECIES-KEM is roughly 5.5 and 10 times faster in encapsulation/decapsulation than PSEC-KEM. However, since the security of ECIES-KEM is based on the gap-ECDH, a quite new assumption, NESSIE did not include ECIES-KEM in its portfolio of cryptographic primitives, while accepted ACE-KEM and PSEC-KEM, since these schemes base their security on well-studied assumptions. We emphasize that in NESSIE evaluation the performance features we present here were not considered, and maybe this could have changed their final decision. In the next section we provide evidence that if these KEMs are implemented over pairing curves, ECIES-KEM should be considered the best candidate.

# 3 Security analysis over pairing curves

Let $E_{a,b}(\mathbb{F}_q)$ be the group of points of an elliptic curve over the prime finite field $\mathbb{F}_q$. Let $G_p = \langle P \rangle$ be a cyclic subgroup of $E_{a,b}(\mathbb{F}_q)$ with $p$ elements, where $p$ is a large prime. Let $\mathbb{G}$ be a cyclic group with $p$ elements. We say that $E$ is a *pairing curve over $\mathbb{F}_q$ with respect to $G_p$* if there exists a map $e : G_p \times G_p \to \mathbb{G}$ with the following properties:

- Bilinear: that is, $e(uP, vQ) = e(P, Q)^{uv}$, for all $P, Q \in G_p$ and all $u, v \in \mathbb{Z}$.
- Non-degenerate: The map does not send all pairs in $G_p \times G_p$ to the identity in $\mathbb{G}$.
- Computable: There is an efficient algorithm to compute $e(P, Q)$ for any $P, Q \in G_p$.

We call them pairing curves because the usual way to implement the map $e$ is using the Weil or Tate pairings [Men93]. In this case, the group $\mathbb{G}$ is the multiplicative group of a certain finite extension $\mathbb{F}_{q^k}$. The number $k$ is called the *embedding degree* and is the smallest positive integer such that $p|(q^k - 1)$. Given a generation method for pairing curves, ECDL problems on these curves can be described. We use the notation Problem$^{\mathsf{pairing}}$ to say that a given Problem is defined with respect to a certain probability distribution in the set of pairing curves, while the absence of superindex means that the problem is defined over an elliptic curve generated at random.

With the map $e$, the ECDDH$^{\mathsf{pairing}}$ problem is solvable in $G_p$. The non-degeneracy property of the Weil and Tate pairings implies that $e(P, P)$ is a $p$-th root of unity, and then $(P, uP, vP, wP)$ is a valid Diffie-Hellman quadruple if and only if $e(uP, vP) = e(P, wP)$. It turns out then that gap-ECDH$^{\mathsf{pairing}}$ and ECDH$^{\mathsf{pairing}}$ problems are polynomial time equivalent in $G_p$, since there exists a polynomial time algorithm replacing the ECDDH oracle solver. We use this fact positively to tightly relate the security of ACE-KEM, ECIES-KEM and PSEC-KEM to the ECDH problem in these curves.

Another consequence of the map $e$ is that solving the ECDL problem in $G_p$ can be transformed into solving the DL problem over the finite field $\mathbb{F}_{q^k}$, which can be computed using an index calculus algorithm running in subexponential time. This has been applied to attack the ECDL problem over supersingular curves in [MOV93, FR94]. We should take this into account when computing secure key sizes for each scheme.

**Revisiting concrete security with respect to ECDH$^{\mathsf{pairing}}$.** We already know that gap-ECDH$^{\mathsf{pairing}}$ and ECDH$^{\mathsf{pairing}}$ problems are equivalent. According to the results summarized in Table 1, this implies that ACE-KEM and PSEC-KEM are straightforward secure with respect to the ECDH$^{\mathsf{pairing}}$ problem. Indeed, they present a very tight reduction to the ECDH$^{\mathsf{pairing}}$, and the concrete security estimation is obtained by replacing $L_{G_p}$ with twice the time needed to compute the map $e$, which will be denoted by $T_e$.

