Evaluation of Security Level of Cryptography: The Revised Version of PSEC-2 (PSEC-KEM)

Alfred Menezes
University of Waterloo
Contact: ajmeneze@uwaterloo.ca

December 14, 2001

Contents

1 Executive Summary

This report evaluates the PSEC-KEM elliptic curve public-key encryption scheme as specified in [26]. PSEC-KEM is provably secure in the random oracle model under the assumption that the (computational) Diffie-Hellman problem for elliptic curves is intractable. We also compare the security and performance of PSEC-KEM with that of ACE-KEM and ECIES-KEM.

2 Introduction

Elliptic curve cryptosystems were first proposed by Neal Koblitz [15] and Victor Miller [21] in 1985. In principle, any discrete logarithm cryptographic scheme can be implemented using an elliptic curve group as the underlying algebraic structure. This includes the ElGamal encryption scheme [11], which was the first discrete logarithm public-key encryption scheme. However, the basic ElGamal encryption scheme is well-known to be insecure in the sense that it does not meet stringent security requirements of semantic security against adaptive chosen-message attacks. (For further discussion on why the basic ElGamal encryption is insecure, see [6].) This has motivated many researchers to propose variants of the ElGamal encryption scheme that are both efficient and can be proven secure under assumptions that can be reasonably justified.

HISTORY OF PSEC-KEM. PSEC-1, PSEC-2 and PSEC-3 are three kinds of ElGamal-like public-key encryption schemes that were submitted to the CRYPTREC 2000 project. The proposers have since withdrawn the PSEC-1 and PSEC-3 submissions, and modified PSEC-2 to address some observations made by Shoup [33]. Throughout this document, we refer only to this revised version of PSEC-2 which is now called PSEC-KEM and is completely specified in [26].

COMPARING PSEC-KEM TO OTHER ENCRYPTION SCHEMES. Statements regarding the security and performance of PSEC-KEM are more meaningful when they are compared to the security and performance of other encryption schemes. PSEC-KEM can be compared to any public-key encryption scheme, but it is advisable to compare it to some accepted and well-known schemes. In addition, it should be compared to schemes that have somewhat similar characteristics, and that would be appropriate for similar applications. The most suitable such schemes are ACE-KEM (Advanced Cryptographic Engine–Key Encapsulation Method) and ECIES-KEM (Elliptic Curve Integrated Encryption Scheme–Key Encapsulation Method).

ORGANIZATION. The remainder of this report is organized as follows. In Section 3, we describe the PSEC-KEM key generation, encryption and decryption procedures. The security of PSEC-KEM is analyzed in Section 4. Here we consider the provable security aspects of PSEC-KEM, the hardness of the underlying Diffie-Hellman problem, and sizes of domain parameters. In Section 5, we compare the performance and security of PSEC-KEM with two other ElGamal-like elliptic curve public-key encryption schemes, namely ACE-KEM and ECIES-KEM. Our conclusions are stated in Section 6.

3 Description of PSEC-KEM

PSEC-KEM is a *key encapsulation method* (KEM). Its goal is to select a symmetric key K and securely transport it to some intended recipient (see Section 4). A KEM differs from a public-key encryption scheme in that the input to the encryption algorithm is just the recipient's public key. The encryption algorithm outputs a symmetric key K and ciphertext c. The decryption algorithm, on input c, recovers K.

Key encapsulation methods are intended to be used with a *data encapsulation method* (DEM). The latter uses the symmetric key established with a KEM to encrypt and authenticate messages of arbitrary lengths. The intent is that when a secure KEM is used with a secure DEM, the resulting *hybrid* encryption scheme is a public-key encryption scheme that is semantically secure against adaptive chosen-ciphertext attacks.

In this section, we describe the PSEC-KEM scheme from [26]. We note that the PSEC specification does not propose a DEM[1]. Thus, the PSEC-KEM specification cannot be used (without modification) to encrypt arbitrarily-long messages.

