

Proposal to IEEE P1363a (revised, February 22, 2000)

Tatsuaki Okamoto (NTT Labs, Japan)

This draft proposes a public-key encryption scheme, EPOC, and digital signature scheme, ESIGN, to IEEE P1363a.

In this document, the description of EPOC is divided into three parts: one is new encryption primitives, IFEP-OU and IFDP-OU, known as OU (for "Okamoto-Uchiyama" —see [OU98]), and two new encryption schemes, IFES2 and IFIES, based on the OU primitives, known as EPOC-1 and EPOC-2 respectively (see [OUF99]).

ESIGN (see [Oka90], [FO99]) is described as a new signature primitive, IFSP-ESIGN and IFVP-ESIGN. The related description is shown in IFSSA .

Contents of this document are as follows:

8. Primitives Based on the Integer Factorization Problem (Updated)

8.1.3.4 OU Key Pairs (new)

8.1.3.5 ESIGN Key Pairs (new)

8.2.10 IFEP-OU (new)

8.2.11 IFDP-OU (new)

8.2.12 IFSP-ESIGN (new)

8.2.13 IFVP-ESIGN (new)

10. Signature Schemes (updated)

10.3 IFSSA (updated)

11. Encryption Schemes (updated)

11.4 IFES2 (new)

11.5 IFIES (new)

A.16.13 An Algorithm for Generating OU Keys (new)

A.16.14 An Algorithm for Generating ESIGN Keys (new)

D.4.3.4 Notes (updated)

Bibliography (updated)

8. Primitives Based on the Integer Factorization Problem (updated)

Two new types of primitives are added in this family. One type is known as OU (for "Okamoto-Uchiyama" —see [OU98]; the other is known as ESIGN (see [Oka90], [FO99]).

8.1.3.4 OU Key Pairs (new)

An OU *public key* consists of a *modulus* n , which is the product, p^2q , of two odd positive prime integers p and q , an integer $k = \lfloor \log_2 p \rfloor$, and a pair of integers (u, v) ($1 < u < n$, $1 < v < n$) such that the order of $u_p = \exp^{(u, p-1)} \bmod p^2$ is p , and $v = v_0^n \bmod n$ for an integer v_0 ($1 < v_0 < n$). The corresponding OU *private key* K is (p, u_p) .

To skip the procedure for computing $L(u_p)$ in decryption, the corresponding OU private key K can be $(p, L(u_p))$, where

$$L(u_p) = (u_p - 1) / p$$

8.1.3.5 ESIGN Key Pairs (new)

An ESIGN *public key* consists of a *modulus* n , which is the product, p^2q , of two odd positive prime integers p and q , an integer $k = \lfloor \log_2 p \rfloor$, and a *public exponent* e ($8 \leq e < n$) which is an integer. The corresponding ESIGN *private key* is a pair of primes, (p, q) .

8.2.10 IFEP-OU (new)

IFEP-OU is OU Encryption Primitive. It is based on the work of [OU98]. It is invoked in the scheme IFES as part of encrypting a message, given the message and the public key of the intended recipient. The message can be decrypted within a scheme by invoking IFDP-OU.

Input:

- the recipient's OU public key (n, u, v, k)
- the message representative, which is an integer f such that $0 \leq f < 2^k$, and an integer r such that $0 \leq r < n$

Assumptions: public key (n, u, v, k) is valid; $0 \leq f < 2^k$

Output: the encrypted message representative, which is an integer g such that $0 \leq g < n$

Operation. The encrypted message representative g shall be computed by the following or an equivalent sequence of steps:

1. Let $g = \exp(u, f) \times \exp(v, r) \bmod n$
2. Output g .

Conformance region recommendation. A conformance region should include:

- at least one valid OU public key (n, u, v, k)
- all message representatives f in the range $[0, 2^k - 1]$

8.2.11 IFDP-OU (new)

IFDP-OU is OU Decryption Primitive. It is based on the work of [OU98]. It is used in the scheme IFES as part of decrypting a message encrypted with the use of IFEP-OU, given the encrypted message representative and the private key of the recipient.