In the case of the PSEC-KEM security proof in [Sho01], the solver of the ECDH problem makes use of a $(t, q_D, q_K, \epsilon)$ adversary against the IND-CCA security of PSEC-KEM to generate a list of $q_D + q_K$ elements containing the solution $uvP$ to the instance $(P, uP, vP)$ with probability roughly $\epsilon$. Since in a random curve the ECDDH problem is assumed to be intractable, we were forced to output an element of the list chosen uniformly at random, so the probability was decreased by a factor $q_D + q_K$. The reduction was then not tight. Since ECDDH$^{\mathsf{pairing}}$ is efficiently solvable,

we can find the correct value $uvP$ by testing the entries on the list, obtaining thus a solver of ECDH$^{\mathsf{pairing}}$ with probability roughly $\epsilon$ within time $t + 2(q_K + q_D)T_e$. Therefore, PSEC-KEM presents a very tight security reduction, allowing the use of shorter ciphertexts than on a random curve, as we shall see in the next section. In Table 3 these concrete security results are summarized. We do not find out if this reduction can be improved using the ECDDH solver in the simulation, since our reduction is already very tight.

| Scheme | Assumption | Reduction |
|---|---|---|
| ACE-KEM | ECDH$^{\mathsf{pairing}}$ | $\epsilon' \approx \epsilon$<br>$t' \approx t + q_K(4T_e + SR_q)$ |
| ECIES-KEM | ECDH$^{\mathsf{pairing}}$ | $\epsilon' \approx \epsilon$<br>$t' \approx t + q_K(4T_e + SR_q)$ |
| PSEC-KEM | ECDH$^{\mathsf{pairing}}$ | $\epsilon' \approx \epsilon$<br>$t' \approx t + 2(q_K + q_D)T_e$ |

Table 3: Security results over a pairing curve

**Hardness of the** ECDH$^{\mathsf{pairing}}$ **problem.** When working with pairing curves we are restricting the set from which the elliptic curves are drawn, and then "some randomness" is lost with respect to the original key generation algorithm in these KEMs. Therefore, we obtain a different ECDH problem, which we have called ECDH$^{\mathsf{pairing}}$ problem, and that could be easier to solve than the ECDH problem. Must we trust the hardness of the ECDH$^{\mathsf{pairing}}$ problem? We answer to this question positively taking into account the current status of ECDL problems over pairing curves in cryptography research. As the survey [DBS04] shows, they are being intensively applied to proof the security of new appealing protocols. The new assumptions arising in these protocols are at least stronger than the assumption that ECDH$^{\mathsf{pairing}}$ problem is hard. As a consequence, the trustness on these new assumptions implies trustness on the ECDH$^{\mathsf{pairing}}$ hardness assumption.

# 4 Efficiency and key size considerations

**Computing the security parameter.** Let us assume that the IND-CCA security of any of these KEMs is $(t, q_D, q_K, \varepsilon)$-broken by some adversary $\mathcal{A}$. Since this adversary can be run repeatedly (with the same input and indepedent internal coin tosses), the expected time to distinguish a real encapsulation from random with advantage roughly 1 is $t/\varepsilon$. Thus, the security parameter of the scheme is $n_{\mathrm{KEM}} = \log(t/\varepsilon) = n + m$, where $n = \log t$ and $m = \log(1/\varepsilon)$.

Usually, $q_D \leq 2^{30}$ (that is, up to one billion decryption queries are allowed), and $q_K \leq t = 2^{55}$. We also consider that evaluating a KDF function is a unit operation (that is, takes the same time as a 3-DES encryption). Using Miller's algorithm, computing a pairing in $E(\mathbb{F}_q)$ with embedding degree $k$ can be done in $\mathcal{O}(k \log q)$ multiplications in $\mathbb{F}_q$ (cf. [Men93]), while computing a square root modulo $q$ takes at most $\mathcal{O}(\log^2 q)$ multiplications in $\mathbb{F}_q$ (cf. [Coh93]). Assuming that a multiplication in $\mathbb{F}_q$ takes 10 times longer than one hash query, and that $k \approx 10$, we obtain

