2010 Sixth International Conference on Information Assurance and Security

A Comparison of the Standardized Versions of ECIES

V. Gayoso Martínez, F. Hernández Álvarez, L. Hernández Encinas
Information Processing and Coding
Applied Physics Institute, CSIC
Madrid, Spain
{*victor.gayoso,fernando.hernandez,luis*}*@iec.csic.es*

C. Sánchez Ávila
Applied Mathematics to Information Technologies
Polytechnic University
Madrid, Spain
carmen.sanchez.avila@upm.es

***Abstract*—Elliptic Curve Cryptography (ECC) can be used as a tool for encrypting data, creating digital signatures or performing key exchanges. In relation to encryption, the Elliptic Curve Integrated Encryption Scheme (ECIES) is the best known scheme based on ECC, and as such it has been included in several cryptographic standards. In the present work, we provide an extensive review and comparison of the versions of ECIES included in documents from ANSI, IEEE, ISO/IEC, and SECG, highlighting the main differences between them that can prevent implementations of ECIES from being fully interoperable.**

***Keywords*-Java Card; cryptography; encryption protocol;**

I. INTRODUCTION

Since the development of public key cryptography by Diffie and Hellman in 1976 [1], several cryptosystems have been proposed. Security and efficiency are the most important features to be requested to any cryptosystem and, in general, both characteristics depend on the mathematical problem on which it is based. The list of problems that are currently considered computationally infeasible to solve includes the Integer Factorization Problem (IFP), the Discrete Logarithm Problem (DLP), and the Elliptic Curve Discrete Logarithm Problem (ECDLP).

In 1985, Miller [2] and Koblitz [3] independently proposed a cryptosystem based on elliptic curves defined over finite fields, whose security relies on the ECDLP [4]. In comparison with other cryptosystems (e.g. RSA), Elliptic Curve Cryptography (ECC) uses significantly shorter keys. The reason for this fact is related to the hardness of the ECDLP, which is believed by some authors to be more difficult to solve than the IFP or the DLP ([3] and [5]).

As it is well-known, an elliptic curve, E, over a finite field, $\mathbb{F}$, can be defined by the Weierstrass equation [6], whose expression in non-homogeneous coordinates is as follows:

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, \qquad (1)$$

where $a_1, a_2, a_3, a_4, a_6 \in \mathbb{F}$ and $\Delta \neq 0$, being Δ the discriminant of the curve E [4]. This condition assures that the curve does not have curve points with two or more different tangent lines.

In practice, the equation (1) is not used, and the following simplified equations (in non-homogeneous form) are used depending on the characteristic of the finite field $\mathbb{F}$ where the elliptic curve is defined:

- If the finite field is a prime field, i.e. $\mathbb{F} = \mathbb{F}_q$, where $q > 3$ is a prime number, the equation defining the (non-supersingular) elliptic curve becomes:

$$y^2 = x^3 + ax + b. \qquad (2)$$

- If the finite field is a binary field, i.e. $\mathbb{F} = \mathbb{F}_{2^m}$, where m is an integer number, then the equation of the (non-supersingular) elliptic curve is:

$$y^2 + xy = x^3 + ax^2 + b. \qquad (3)$$

The most extended encryption scheme in ECC is the Elliptic Curve Integrated Encryption Scheme (ECIES). In this contribution, we present an extensive review and comparison of the versions of ECIES included in documents from ANSI, IEEE, ISO/IEC, and SECG, highlighting the main differences between them that can prevent implementations of ECIES from being fully interoperable.

This paper is organized as follows: Section II presents the functional design of ECIES. Section III provides a comparison among the different versions of ECIES included in the ANSI X9.63, IEEE 1363a, ISO/IEC 18033-2, and SEC 1 documents. In Section IV we offer a detailed list of the allowed functions in each standard. Finally, we will summarize the most important findings and conclusions in Section V.