3.1 PSEC-KEM Domain Parameters

The PSEC-KEM scheme as described in [26] has certain parameters that (i) an encryptor and decryptor must agree upon and mutually support (for interoperability); and (ii) require the support of advanced implementation techniques. These parameters are typically fixed for a *domain* of intercommunicating users, and should be carefully selected since they affect both security and performance.

1. The *elliptic curve domain parameters* consist of:
   - An elliptic curve E defined over a finite field $\mathbb{F}_q$.
   - A method for representing elements of $\mathbb{F}_q$.
   - A point $G \in E(\mathbb{F}_q)$ of prime order.
   - The order n of G. The bitlength of n is denoted by k.
2. Another domain parameter is the *key derivation function* KDF. The PSEC-KEM specification recommends the key derivation function KDF1 which is constructed from a hash function by concatenating the values produced by hashing the input together with a 32-bit counter that is incremented for each hash operation. The PSEC-KEM specification recommends the hash function SHA-1 [22]. Since there is only one recommended hash

[1] Shoup [33] has proposed a DEM that uses a symmetric-key encryption scheme S to encrypt the plaintext, and a MAC algorithm M to authenticate the ciphertext. The DEM is provably secure under the assumptions that S is secure against passive attacks, and M is a secure one-time MAC algorithm.

function, it is not necessary to include any information about KDF as part of the domain parameters. However, if the specification is to allow upgrades to more secure hash functions such as SHA-256, SHA-384 and SHA-512 [24], then a hash function identifier should be included as a domain parameter.

3. The bitlength l of the random seed.

The establishment or conveyance of domain parameters is generally outside the scope of a public-key encryption scheme.

3.2 PSEC-KEM Key Generation

To generate a key pair for the elliptic curve domain parameters $(E, \mathbb{F}_q, G, n)$, an entity A does the following:

1. Select a random integer d from the interval $[0, n-1]$.
2. Compute $Q = dG$.
3. A's public key is Q.
4. A's private key is d.

NOTE. A method should be included for *validating* the elliptic curve domain parameters $(E, \mathbb{F}_q, G, n)$ and the public key Q. Such methods are described in several standards such as ANSI X9.63 [2].

3.3 PSEC-KEM Encryption

To generate and encrypt a symmetric key K for an entity A whose public key is Q, an entity B does the following:

1. Randomly select $r \in \{0,1\}^l$.
2. Compute $(t', K) = \mathrm{KDF}(r)$, where t' has bitlength $k+128$ and K has the desired bitlength.
3. Compute $t = t' \bmod n$.
4. Compute the elliptic curve points $T = tG$ and $U = tQ$.
5. Compute $s = r \oplus \mathrm{KDF}(T, U)$.
6. The ciphertext is (s, T).

NOTE. Rationale should be provided for why t' is chosen to have bitlength $k+128$ and not just k.

3.4 PSEC-KEM Decryption

To recover a symmetric key from ciphertext (s,T), an entity A with private key d does the following:

1. Compute the elliptic curve point $U = dT$.
2. Compute $r = s \oplus \mathrm{KDF}(T,U)$.
3. Compute $(t',K) = \mathrm{KDF}(r)$, where t' has bitlength $k+128$ and K has the desired bitlength.
4. Compute $t = t' \bmod n$.
5. Compute the elliptic curve point $T' = tG$.
6. Verify that $T' = T$; if not then output "invalid" and stop.
7. The symmetric key is K.

4 Security Analysis

The security objective of any public-key encryption scheme is *semantic security against adaptive chosen-ciphertext attacks* [12, 31]. Informally, this means that an adversary can learn nothing about the plaintext corresponding to a given ciphertext c, even when the adversary is allowed to obtain the plaintexts corresponding to ciphertexts (not equal to c) of its choice. Another equivalent formulation is the following. The goal of an adversary who launches an attack against a legitimate entity is to find two plaintexts for which the adversary is able to distinguish whether a challenge ciphertext is the encryption of the first plaintext or the second plaintext. To achieve this goal, the adversary is permitted to access a decryption oracle that will decrypt any ciphertext of the adversary's choosing, with the exception of the challenge ciphertext. The encryption scheme is semantically secure if no such adversary exists.