$$t'_{\mathrm{ECIES}} \approx t + 2^{55} \cdot 10^2 \cdot (4 \log q + \log^2 q) \approx 2^n + 2^{62} \cdot \log^2 q$$

for ACE and ECIES-KEM, and

$$t'_{\mathrm{PSEC}} \approx t + 2^{56} \cdot 10^2 \cdot \log q \approx 2^n + 2^{63} \cdot \log q$$

for PSEC-KEM. In the following, we compute the exact security only for $t = 2^{80}$ for ECIES-KEM and PSEC-KEM, since the result for ACE-KEM is equal to the former. Setting $m = 0$ and $n = 80$, we obtain $n_{\mathrm{ECIES}} = 80$ (respectively $n_{\mathrm{PSEC}} = 80$), that is, a $2^{80}$ security level in each scheme. Let us compute the minimal parameter length to obtain this security level. An advantage roughly 1 in the IND-CCA game implies that the solver computes CDH successfully with probability roughly 1 in time $t'_{\mathrm{ECIES}}$ (respectively $t'_{\mathrm{PSEC}}$). Assuming that $|q| \approx 200$, then

$$t'_{\mathrm{ECIES}} \approx 2^{80} + 2^{62} \cdot 2^{15} \quad \text{and} \quad t'_{\mathrm{PSEC}} \approx 2^{80} + 2^{63} \cdot 2^{8}.$$

Both reductions are pretty meaningful and then, to get a $2^{80}$ security level on any of these KEMs a group $G$ is enough with at least a $2^{80}$ security of the $\mathrm{ECDH}^{\mathsf{pairing}}$ problem. If we make the *additional assumption* that the $\mathrm{ECDH}^{\mathsf{pairing}}$ and $\mathrm{ECDL}^{\mathsf{pairing}}$ problems have comparable security, then we need a group $G$ with $149 \leq |p| \leq 165$ following [LV01]. In the case of PSEC, this is a great improvement compared to a length of roughly 270 bits needed over a random curve, as we are going to show next.

**PSEC-KEM parameter length over a random curve.** Let us study now PSEC-KEM key size for a random curve using the tools introduced above. Let us assume the IND-CCA security of PSEC-KEM is $(t, q_D, q_K, \epsilon)$-broken by some adversary $\mathcal{A}$. Setting $m = 0$ and $n = 80$ (that is, a $2^{80}$ security level in the scheme), we obtain

$$t' \approx t = 2^{80} \quad \text{and} \quad \epsilon' \approx 1/2^{55}.$$

From the last expression, an advantage roughly 1 in the IND-CCA game implies that the solver computes ECDH succesfully with probability roughly $2^{-55}$ in time $t' = 2^{80}$. However, an algorithm solving ECDH with probability roughly 1 is needed to find the parameter length. Running this algorithm with independent internal coin tosses $2^{55}$ times and returning the most frequent answer, ECDH is solved with probability roughly 1. The computational effort needed to do this is $2^{55} \cdot 2^{80} = 2^{135}$. Assuming that ECDH and ECDL problems have equivalent hardness over a random elliptic curve, we conclude that PSEC-KEM needs a subgroup $G_p$ with $|p| \approx 270$, since the best attack known is using the Pollard $\rho$ method.

**Security reduction to ECDL.** Using a technique due to Maurer [Mau94], one can build certain elliptic curves with a cyclic group $G_p$ for which ECDH and ECDL problems are equivalent. As it was claimed in the previous section, the running time of this reduction is $\mathcal{O}(B \cdot (\log(p)^2)$ group operations in $G_p$ and field operations in $\mathbb{F}_p$, and $\mathcal{O}(\log(p)^3)$ calls to the ECDH solver for $G_p$ [MW00]. Since in our case the computation of ECDH instances is by far the most expensive operation, the reduction to the ECDL problem can be carried out with a $2^{22}$ factor decrease in security for all three schemes, therefore with a total time of $2^{80} \cdot 2^{22}$. Due to this somewhat small factor, the security of the scheme and the ECDL problem are tightly related. This allows to conclude that all three KEMs achieve provable security in the RO model, with the $2^{80}$ IND-CCA bound, in a group $G_p$ with a $2^{102}$ security of the ECDL problem, provided that the ECDL to ECDH reduction of [Mau94] holds for this group.