Input:

- the recipient's OU private key K
- the encrypted message representative, which is an integer g such that $0 \leq g < n$

Assumptions: private key K is valid; $0 \leq g < n$

Output: the message representative, which is an integer f such that $0 \leq f < 2^k$: or "invalid"

Operation. The message representative f shall be computed by the following or an equivalent sequence of steps:

1. Compute $g_p = \exp(g, p - 1) \bmod p^2$.
2. Compute $L(g_p) = (g_p - 1) / p$ and $L(u_p) = (u_p - 1) / p$. (If $L(u_p)$ is a part of the private key K , compute only $L(g_p)$.)
3. Compute $f = L(g_p) / L(u_p) \bmod p$.
4. Check whether $0 \leq f < 2^k$
5. If it holds, output f . Otherwise, output "invalid".

Conformance region recommendation. A conformance region should include:

- at least one valid OU private key K
- all encrypted message representatives g in the range $[0, n - 1]$, where n is from the public key

8.2.12 IFSP-ESIGN (new)

IFSP-ESIGN is ESIGN Signature Primitive. It is based on the work of [Oka90], [FO99]. It is used in a signature scheme with appendix and is invoked in the scheme IFSSA as part of signature generation.

Input:

- the signer's ESIGN private key K
- the message representative, which is an integer f such that $0 \leq f < 2^k$

Assumptions: private key K is valid; $0 \leq f < 2^k$

Output: the signature, which is an integer s such that $0 \leq s < n$

Operation. The signature s shall be computed by the following or an equivalent sequence of steps:

1. Generate a random integer r ($0 \leq r < pq$).
2. Set $z = f \times 2^{2(k+1)}$
3. Compute $a = z - \exp(r, e) \bmod n$.
4. Compute $w_0 = \lceil a / (pq) \rceil$
5. Compute $w_1 = w_0 \times pq - a$
6. If $w_1 \geq 2^{2k+1}$, then go to step 1
7. Compute $t = w_0 / (e \times \exp(r, e - 1)) \bmod p$
8. Compute $s = r + tpq \bmod n$.

Conformance region recommendation. A conformance region should include:

- at least one valid ESIGN private key K
- all message representatives f in the range $[0, n - 1]$, where n is from the private key K

8.2.13 IFVP-ESIGN (new)

IFVP-ESIGN is ESIGN Verification Primitive. It is based on the work of [Oka90], [FO99]. It is used in a signature scheme with appendix and can be invoked in the scheme IFSSA as part of signature verification.

Input:

- the signer’s ESIGN public key (n, e, k)
- the message representative, which is an integer f such that $0 \leq f < n$, and the signature to be verified, which is an integer s

Assumptions: public key (n, e, k) is valid

Output: “valid” or “invalid”

Operation. The message representative f and signature representative s shall be verified by the following or an equivalent sequence of steps:

- 1 If s is not in $[0, n - 1]$, output “invalid” and stop.
- 2 Check whether the following equation holds or not.

$$f = \lfloor (\exp(s, e) \bmod n) / 2^{2(k+1)} \rfloor$$
- 3 If it holds, output “valid”. Otherwise, output “invalid”.

Conformance region recommendation. A conformance region should include:

- at least one valid ESIGN public key (n, e, k)
- all purported signatures s that can be input to the implementation; this should at least include all s in the range $[0, n - 1]$

10. Signature Schemes (updated)

10.3 IFSSA (updated)

Add the following primitives to the list of recommended primitives for signatures with appendix: IFSP-ESIGN (Section 8.2.12) and IFVP-ESIGN (Section 8.2.13).