II. ELLIPTIC CURVE INTEGRATED ENCRYPTION SCHEME (ECIES)

A. ECIES versions

In 1997, Mihir Bellare and Philip Rogaway [7] presented the Discrete Logarithm Augmented Encryption Scheme (DLAES), which was subsequently improved by the same authors and Michel Abdalla, being first renamed as the Diffie-Hellman Augmented Encryption Scheme (DHAES) in 1998 [8] and later as the Diffie-Hellman Integrated Encryption Scheme (DHIES) in 2001 [9], in order to avoid confusions with the Advanced Encryption Standard (AES). DHIES represents an enhanced version of ElGamal encryption scheme, using elliptic curves in an integrated

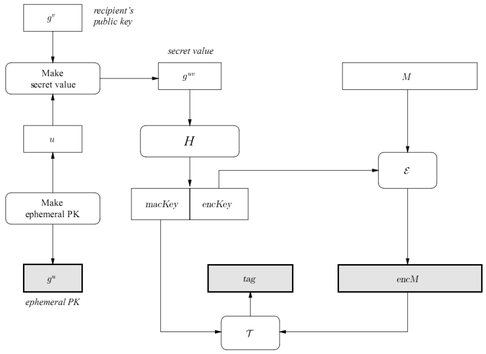

Figure 1. DHIES functional diagram.

scheme which includes public key operations, encryption algorithms, MAC codes and hash computations. Because of the integration of different functions, DHIES is considered to be secure against chosen ciphertext attacks without having to increase the number of operations or the key length [9].

Figure 1 presents the version of DHIES included in [9], where M represents the clear message, g is a generator of a multiplicative cyclic group G, g^u and g^v are the sender's and recipient's public keys, respectively, and u and v are the sender's and recipient's private keys, respectively. We denote by $\mathcal{E}$ the symmetric encryption algorithm, while $\mathcal{T}$ is the MAC code generation function, and H is the hash function.

DHIES was evaluated by ANSI and included with some modifications in the ANSI X9.63 standard [10] in 2001. Independently, IEEE had released in 2000 the IEEE 1363 standard [11], so when ANSI X9.63 was made public, they reviewed both DHIES and the version proposed by ANSI, and included a modified version in the amendment IEEE 1363a [12] released in 2004. All these versions of the scheme receive the generic name of ECIES (Elliptic Curve Integrated Encryption Scheme), even though the versions are not identical.

During those years, another ISO/IEC expert group was collaborating in the creation of the set of standards to be known as the 18033 family, so they took as starting point all the previous versions of ECIES and the existing studies about its security and produced another slightly different version of ECIES, which was included in the ISO/IEC 18033-2 standard [13] in 2006.

In order to finish this compilation of ECIES versions, it is necessary to point out a document provided by the Standards for Efficient Cryptography Group (SECG), an industry consortium, that included ECIES in the SEC 1 document [14] released in 2000 (version 1.0) and updated in 2009 (version 2.0).

B. *ECIES functional components*

As it name indicates, ECIES is an integrated encryption scheme that uses the following functions:

- Key Agreement (KA): Function used for the generation of a shared secret by two parties.
- Key Derivation (KDF): Mechanism that produces a set of keys from keying material and some optional parameters.
- Hash (HASH): Digest function.
- Encryption (ENC): Symmetric encryption algorithm.
- Message Authentication Code (MAC): Information used to authenticate a message.

After comparing the ECIES versions defined by ANSI, IEEE, ISO/IEC, and SECG, two groups of differences are clearly detected:

- Functionality: Implementation details, usage of optional elements, binary representation conventions, etc.
- Allowed functions: Specific functions (HASH, ENC, etc.) allowed for their use in each standard.

III. COMPARISON OF ECIES FUNCTIONALITY

In this section, we analyse in pairs the main differences in terms of functionality among the different standard implementations of ECIES.

A. *DHIES and ANSI X9.63 versions*

The following list presents the main differences between the original DHIES specification [15] and the version implemented in the ANSI X9.63 standard [10].