This theoretical equivalence gives us a good indication about the hardness of the ECDH$^{\mathsf{pairing}}$ problem, since the reduction factor is small enough.

| Curves | Related Problem | Assumptions | Minimal security level |
|---|---|---|---|
| Pairing curves | ECDH$^{\mathsf{pairing}}$ | RO | $2^{80}$ |
| Maurer pairing curves | ECDL$^{\mathsf{pairing}}$ | RO | $2^{103}$ |

Table 4: Discrete log KEMs for the $2^{80}$ security bound

**Performance.** It is now time to study the performance of each scheme over pairing curves. Since all three security reductions are very tight, we have seen that a $2^{80}$ IND-CCA security is achieved under a $2^{80}$ security level for the ECDH$^{\mathsf{pairing}}$ problem. Assuming that ECDH$^{\mathsf{pairing}}$ and ECDL$^{\mathsf{pairing}}$ problems have comparable security in a pairing curve, and that the embedding degree is large enough to keep the DL infeasibility in $\mathbb{F}_{q^k}$ (in which case, the best attack known is to use the Pollard $\rho$ method in $G_p$), a pairing curve $E_{a,b}(\mathbb{F}_q)$ with a group $G_p$ with $|p| \approx 160$ is needed. However, as explained in the next section, the state of the art in pairing curves prevents us for claiming that in general $|p| \approx |q|$, but $|p| \leq |q| \leq 2\,|p|$.

In the following section we quote some methods for generating pairing curves where $|p| \approx |q| \approx 160$. With these values, the performance features of ACE-KEM and ECIES-KEM are equivalent to those of Table 2, while in PSEC-KEM $(K, C)$ length is reduced from 66 to 52 bytes, and public/secret keys are reduced from 34 to 20 bytes, thus obtaining a great improvement. From these values, we easily see that ECIES-KEM presents the best performance in every feature. Since all three KEMs base their security on the same problem, that is, the ECDH$^{\mathsf{pairing}}$ problem, we conclude that ECIES-KEM should be considered the best option among these KEMs if they are implemented over pairing curves.

## 5 Generating suitable pairing curves

In this section we indicate how to generate curves in which the schemes can be performed, and we also discuss why they are suitable. Our aim is to find out how to generate pairing curves $E_{a,b}(\mathbb{F}_q)$ to perform the schemes and where the ECDH$^{\mathsf{pairing}}$ problem is assumed to be hard. We start by describing the conditions that a candidate curve must hold. In the first place, we want pairing curves with a small embedding degree $k$, in order to obtain an efficient pairing computation. However, we cannot use embedding degrees as small as possible: we must take into account that the field $\mathbb{F}_{q^k}$ has to be large enough to fit into the required security level. In our case, we are looking for a $2^{80}$ security level of the DL problem, which corresponds to $1024 \leq |q^k| \leq 1464$, using the estimates by Lenstra and Verheul [LV01] and the parameters used nowadays.

Unfortunately, curves with a small embedding degree are extremely rare, as shown in [BK98]. An exception are supersingular elliptic curves [Men93], which have $k \leq 6$. However, inasmuch as we are looking for small security parameters, only supersingular elliptic curves with $k = 6$ can fit our purposes. Nevertheless, it is not easy to generate

such curves over prime finite fields, and the popular constructions use the field $\mathbb{F}_{3^m}$ (cf. [Gal01]).

Following [Gal04], an algorithm for generating curves with arbitrary $k$ and with a large prime factor $p$ of any form is proposed in [CP01]. Although it solves the embedding degree problem, it has the drawback that it produces curves with $q > p^2$. For instance, this means that for $k = 10$ and $|p| \approx 160$, the algorithm returns a curve $E_{a,b}(\mathbb{F}_q)$ with $|q| > 320$. It is an active area of research to obtain pairing curves in which $|q| \approx |p|$ and $k \geq 7$. First steps in this direction have been taken, for instance, in [DEM02, BW03, SB04]. In [SB04] a method for generating pairing curves with $|p| \approx |q|$ and $k = 6$ is described, while in [BW03] curves are generated with $k \geq 7$ and $|q| << 2|p|$.