In addition, add the following to Section 10.3.1:

- the message encoding method for signatures with appendix, which should be EMSA3 (Section 12.1.3) and l with EMSA3 should be set as k with IFSP-ESIGN, or a technique designated for use with IFSSA in an addendum to this standard, if the signature primitive is IFSP-ESIGN.

11. Encryption Schemes (updated)

11.4 IFES2 (new)

IFES2 is Integer Factorization Encryption Scheme based on the encryption primitives, IFEP-OU (Section 8.2.10) and IFDP-OU (Section 8.2.11), and the conversion, [FO99a].

11.4.1 Scheme Options

The following options shall be established or otherwise agreed upon between the parties to the scheme (the sender and the recipient):

- the encryption primitive, which should be IFEP-OU (Section 8.2.10) and IFDP-OU (Section 8.2.11), or a technique designated for use with IFIES in an addendum to this standard
- a hash function *Hash* with output length *hLen* octets, which shall be SHA-1 (Section 14.1.1), or RIPEMD-160 (Section 14.1.2), or a technique designated for use with EME2 in an addendum to the standard
- a mask generation function *G*, which shall be MGF1, or a technique designated for use with EME2 in an addendum to this standard

The above information may remain the same for any number of executions of the encryption scheme, or it may be changed at some frequency. The information need not be kept secret.

11.4.2 Encryption Operation

Input:

- a message, which is an octet string *M* of length *mLen* octets
- an integer *k* (same as *k* with IFEP-OU) and the octet length of *seed*, *seedLen*, such that $mLen + seedLen \leq 8 \times k$

Output: the encrypted message representative, *EG*, or “error”

A ciphertext *EG* shall be generated by a sender from a message *M* of bit length $l (= 8 \times mLen)$ by the following or an equivalent sequence of steps:

1. Generate a fresh, random octet string *seed* of length *seedLen* octets (for more on generation of random strings, see Annex D.6).
2. Let $EM = M \parallel seed$
3. Apply the hash function *Hash* to *EM* to produce an octet string *cHash* of length *rLen*.
4. Convert *EM* and *cHash* to integers *f* and *r* with the primitive OS2IP, respectively.
5. Obtain the recipient’s purported public key (n, u, v, k) for the operation of IFEP-OU (Section 8.2.10).
6. Compute the encrypted message representative *g* from (f, r) with the encryption primitive IFEP-OU (Section 8.2.10).
7. Convert *g* to an octet string *EG* using FE2OSP.
8. Output *EG* as the encrypted message representative.

Conformance region recommendation. A conformance region should include:

- at least one valid set of domain parameters
- all valid public keys (n, u, v, k) associated with the set of domain parameters; if key validation is performed, invalid public keys that are appropriately handled by the implementation may also be included in the conformance region
- a range of messages *M*

11.4.3 Decryption Operation

Input:

- ciphertext *EG*

Output: the message, which is an octet string M ; or “error”

The plaintext M shall be recovered from the ciphertext EG , by the following or an equivalent sequence of steps:

1. Convert EG to an integer g with the primitive I2OSP.
2. Select the private key (p, u_p) for the operation of IFDP-OU (Section 8.2.11).
3. Compute the message representative f from g with the decryption primitive IFDP-OU (Section 8.2.11)
4. Convert f to an octet string EM using FE2OSP.
5. Let M be the leftmost $mLen$ octets of EM .
6. Apply the hash function $Hash$ to EM to produce an octet string $cHash$ of length $rLen$.
7. Convert $cHash$ to an integers r with the primitive OS2IP.
8. Check whether the following equation holds or not:

$$g = \exp(u, f) \times \exp(v, r) \bmod n.$$
9. If it holds, output the message M . Otherwise output “error”.

11.5 IFIES (new)

IFIES is Integer Factorization Integrated Encryption Scheme based on the encryption primitives, IFEP-OU (Section 8.2.10) and IFDP-OU (Section 8.2.11), and the conversion, [FO99b].