The security definition for a key encapsulation method is slightly different from that of a public-key encryption scheme. Here, an adversary is given a target key-ciphertext pair $(\overline{K},c)$, where c is a ciphertext generated by the legitimate entity, and $\overline{K}$ is either the symmetric key K corresponding to c or a symmetric key K' which has been randomly generated independently of K. The adversary's task is to decide whether $\overline{K} = K$ or $\overline{K} = K'$. To achieve this goal, the adversary is permitted to access a decryption oracle that will decrypt any ciphertext of the adversary's choosing except for c itself. The KEM is *semantically secure* if no such adversary exists. We denote this notion of security, which is due to Shoup [33], by IND-CCA (indistinguishability against chosen-ciphertext attacks).

The PSEC-KEM submission [27] (see also [33]) includes a proof that PSEC-KEM is IND-CCA (as a key encapsulation method) in the random oracle model under the assumption that the (computational) Diffie-Hellman problem for elliptic curves (ECDHP) is intractable. By

"random oracle" we mean that the KDF is modeled as a public random function [3]. The ECDHP is studied in Section 4.4.

4.1 Random Oracle Assumption

The security proof for PSEC-KEM takes place in the random oracle model [3], and therefore does not imply security in the real world where the hash function is no longer a random function. This perspective was given some credence by Canetti, Goldreich and Halevi [8] who provided examples of signature and encryption schemes which they proved secure in the random oracle model, but which are insecure with *any* reasonable instantiation of the random function. However, the encryption and signature schemes in [8] are rather contrived, and a security proof in the random oracle model is now widely accepted as providing valuable *heuristic* evidence for security. Additionally, the security proof of PSEC-KEM implies that any attack on PSEC-KEM must either solve the Diffie-Hellman problem, or exploit some properties of the hash function employed.

4.2 Tightness of the Reduction

Security proofs typically work by providing a *reduction* from the solution of a problem P to the breaking of a cryptographic scheme S. That is, one shows how P can be solved if one is given an oracle for breaking S. In the case of the PSEC-KEM proof, S is PSEC-KEM while P is the ECDH problem. The success of the reduction is measured by the tightness of the reduction. More precisely, a reduction is *tight* if when an adversary can break S in time t with probability ε, then the reduction algorithm that solves P runs in time t' with success probability ε', where $t' \approx t$ and $\varepsilon' \approx \varepsilon$. A tight reduction is important because one then has the assurance that the security level of S is very closely related to the security level of P. Thus one can select security parameters in S to be of the same size as the parameters that make P intractable against all known attacks.

The security proof of PSEC-KEM has a very tight reduction (see Theorem 3.1 of [27]). Thus, for example, one can use a 160-bit parameter n and be assured that breaking PSEC-KEM will require roughly the same amount of work as it takes to solve the ECDH problem for a 160-bit n.

4.3 Elliptic Curve Discrete Logarithm Problem

The *elliptic curve discrete logarithm problem* (ECDLP) is the following: Given elliptic curve domain parameters E, $\mathbb{F}_q$, G, n and a point $Q = dG$ (where $0 \leq d \leq n-1$), find d.

Intractability of the ECDLP is clearly *necessary* for the security of PSEC-KEM. For, if the ECDLP were easy, then an adversary could compute the private key d from a public key Q and

thereafter decrypt all ciphertexts intended for the legitimate owner of Q. Intractability of the ECDLP is also *sufficient* for the security of PSEC-KEM (in the random oracle model) since it implies the intractability of the ECDHP (see Section 4.4 for a more precise formulation of this statement).

The ECDLP has been extensively studied for the last sixteen years and no general-purpose subexponential-time algorithm has been discovered. The best general-purpose algorithm known is Pollard's rho algorithm [30, 28] which has an expected running time of $\sqrt{\pi n}/2$ steps and can be effectively parallelized. Selecting n to be at least 2^{160} provides effective resistance against this attack.