# 6 Conclusions

In this paper we have studied the performance and security properties of the elliptic curve KEMs proposed to be standardized when they are implemented on pairing curves. First of all, we have summarized the properties claimed in the recent literature, and we have fixed some inexact results. One important contribution has been to point out that there is no evidence supporting that the 160 bits key size used up to now for PSEC-KEM over a random curve results in a secure implementation. In fact, a rigorous security analysis shows that a secure key size must be near 270 bits. This new value has a negative impact on PSEC-KEM performance. Another contribution has been to show that, on the one hand, despite their different behaviour from a security point of view in a random curve, the elliptic curve KEMs present a very tight security reduction to the $\text{ECDH}^{\mathsf{pairing}}$ problem; and, on the other hand, to suggest that ECIES-KEM should be considered the best option among these KEMs in environments using pairing curves (where the ECDDH assumption does not hold). This could seem quite suprising, since ECIES-KEM in a random curve was not included in the portfolio of recommended cryptographic primitives by NESSIE [NES03b].

We have discussed the hardness of the $\text{ECDH}^{\mathsf{pairing}}$ problem taking into account the current state of the art in protocol design and cryptanalysis. Even though the $\text{ECDH}^{\mathsf{pairing}}$ problem may be easier than the standard ECDH problem, it is harder than the usual problems considered in provably secure schemes using bilinear maps. In contrast to the gap-ECDH problem, which in the context of KEMs is mainly a theoretical tool (there is no ECDDH solver anywhere), $\text{ECDH}^{\mathsf{pairing}}$ is defined within the cryptographic practice. We think that if ECDL problems over pairing curves are considered secure to built on cryptographic protocols (which is the current trend), we could use pairing curves to implement protocols that were designed to be implemented in a more general context but which present better performance properties when restricted to pairing curves. A major breakthrough in the efficient implementation of these KEMs would be to find methods to generate pairing curves with embedding degree at least 7 and $|q| \approx |p|$ .

Finally, we hope that the results presented here along with the known properties in the literature will help to find the most suitable contexts for the use of each elliptic curve KEM.

**Acknowledgements.** We would like to thank Alex W. Dent for helpul comments

on an early draft of this paper.

## References


[AM93] A.O.L. Atkin and F. Morain. Elliptic curves and primality proving. *Mathematics of Computation*, 61:29–67, 1993.

[BF01] D. Boneh and M. Franklin. Identity-Based encryption from the Weil pairing. In *Advances in Cryptology – CRYPTO ' 01*, volume 2139 of *Lecture Notes in Computer Science*, pp. 213–229, 2001.

[BK98] R. Balasubramanian and N. Koblitz. The improbability that an elliptic curve has subexponential discrete log problem under the Menezes-Okamoto-Vanstone algorithm. *Journal of Cryptology*, 11(2):141–145, 1998.

[BR93] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM CCS*, pp. 62–73. ACM Press, 1993.

[BW03] F. Brezing and A. Weng. Elliptic curves suitable for pairing based cryptography. Cryptology ePrint Archive, Report 2003/143, 2003. `http://eprint.iacr.org/`.

[Coh93] H. Cohen. *A Course in Computational Algebraic Number Theory*, volume 138 of *Graduate Texts in Mathematics*. Springer, 1993.

[CP01] C. Cocks and R.G.E. Pinch. Identiy-based cryptosystems based on the Weil pairing, 2001. Unpublished manuscript.

[CS] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. To appear at *SIAM Journal of Computing*. Available at `www.shoup.net`.

[DBS04] R. Dutta, R. Barua, and P. Sarkar. Pairing-based cryptography : A survey. Cryptology ePrint Archive, Report 2004/064, 2004. `http://eprint.iacr.org/`.

[DEM02] R. Dupont, A. Enge, and F. Morain. Building curves with arbitrary small mov degree over finite prime fields. Cryptology ePrint Archive, Report 2002/094, 2002. `http://eprint.iacr.org/`.