- DHIES does not allow arbitrary parameters in neither the KDF nor the MAC function, whilst X9.63 does allow parameters in both functions.
- DHIES uses a HASH function to produce the MAC and ENC keys. In comparison, ANSI X9.63 uses a KDF construction where the data is processed during several rounds.
- DHIES interprets the leftmost bits of the output of the KDF function as the MAC key, and the rightmost bits as the ENC key. In ANSI X9.63, the order is precisely the opposite.
- DHIES allows to use as the symmetric encryption cipher either a stream or a block cipher, letting the option of the specific cipher to use open. In comparison, ANSI X9.63 only allows the usage of the XOR function.

B. *ANSI X9.63 and IEEE 1363a versions*

The list included hereafter reflects the main differences between the ECIES implementations included in the ANSI X9.63 [10] and IEEE 1363a [12] standards.

- ANSI X9.63 only uses the DH with cofactor procedure as the KA function, while IEEE 1363a allows both versions with and without cofactor.
- ANSI X9.63 allows to use an arbitrary parameter as an input to the KDF function, but does not mention the content of that optional parameter. In comparison, the so-called DHAES mode in IEEE 1363a mandates to

use the binary representation of the sender's public key as an input parameter.

- ANSI X9.63 produces the first coordinate of the element generated as the output of the KA function, while IEEE 1363a uses the whole element generated.
- ANSI X9.63 always interprets the leftmost bits of the output of the KDF function as the ENC key, and the rightmost bits as the MAC key. In comparison, IEEE 1363a interprets the output as $k_{MAC}||k_{ENC}$ when using a stream cipher, and the opposite when using a block cipher.

C. IEEE 1363a and ISO/IEC 18033-2 versions

The following list presents the main differences between the IEEE 1363a [12] and ISO/IEC 18033-2 [13] standards.

- IEEE 1363a allows the usage of parameters in the KDF function, whereas ISO/IEC 18033-2 does not allow parameters in that function.
- IEEE 1363a includes the option to use either bit or byte strings, whilst ISO/IEC 18033-2 mandates the usage of byte strings.
- IEEE 1363a suggests to use always the same set of parameters and functions for a given public key. In comparison, ISO/IEC 18033-2 mandates not to change under any circumstance those parameters for the same receiver's public key.
- IEEE 1363a states that the minimum key length must be 160 bits. In contrast, ISO/IEC 18033-2 does not mention any minimum key length.

It is worth mentioning that there were more differences between IEEE 1363a and the first draft versions of ISO/IEC 18033-2 [16]. These differences were removed in the final version of the document in order to obtain a higher level of compatibility with previous standards and implementations.

D. ISO/IEC 18033-2 and SECG SEC 1 versions

The following list provides the main differences between the ECIES implementations included in the ISO/IEC 18033-2 standard [13] and the SEC 1 document [14].

- ISO/IEC 18033-2 does not allow input parameters in the KDF function, whilst SEC 1 allows to include this additional information, even though in the example test vectors included in the GEC 2 document [17] no additional parameters were used.
- SEC 1 does not explicitly include the sender's ephemeral public key in the KDF computation. However, it mentions that the public key could be one of the elements that could be used as input parameter in that function.
- ISO/IEC 18033-2 does not mention minimum key lengths, whereas SEC 1 states that the selection of the field must be guided by the following requirements:

  $\mathbb{F}_q$: $\lceil \log_2 q \rceil \in \{192, 224, 256, 384, 521\}$.
  $\mathbb{F}_{2^m}$: $m \in \{163, 233, 239, 283, 409, 571\}$.