There are various special-purpose attacks known on the ECDLP—these attacks dictate that some care must be exercised when selecting elliptic curve domain parameters for PSEC-KEM. In particular, elliptic curves where either (i) n divides $q^k - 1$ for some $1 \leq k \leq 20$; or (ii) $n = q$ should definitely not be used as the ECLDP in these cases can be solved in subexponential time (or better) by the Weil/Tate pairing attack and the prime field anomalous curve attack, respectively. Furthermore, due to recent developments on Frey's Weil descent attack, one should be cautious about using elliptic curves over fields $\mathbb{F}_{2^m}$ where m is composite, and over fields $\mathbb{F}_{p^m}$ where p is odd and $m = 5$ or $m = 7$—preferably these fields should be avoided altogether.

For a detailed survey of the state-in-the-art in algorithms for the ECDLP, see the CRYPTREC report [10].

4.4 DH, DDH and GAP-DH Problems

Let G be a (cyclic) group of prime order n, and let $g \in G$ be a generator. We recall the definition of the DL, DH, DDH and GAP-DH problems in G.

1. *Discrete Logarithm Problem* (DLP): Given g, n and g^x (where $x \in_R [0, n-1]$), find x.
2. *Diffie-Hellman Problem* (DHP): Given g, n, g^x and g^y (where $x, y \in_R [0, n-1]$), find g^{xy}.
3. *Decision Diffie-Hellman Problem* (DDHP): Given g, n, g^x, g^y (where $x, y \in_R [0, n-1]$) and g^z (where either $z \in_R [0, n-1]$ with probability $\frac{1}{2}$ or $z = xy \bmod n$ with probability $\frac{1}{2}$), decide whether or not $z \equiv xy \pmod{n}$. See [5] for a survey of the DDHP and its cryptographic applications.
4. *Gap Diffie-Hellman Problem* (GAP-DHP): Solve the DHP given an oracle for solving the DDHP. This problem was first proposed by Okamoto and Pointcheval [29].

It is clear that the DDHP reduces in polynomial time to the DHP, and the DHP reduces in polynomial time to the DLP. Polynomial time reductions of DLP to DHP, or of DHP to DDHP, are not known. There are, however, some groups G in which the DLP is believed to be intractable but where the DLP can be reduced in polynomial time to the DHP in G; examples include arbitrary cyclic groups of order n where $\phi(n)$ is smooth (see [4]). There are also some groups

in which no polynomial time algorithms for the DHP are known, but where the DDHP can be solved in polynomial time; examples include certain supersingular elliptic curves and certain elliptic curves of trace 2 (see [13], [14] and [35]).

Maurer [18, 19] proved the polynomial time equivalence of the DLP and DHP in groups G of prime order n under the assumption that certain elliptic curves exist and can be efficiently found. Namely, if one is given an elliptic curve E defined over $\mathbb{F}_n$ such that $E(\mathbb{F}_n)$ is cyclic and $\#E(\mathbb{F}_n)$ is $(\log n)^c$-smooth (i.e., the largest prime factor of $\#E(\mathbb{F}_n)$ is at most $(\log n)^c$), then the DLP in G can be reduced in polynomial time to the DHP in G. The coefficients of the elliptic curve E comprise a polynomial size "hint" depending only on n that, once known, allows one to solve *any* instance of the DLP in *any* group G of order n in polynomial time given a Diffie-Hellman oracle for G. It is not known if this hint exists; however, heuristic arguments about the distribution of smooth integers in the Hasse interval $[n-2\sqrt{n}+1, n+2\sqrt{n}+1]$ suggest that such hints do exist. Even if the hint does exist, it is not known how to find it in polynomial time—exhaustive search would, in general, take $O(n^c)$ time. Nevertheless, Maurer's result can be viewed as providing some evidence for the equivalence of the DLP and DHP.

Of more relevance to the security of PSEC-KEM is the result of Boneh and Lipton [7] which states that if the ECDLP cannot be solved in $L_{\frac{1}{2}}[n]$ subexponential time, then the ECDHP also cannot be solved in $L_{\frac{1}{2}}[n]$ time. Since it is widely believed that the ECDLP cannot be solved in subexponential time (see Section 4.3), this provides very strong evidence for the hardness of the ECDHP[2].