[Den02] A.W. Dent. ECIES-KEM vs. PSEC-KEM. Technical Report `NES/DOC/RHU/WP5/028/2`, NESSIE, 2002.

[FR94] G. Frey and H.G. Rück. A remark concerning $m$-divisibility and the discrete logarithm problem in the divisor class group of curves. *Mathematics of Computation*, 62:865–874, 1994.

[Gal01] S. Galbraith. Supersingular curves in cryptography. In *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pp. 495–513, 2001.

[Gal04] S. Galbraith. Pairings, 2004. Unpublished manuscript.

[GJ03] E.J. Goh and S. Jarecki. A signature scheme as secure as the Diffie-Hellman problem. In *Advances in Cryptology – EUROCRYPT ' 2003*, volume 2656 of *Lecture Notes in Computer Science*, pp. 401–415, 2003.

[JN03] A. Joux and K. Nguyen. Separating Decision Diffie-Hellman from Diffie-Hellman in cryptographic groups. *Journal of Cryptology*, 16(4):239–247, 2003.

[Jou00] A. Joux. A one round protocol for tripartite Diffie-Hellman. In *ANTS 2000*, volume 1838 of *Lecture Notes in Computer Science*, pp. 385–394, 2000.

[LV01] A.K. Lenstra and E.R. Verheul. Selecting cryptographic key sizes. *Journal of Cryptology*, 14(4):255–293, 2001.

[LZ94] G.J. Lay and H.G. Zimmer. Constructing elliptic curves with given group order over large finite fields. In *ANTS '94*, volume 877 of *Lecture Notes in Computer Science*, pp. 250–263, 1994.

[Mau94] U. Maurer. Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms. In *Advances in Cryptology — CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pp. 271–281, 1994.

[Men93] A. Menezes. *Elliptic Curve Public Key Cryptosystems*, volume 234 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, 1993.

[MOV93] A.J. Menezes, T. Okamoto, and S.A. Vanstone. Reducing elliptic curve logarithms to a finite field. *IEEE Transactions on Information Theory*, 39:1639–1646, 1993.

[MSV04] A. Muzereau, N.P. Smart, and F. Vercauteren. The equivalence between the DHP and DLP for elliptic curves used in practical applications. *LMS J. Comput. Math*, 7:50–72, 2004.

[MW00] U. Maurer and S. Wolf. The Diffie-Hellman protocol. *Designs, Codes, and Cryptography*, 19:147–171, 2000.

[NES03a] NESSIE. NESSIE security report. version 2.0, 2003. `http://www.cryptonessie.org/`.

[NES03b] NESSIE. Portfolio of recommended cryptographic primitives. NESSIE consortium., 2003. `http://www.cryptonessie.org/`.

[OP01] T. Okamoto and D. Pointcheval. The gap-problems: a new class of problems for the security of cryptographic schemes. In *PKC 2001*, volume 1992 of *Lecture Notes in Computer Science*, pp. 104–118, 2001.

[Pol78] J.M. Pollard. Monte carlo methods for index computation mod $p$. *Mathematics of Computation*, 32:918–924, 1978.

[SB04] M. Scott and P.S.L.M Barreto. Generating more MNT elliptic curves. Cryptology ePrint Archive, Report 2004/058, 2004. `http://eprint.iacr.org/`.

[Sho00] V. Shoup. Using hash functions as a hedge against chosen ciphertext attacks. In *Advances in Cryptology – EUROCRYPT ' 2000*, volume 1807 of *Lecture Notes in Computer Science*, pp. 275–288, 2000.

[Sho01] V. Shoup. A proposal for an ISO standard for public key encryption. Technical Report 2.1, 2001.

[Sho04] V. Shoup. Draft ISO/IEC 18033-2: An emerging standard for public-key encryption. Technical report, ISO/IEC, 2004.

[Sma99] N. Smart. The discrete logarithm problem on elliptic curves of trace one. *Journal of Cryptology*, 12(3):193–196, 1999.