IV. ECIES ALLOWED FUNCTIONS COMPARISON

This section presents the comparison of the allowed KA, KDF, HASH, ENC, and MAC functions included in the aforementioned standards, as summarized in Table I, where:

- DH denotes the Diffie Hellman key agreement function and DHC is the Diffie Hellman function with cofactor.
- X9.63-KDF is defined in [10], KDF1 and KDF2 are functions included in [13], and NIST-800-56 is the KDF concatenation function of [18].
- SHA-1 is included in [19], SHA-2 represents the family composed by SHA-256, SHA-384, and SHA-512, SHA-2* is the SHA-2 family with the addition of the SHA-224 hash algorithm, RIPEMD is the set of hash algorithms defined in [20], and WHIRLPOOL is included in [21].
- TDES is the Triple DES algorithm in CBC mode [22], AES represents the Advanced Encryption Standard family (i.e., AES-128, AES-192, and AES-256), MISTY1 is the algorithm specified in [23] and CAST-128 is included in [24].
- DEA is the MAC function specified in [25], X9.71 is included in [26], MAC1, HMAC-SHA-1, and HMAC-RIPEMD are defined in [27], HMAC-SHA-2 represents the family of HMAC algorithms (i.e., HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512) described in [28], HMAC-SHA-2* is the same as HMAC-SHA-2 with the addition of the HMAC-SHA-224 function, and CMAC-AES is the set of AES-related HMAC functions included in [29].

Table I
KA FUNCTION

	X9.63	1363a	18033-2	SEC 1
KA	DH	DH DHC	DH DHC	DH DHC
KDF	X9.63-KDF	X9.63-KDF	KDF1 KDF2	X9.63-KDF NIST-800-56
HASH	SHA-1	SHA-1 SHA-2 RIPEMD	SHA-1 SHA-2 RIPEMD WHIRLPOOL	SHA-1 SHA-2*
ENC	XOR	TDES AES	TDES AES MISTY1 CAST-128	XOR AES
MAC	DEA ANSI X9.71	MAC1	H-SHA-1 H-SHA-2 H-RIPEMD	H-SHA-1 H-SHA-2* CMAC-AES

V. CONCLUSION

After reviewing the different versions of ECIES included in the standards, it is clear that it is not possible to implement a software version compatible with all the standards, regarding both the specific operations and the list of allowed functions and algorithms. In addition to this, implementations may face another important problem, which is the limitation in the functions available to the developer in the programming interface of the target device.

The direct consequence is that, when implementing ECIES, the first step should be to evaluate the capabilities provided by the final platform and, from that point, decide which version of ECIES to implement. Regarding this point, even though the newer versions (e.g. ISO/IEC 18033-2 and SEC 1) may not be fully compatible with legacy devices, they provide access to the most recent and secure functions (e.g. SHA-2, AES, etc.), so it should be recommended to use one of these versions.

ACKNOWLEDGMENT

This work has been partially supported by Ministerio de Ciencia e Innovación (Spain), under the grant TEC2009-13964-C04-02, and Ministerio de Industria, Turismo y Comercio (Spain), in collaboration with CDTI and Telefónica I+D, under the project Segur@ CENIT-2007 2004.

REFERENCES

[1] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inform. Theory*, vol. 22, pp. 644–654, 1976.

[2] V. S. Miller, "Use of elliptic curves in cryptography," *Lecture Notes in Comput. Sci.*, vol. 218, pp. 417–426, 1986.

[3] N. Koblitz, "Elliptic curve cryptosystems," *Math. Comp.*, vol. 48, pp. 203–209, 1987.

[4] A. J. Menezes, *Elliptic Curve Public Key Cryptosystems*. Boston, MA, USA: Kluwer Academic Publishers, 1993.

[5] BSI TR 03111, *Elliptic Curve Cryptography*, Bundesamt für Sicherheit in der Informationstechnik, 2009, http://www.bsi.de/literat/tr/tr03111/BSI-TR-03111.pdf.

[6] J. H. Silverman, *The Arithmetic of Elliptic Curves*, ser. Graduate texts in Mathematics. New York, NY, USA: Springer-Verlag, 1986, vol. 106.