11.5.1 Scheme Options

The following options shall be established or otherwise agreed upon between the parties to the scheme (the sender and the recipient):

- the encryption primitive, which should be IFEP-OU (Section 8.2.10) and IFDP-OU (Section 8.2.11), or a technique designated for use with IFIES in an addendum to this standard
- a hash function $Hash$ with output length $hLen$ octets, which shall be SHA-1 (Section 14.1.1), or RIPEMD-160 (Section 14.1.2), or a technique designated for use with EME2 in an addendum to the standard
- a mask generation function G , which shall be MGF1, or a technique designated for use with EME2 in an addendum to this standard

The above information may remain the same for any number of executions of the encryption scheme, or it may be changed at some frequency. The information need not be kept secret.

11.5.2 Encryption Operation

Input:

- a message, which is an octet string M of length $mLen$ octets
- an integer k (same as k with IFEP-OU) and the octet length of $seed$, $seedLen$, such that $seedLen \leq 8 \times k$

Output: the encrypted message representative, (EG, C) or “error”

A ciphertext (EG, C) shall be generated by a sender from a message M of octet length $mLen$, by the following or an equivalent sequence of steps:

1. If the length of M is greater than the length limitation ($2^{61} - 1$ octets for SHA-1 or RIPEMD-160), output “error” and stop.
2. Generate a fresh, random octet string $seed$ of length $seedLen$ octets (for more on generation of random strings, see Annex D.6).
3. Let $EM = M \parallel seed$
4. Apply the hash function $Hash$ to EM to produce an octet string $cHash$ of length $rLen$.
5. Convert $seed$ and $cHash$ to integers f and r with the primitive OS2IP, respectively.
6. Obtain the recipient’s purported public key (n, u, v, k) for the operation of IFEP-OU (Section 8.2.10).
7. Compute the encrypted message representative g from (f, r) with the encryption primitive IFEP-OU (Section 8.2.10).
8. Convert g to an octet string EG using FE2OSP.
9. Apply the mask generation function G to $seed$ to produce an octet string EK of length $mLen$.
10. Mask the message M with the key EK to produce an encrypted message $C = M \oplus EK$.
11. Output the pair (EG, C) as the ciphertext.

Conformance region recommendation. A conformance region should include:

- at least one valid set of domain parameters
- all valid public keys (n, u, v, k) associated with the set of domain parameters; if key validation is performed, invalid public keys that are appropriately handled by the implementation may also be included in the conformance region
- a range of messages M

11.5.3 Decryption Operation

Input:

- ciphertext (EG, C)

Output: the message, which is an octet string M ; or “error”

The plaintext M shall be recovered from the ciphertext (EG, C) , by the following or an equivalent sequence of steps:

1. If the length of C is greater than the length limitation ($2^{61} - 1$ octets for SHA-1 or RIPEMD-160), output “error” and stop.
2. Convert EG to an integer g with the primitive I2OSP.
3. Select the private key (p, u_p) for the operation of IFDP-OU (Section 8.2.11).
4. Compute the message representative f from g with the decryption primitive IFDP-OU (Section 8.2.11).
5. Convert f to an octet string $seed$ using FE2OSP.
6. Apply the mask generation function G to $seed$ to produce an octet string EK of length $mLen$.
7. Unmask the encrypted message C with the key EK to produce a message $M = C \oplus EK$.
8. Let $EM = M \parallel seed$.
9. Apply the hash function $Hash$ to EM to produce an octet string $cHash$ of length $rLen$.
10. Convert $cHash$ to integers r with the primitive OS2IP.
11. Check whether the following equation holds or not:

$$g = \exp(u, f) \times \exp(v, r) \text{ mod } n.$$
12. If it holds, output the message M . Otherwise output “error”.

A.16.13 An Algorithm for Generating OU Keys (new)

To be added later.

A.16.14 An Algorithm for Generating ESIGN Keys (new)

To be added later.

D.4.3.4 Notes (updated)

To be added later.

Bibliography (updated)

To be added later.