Finally, we note that Shoup [32] has proved lower bounds of the form $\Omega(\sqrt{n})$ for the DLP, DHP and DDHP in generic groups of prime order n. (A generic group is one whose elements have random labelings and which comes equipped with an efficient oracle for performing the group operation.) Shoup's result provides some evidence for the intractability of the DLP, DHP and DDHP in groups that are used in cryptography. It can also be proven that the DHP cannot be reduced in polynomial time to the DDHP in the generic group model. This provides some evidence for the intractability of the GAP-DHP.

4.5 Parameter Recommendations

The PSEC-KEM specification [26] sets the following minimum security requirements:

- $k \geq 160$.
- $l \geq 128$.

In addition, the following domain parameters are recommended:

[2] Since an $L_q[\frac{1}{2}]$ algorithm *is* known for the discrete logarithm problem in the multiplicative group $\mathbb{F}_q^*$ of a finite field, the Boneh-Lipton result does *not* apply, and hence we do not have a proof of equivalence of the DLP and DHP for such groups. This is one important reason why an elliptic curve group is more attractive than the multiplicative group of a finite field for use in protocols whose security is based on the hardness of the DHP.

- $k = 160$.
- KDF = KDF1(SHA-1).
- $l = 160$.

These recommendations are appropriate for providing an 80-bit level of security. What is lacking are concrete recommendations for the elliptic curve domain parameters. In particular, the known classes of weak elliptic curves should be explicitly prohibited (see [10]). Also, further details should be provided on the significance of the parameter l and the rationale for the choice $l = 160$.

SELECTING PARAMETERS FOR LONG-TERM SECURITY. The recommended parameters only provide one (fixed) level of security. If PSEC-KEM is to be used for long-term security, it is imperative that the PSEC-KEM specification provide guidance on the selection of domain parameters (n, elliptic curve, hash function, l) corresponding to increasing levels of security. For example, recommendations could be provided for achieving 80-bit, 112-bit, 128-bit, 192-bit and 256-bit levels of security (corresponding to the security levels believed to be possessed by SKIPJACK, Triple-DES, AES-128, AES-192 and AES-256 [25]). Sample elliptic curves for these security levels are provided in FIPS 186-2 [23], while the hash functions SHA-256, SHA-384 and SHA-512 [24] provide 128, 192 and 256 bits of security, respectively.

5 Comparison With ACE-KEM and ECIES-KEM

In this section, we compare the performance and security of PSEC-KEM with ACE-KEM and ECIES-KEM, which are two elliptic curve key encapsulation methods that offer the same functionality as PSEC-KEM. ACE-KEM is a modification of the Cramer-Shoup public-key encryption scheme [9], while ECIES-KEM is a modification of the ECIES public-key encryption scheme [1, 34]. ACE-KEM and ECIES-KEM are fully specified in [33]. All of PSEC-KEM, ACE-KEM and ECIES-KEM can be used together with a data encapsulation method to yield a full-fledged public-key encryption scheme capable of encrypting messages of arbitrary lengths.

In the descriptions of ACE-KEM and ECIES-KEM below, domain parameters are elliptic curve parameters $(E, \mathbb{F}_q, G, n)$, a hash function H, and a key derivation function KDF. To simplify the description of the schemes, we assume that the *cofactor* $h = \#E(\mathbb{F}_q)/n$ is 1; that is, $\#E(\mathbb{F}_q) = n$.

5.1 ACE-KEM

Key Generation. To generate a key pair, an entity A does the following:

1. Select $d, x, y, z \in_R [0, n-1]$.
2. Compute $Q = dG$, $X = xG$, $Y = yG$ and $Z = zG$.

3. A's public key is (Q, X, Y, Z).
4. A's private key is (d, x, y, z).

Encryption. To generate and encrypt a symmetric key K for an entity A whose public key is (Q, X, Y, Z), an entity B does the following:

1. Select $t \in_R [0, n-1]$.
2. Compute $T = tG$, $U_1 = tQ$ and $U_2 = tZ$.
3. Compute the integer $\alpha = H(T, U_1)$.
4. Compute the integer $t' = \alpha \cdot t \bmod n$.
5. Compute $V = tX + t'Y$.
6. Compute $K = \mathrm{KDF}(T, U_2)$.
7. The ciphertext is (T, U_1, V); the symmetric key is K.