[7] M. Bellare and P. Rogaway, "Minimizing the use of random oracles in authenticated encryption schemes," *Lecture Notes in Comput. Sci.*, vol. 1334, pp. 1–16, 1997.

[8] M. Abdalla, M. Bellare, and P. Rogaway, *DHAES: An Encryption Scheme Based on the Diffie-Hellman Problem*, contribution to IEEE P1363a, 1998, http://grouper.ieee.org/groups/1363/P1363a/contributions/dhaes.pdf.

[9] ——, *DHIES: An encryption scheme based on the Diffie-Hellman Problem*, unpublished, 2001, http://www.cs.ucdavis.edu/~rogaway/papers/dhies.pdf.

[10] ANSI X9.63, *Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography*, American National Standards Institute, 2001.

[11] IEEE 1363, *Standard Specifications for Public Key Cryptography*, Institute of Electrical and Electronics Engineers, 2000.

[12] IEEE 1363a, *Standard Specifications for Public Key Cryptography - Amendment 1: Additional Techniques*, Institute of Electrical and Electronics Engineers, 2004.

[13] ISO/IEC 18033-2, *Information Technology – Security Techniques – Encryption Algorithms – Part 2: Asymmetric Ciphers*, International Organization for Standardization / International Electrotechnical Commission, 2006.

[14] SECG SEC1, *Elliptic Curve Cryptography*, Standards for Efficient Cryptography Group, ver. 2, 2009, http://www.secg.org/download/aid-780/sec1-v2.pdf.

[15] M. Abdalla, M. Bellare, and P. Rogaway, "The oracle Diffie-Hellman assumptions and an analysis of DHIES," *Lecture Notes in Comput. Sci.*, vol. 2020, pp. 143–158, 2001.

[16] V. Shoup, *A Proposal for an ISO Standard for Public Key Encryption*, Cryptology ePrint Archive, Report 2001/112, 2001, http://eprint.iacr.org/2001/112.pdf.

[17] SECG GEC 2, *Test Vectors for SEC 1*, Standards for Efficient Cryptography Group, 1999, http://www.secg.org/download/aid-390/gec2.pdf.

[18] NIST SP 800-56A, *Recommendation for Pair-wise Key Establishment Schemes Using Discrete Logarithm Cryptography*, National Institute of Standards and Technology, 2005.

[19] NIST FIPS 180-2, *Secure Hash Standard*, National Institute of Standards and Technology, 2002.

[20] H. Dobbertin, A. Bosselaers, and B. Preneel, "RIPEMD-160: A strengthened version of RIPEMD," *Lecture Notes in Comput. Sci.*, vol. 1039, pp. 71–82, 1996.

[21] ISO/IEC 10118-3, *Information Technology – Security Techniques – Hash-functions – Part 3: Dedicated Hash-functions*, International Organization for Standardization / International Electrotechnical Commission, 2004.

[22] ANSI X9.52, *Triple Data Encryption: Modes of Operation*, American National Standards Institute, 1998.

[23] M. Matsui, *Specification of MISTY1 - A 64-bit Block Cipher*, proposal sent to NESSIE, 2000.

[24] C. Adams, *RFC 2144 - The CAST-128 Encryption Algorithm*, Internet Engineering Task Force, 1997, http://www.ietf.org/rfc/rfc2144.txt.

[25] ANSI X9.19, *Financial Institution Retail Message Authentication*, American National Standards Institute, 1996.

[26] ANSI X9.71, *Keyed Hash Message Authentication Code*, American National Standards Institute, 2001.

[27] H. Krawczyk, M. Bellare, and R. Canetti, *RFC 2104 - HMAC: Keyed Hashing for Message Authentication*, Internet Engineering Task Force, 1997, http://www.ietf.org/rfc/rfc2104.txt.

[28] NIST FIPS 198, *The Keyed-Hash Message Authentication Code (HMAC)*, National Institute of Standards and Technology, 2002.

[29] *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*, National Institute of Standards and Technology, 2005.