Decryption. To recover a symmetric key from ciphertext (T, U_1, V), an entity A with private key (d, x, y, z) does the following:

1. Compute the integer $\alpha = H(T, U_1)$.
2. Compute the integer $r = x + \alpha y \bmod n$.
3. Compute $U_1' = dT$ and $V' = rT$.
4. If $U_1' \neq U_1$ or if $V' \neq V$, then output "invalid" and stop.
5. Compute $U_2 = zT$.
6. Compute the symmetric key $K = \mathrm{KDF}(T, U_2)$.

5.2 ECIES-KEM

Key Generation. To generate a key pair, an entity A does the following:

1. Select $d \in_R [0, n-1]$.
2. Compute $Q = dG$.
3. A's public key is Q.
4. A's private key is d.

Encryption. To generate and encrypt a symmetric key K for an entity A whose public key is Q, an entity B does the following:

1. Select $t \in_R [1, n-1]$.
2. Compute $T = tG$ and $U = tQ$.
3. Compute $K = \text{KDF}(T, U)$.
4. The ciphertext is T; the symmetric key is K.

Decryption. To recover a symmetric key from ciphertext T, an entity A with private key d does the following:

1. Compute $U = dT$.
2. Compute the symmetric key $K = \text{KDF}(T, U)$.

5.3 Comparison

This section compares the performance and security attributes of PSEC-KEM, ACE-KEM and ECIES-KEM.

PERFORMANCE. The dominant computational steps in all three KEMs are the elliptic curve point multiplication operations. The number of point multiplications required for encryption and decryption is summarized in Table 1. Note that the data in Table 1 does not give the complete picture since some of the point multiplications counted can be performed significantly faster than others. For example, the computation of tG can be sped up if G is a fixed domain parameter and memory is available for precomputed multiples of G. Also, two of the point multiplications in ACE-KEM encryption ($V = tX + t'Y$) can be sped up using Shamir's trick (see Algorithm 14.88 in [20]).

	PSEC-KEM	ACE-KEM	ECIES-KEM
Encryption	2	5	2
Decryption	2	3	1

Table 1: Number of elliptic curve point multiplications for encryption and decryption.

We conclude that ACE-KEM encryption and decryption is significantly slower than their PSEC-KEM and ECIES-KEM counterparts. PSEC-KEM and ECIES-KEM offer similar performance with ECIES-KEM decryption being a little faster than PSEC-KEM decryption (the extra point multiplication in PSEC-KEM decryption is $T' = tG$—here G is fixed, and so this operation can be sped up considerably).

SECURITY. In the following, we only consider the elliptic curve versions of the DH, DDH and GAP-DH problems. For each problem P, the "P assumption" is that P is intractable. We say that a KEM is secure if it is IND-CCA (see Section 4).

Recall that PSEC-KEM has been proven secure in the random oracle model under the DH assumption.

ACE-KEM has been proven secure under the DDH assumption (plus some reasonable assumptions on the hash function and KDF employed) [9, 33]. ACE-KEM has also been proven secure in the random oracle model under the DH assumption [33].

ECIES-KEM has been proven secure under the assumption that a non-standard interactive variant of the DHP is intractable [1]. It has also been proven secure in the random oracle model under the GAP-DH assumption [33].

The security proof for PSEC-KEM, security proof (under the DDH assumption) for ACE-KEM, and the security proof (under the GAP-DH assumption) for ECIES-KEM are all tight—thus one cannot judge one of these KEMs to be superior to the others based only on the tightness of the security proofs. Hence, to compare the three KEMs from a security point of view, one has to examine the relative plausibility of the assumptions made in their security proofs.

Considering only the three proofs in the random oracle model, one can conclude that AES-KEM and PSEC-KEM provide equivalent security assurances, which is greater than that of ECIES-KEM (since the GAP-DH assumption is stronger than the DH assumption). Recall, however, that security proofs in the random oracle model only provide heuristic evidence for security since the assumption that the hash functions are random does not hold for real-world hash functions. If one considers only security proofs in a standard model (i.e., without a random oracle assumption), then ACE-KEM is superior to both ECIES-KEM and PSEC-KEM—the security proof of ECIES-KEM provides limited assurances since the variant of the DH assumption made is non-standard and very strong, while a security proof for PSEC-KEM is not known.

6 Conclusions

PSEC-KEM is an elliptic curve scheme for generating and encrypting symmetric keys.

SECURITY. PSEC-KEM has been proven IND-CCA secure in the random oracle model under the ECDH assumption. Since the ECDHP has, in a certain sense, been proven to be equivalent to the ECDLP, and since the reduction in the security proof is very tight, one can be assured that the security level of PSEC-KEM is closely related to the security level of the ECDLP.

The PSEC-KEM specification should include a data encapsulation method (DEM). Also, guidance should be provided on the choice and size of domain parameters for long-term security.

COMPARISON WITH ACE-KEM AND ECIES-KEM. PSEC-KEM and ECIES-KEM have roughly the same encryption and decryption performance, while ACE-KEM is significantly slower. It is not possible to say with certainty which of the three KEMs are superior from a security point of view because of the differing assumptions made in their security proofs. Roughly speaking though, ACE-KEM provides the greatest assurance. In fact, Shoup [33] has observed

that ACE-KEM is at least as secure as ECIES-KEM because any adversary C_1 of ACE-KEM can be converted to an adversary C_2 of ECIES-KEM which succeeds with the same probability and has approximately the same running time as C_1. PSEC-KEM and ECIES-KEM are roughly comparable since PSEC-KEM has better assurances in the random oracle model, but ECIES-KEM has better assurances in the standard model.

References

[1] M. Abdalla, M. Bellare and P. Rogaway, "The oracle Diffie-Hellman assumption and an analysis of DHIES", *Topics in Cryptology–CT-RSA 2001*, Lecture Notes in Computer Science, **2020** (2001), 143-158.

[2] ANSI X9.63, *Public Key Cryptography for the Financial Services Industry: Elliptic Curve Key Agreement and Key Transport Protocols*, ballot version, May 2001.

[3] M. Bellare and P. Rogaway, "Random oracles are practical: a paradigm for designing efficient protocols", *1st ACM Conference on Computer and Communications Security*, 1993, 62-73. Full version available at http://www.cs.ucdavis.edu/~rogaway/papers/index.html

[4] B. den Boer, "Diffie-Hellman is as strong as discrete log for certain primes", *Advances in Cryptology–Crypto '88*, Lecture Notes in Computer Science, **403** (1990), Springer-Verlag, 530-539.

[5] D. Boneh, "The decision Diffie-Hellman problem", *Algorithmic Number Theory, Proc. Third Intern. Symp., ANTS-III*, Lecture Notes in Computer Science, **1423** (1998), Springer-Verlag, 48-63.

[6] D. Boneh, A. Joux and P. Nguyen, "Why textbook ElGamal and RSA encryption are insecure", *Advances in Cryptology–Asiacrypt 2000*, Lecture Notes in Computer Science, **1976** (2000), Springer-Verlag, 30-43.

[7] D. Boneh and R. Lipton, "Algorithms for black-box fields and their application to cryptography", *Advances in Cryptology–Crypto '96*, Lecture Notes in Computer Science, **1109** (1996), Springer-Verlag, 283-297.

[8] R. Canetti, O. Goldreich and S. Halevi, "The random oracle methodology, revisited", *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing*, 1998, 209-218.

[9] R. Cramer and V. Shoup, "Signature schemes based on the strong RSA assumption", *ACM Transactions on Information and System Security*, **3** (2000), 161-185.

[10] CRYPTREC Report, "Evaluation of security level of cryptography: The elliptic curve discrete logarithm problem", December 14 2001.

[11] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms", *IEEE Transactions on Information Theory*, **31** (1985), 469-472.

[12] S. Goldwasser and S. Micali, "Probabilistic encryption", *Journal of Computer and System Sciences*, **29** (1984), 270-299.

[13] A. Joux, "A one round protocol for tripartite Diffie-Hellman", *Algorithmic Number Theory, Proc. Third Intern. Symp., ANTS-IV*, Lecture Notes in Computer Science, **1838** (2000), Springer-Verlag, 385-393.

[14] A. Joux and K. Nguyen, "Separating decision Diffie-Hellman from Diffie-Hellman in cryptographic groups", preprint, 2001.

[15] N. Koblitz, "Elliptic curve cryptosystems", *Mathematics of Computation*, **48** (1987), 203-209.

[16] A. Lenstra, "Unbelievable security: Matching AES security using public key systems", *Advances in Cryptology–Asiacrypt 2001*, Lecture Notes in Computer Science, **2248** (2001), Springer-Verlag, 67-86

[17] A. Lenstra and E. Verheul, "Selecting cryptographic key sizes", *Public Key Cryptography–Proceedings of PKC 2000*, Lecture Notes in Computer Science, **1751** (2000), Springer-Verlag, 446-465. Full version to appear in *Journal of Cryptology*.

[18] U. Maurer, "Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms", *Advances in Cryptology–Crypto '94*, Lecture Notes in Computer Science, **839** (1994), Springer-Verlag, 271-281.

[19] U. Maurer and S. Wolf, "The Diffie-Hellman protocol", *Designs, Codes and Cryptography*, **19** (2000), 147-171.

[20] A. Menezes, P. van Oorschot and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.

[21] V. Miller, "Uses of elliptic curves in cryptography", *Advances in Cryptology–Crypto '85*, Lecture Notes in Computer Science, **218** (1986), Springer-Verlag, 417-426.

[22] National Institute of Standards and Technology, *Secure Hash Standard (SHS)*, FIPS Publication 180-1, 1995.

[23] National Institute of Standards and Technology, *Digital Signature Standard*, FIPS Publication 186-2, 2000.

[24] National Institute of Standards and Technology, *Secure Hash Standard (SHS)*, Draft FIPS 180-2, 2001. Available from http://csrc.nist.gov/encryption/tkhash.html

[25] National Institute of Standards and Technology, *Advanced Encryption Standard (AES)* FIPS Publication 197, 2001.

[26] NTT Information Sharing Platform Laboratories, "PSEC-KEM specification", September 26 2001.

[27] NTT Labs, "Self evaluation of PSEC-KEM", September 2001.

[28] P. van Oorschot and M. Wiener, "Parallel collision search with cryptanalytic applications", *Journal of Cryptology*, **12** (1999), 1-28.

[29] D. Pointcheval and T. Okamoto, "The GAP problems: A new class of problems for the security of cryptographic schemes", *Proceedings of the 2001 International Workshop on Practice and Theory in Public Key Cryptography (PKC 2001)*, Lecture Notes in Computer Science, **1992** (2001), 104-118.

[30] J. Pollard, "Monte Carlo methods for index computation mod *p*", *Mathematics of Computation*, **32** (1978), 918-924.

[31] C. Rackoff and D. Simon, "Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack", *Advances in Cryptology–Crypto '91*, Lecture Notes in Computer Science, **576** (1992), 433-444.

[32] V. Shoup, "Lower bounds for discrete logarithms and related problems", *Advances in Cryptology–Eurocrypt '97*, Lecture Notes in Computer Science, **1233** (1997), Springer-Verlag, 256-266.

[33] V. Shoup, "A proposal for an ISO standard for public key encryption (version 2.0)", September 17 2001. Available at http://shoup.net/papers/

[34] Standards for Efficient Cryptography Group, *SEC 1: Elliptic Curve Cryptography*, version 1.0, 2000. Available at http://www.secg.org

[35] E. Verheul, "Evidence that XTR is more secure than supersingular elliptic curve cryptosystems", *Advances in Cryptology–Eurocrypt 2001*, Lecture Notes in Computer Science, **2045** (2001), Springer-Verlag, 195-210.