IEEE P1363.3

Standard Specifications for Public Key Cryptography: Identity Based Key Agreement Scheme (IBKAS)

Abstract. This document specifies pairing based, identity based, and authenticated key agreement techniques. One of the advantages of Identity Based key agreement techniques is that there is no public key transmission and verification needed.

Contents

1. Definitions

For the purposes of this standard, the following terms and definitions apply.

key agreement: a method by which two entities, using each other's public keys (identities) and their own private keys, agree on a common secret key that only they know.

key confirmation: the assurance provided to each party participating in a key agreement protocol that the other party is capable of computing the agreed-upon key, and that it is the same for both.

key derivation: the process of deriving a secret key from a secret value.

private key: the private element of the public/private key pair.

identity based public key: the public element of the public/private key pair, which is usually derived from an entity's identity.

shared secret key: a secret key shared by two parties, usually derived as a result of a key agreement scheme. See also: key agreement; secret key.

shared secret value: a secret value shared by two parties, usually during a key agreement scheme. See also: key agreement; secret value.

2. Types of Cryptographic Techniques

This section gives an overview of the types of cryptographic techniques that are specified in this standard as well as some requirements for conformance with those techniques.

2.1 General Model

Different types of cryptographic techniques can be viewed abstractly according to the following three-level general model.

— *Primitives* – basic mathematical operations. Primitives are not meant to achieve security just by themselves, but they serve as building blocks for schemes.

— *Schemes* – a collection of related operations combining primitives and additional methods. Schemes can provide complexity-theoretic security which is enhanced when they are appropriately applied in protocols.

— *Protocols* – sequences of operations to be performed by multiple parties to achieve some security goal. Protocols can achieve desired security for applications if implemented correctly.

2.2 Primitives

The following types of primitives are defined in this document:

— Secret Value Derivation Primitives (SVDP), components of key agreement schemes.

2.3 Schemes

The following types of schemes are defined in this document:

— Identity Based Key Agreement Schemes (IBKAS), in which two parties use their identity based public and private key pairs and possibly other information to agree on a shared secret key. The shared secret key may then be used for symmetric cryptography.

Schemes in this standard are presented in a general form based on certain primitives and additional methods. The specification of a scheme consists of the following information:

— *scheme options*, such as choices for primitives and additional methods
— one or more *operations*, depending on the scheme, expressed as a series of *steps*
— *conformance region recommendations* for implementations conforming with the scheme

As for primitives, the specifications are functional specifications, not interface specifications. As such, the format of inputs and outputs and the procedure by which an implementation of a scheme is invoked are outside the scope of this standard. See Annex E for more information on input and output formats.

2.4 Additional Methods

This standard uses the following methods from IEEE 1363-2000:

— Key Derivation Functions (KDF), which are a component of key agreement schemes.

— Auxiliary Functions, which are building blocks for other additional methods.

  — Hash Functions

The additional methods are strongly recommended for use in the schemes. The use of an inadequate key derivation function or auxiliary function may compromise the security of the scheme in which it is used. Therefore, any implementation which chooses not to follow the recommended additional methods for a particular scheme should perform its own thorough security analysis of the resulting scheme.

2.5 Table Summary

This section gives a summary of all the schemes in this standard, together with the primitives and additional methods that are invoked within a scheme.

Scheme or Protocol	Primitives	Additional Methods
Identity based key agreement schemes		
IBKAS-IDAK	IBSVDP-IDAK	
IBKAS-SCK	IBSVDP-SCK	

3. Primitives for Identity Based Key Agreement Problem

3.1 Primitives Borrowed from Other Sections

In this section, we assume that the following primitives have been defined (in other sections, e.g., in the IBE primitive section): IBPUBKDP and IBPRIKDP. It should be noted that the two primitives described in this section may not be exact. The final version should use the primitive definitions from other sections.

3.1.1 IBPUBKDP

IBPUBKDP is Identity Based Public Key Derivation Primitive which derives an entity's public key from its public identity string.

Input:
— the EC domain parameters q, a, b, r and G
— a hash function H
— the party's identity octet string id
— the master secret α

Assumptions: EC domain parameters q, a, b, r and G are valid

Output: the derived public key W, which is a generator of the EC group; or "error" (note that this should generally be defined by computing $H(id)$ first, and then map this element to an EC generator

3.1.2 IBPRIKDP

IBPRIKDP is Identity Based Private Key Derivation Primitive.

Input:
— the EC domain parameters q, a, b, r and G
— a master secret octet string α
— the party's public key W, which is derived using the IBPUBKDP primitive

Assumptions: EC domain parameters q, a, b, r and G are valid, the public key W is valid.

Output: the derived private key U, which is a point on the EC group; or "error"

Note: $U = \alpha W$

3.2 Primitives

As detailed in Section ???, an implementation of a primitive may make certain assumptions about its inputs, as listed with the specification of each primitive. For example, if EC domain parameters q, a, b, r and G and a public key W are inputs to a primitive, the implementation may generally assume that the domain parameters are valid, i.e., that G has order r on the elliptic curve defined by a and b, $W = H(id)$ where id is the identity string and H is an appropriate hash function. The behavior of an implementation is unconstrained in the case that W is not an appropriate public key, and in such a case the implementation may or may not output an error condition. It is up to the properly implemented scheme to ensure that only appropriate inputs are passed to a primitive, or to accept the risks of passing inappropriate inputs.

3.2.1 IBSVDP-IDAK

IBSVDP-IDAK is Identity Based Secret Value Derivation Primitive, IDAK version. It is based on the work of [Wang2005]. This primitive derives a shared secret value from one party's two key pairs and another party's two public keys. If two parties correctly execute this primitive, they will produce the same output. This primitive can be invoked by a scheme to derive a shared secret key; specifically, it may be used with the scheme IBKAS-IDAK. It assumes that the input keys are valid.

In this primitive, let $h = \lceil (\log_2 r) / 2 \rceil$. Note that h depends only on the EC domain parameters, and hence can be computed once for a given set of domain parameters.

Input:

— the EC domain parameters q, a, b, r and G associated keys U, (u, V), W', V' (the domain parameters shall be the same for these keys)
— a modified pairing function $\hat{e}(.,.)$
— the party's own identity based private key U which is an elliptic curve point
— the party's own second key pair (u, V), where $V = (x, y)$
— the other party's identity based public key W'
— the other party's second public key $V' = (x', y')$
— a hash function H that maps an octet string to an octet string of length h.

Assumptions: private key W, key pair (u, V), public keys W', V', and EC domain parameters q, a, b, r and G are valid; all the keys are associated with the domain parameters

Output: the derived shared secret value, which is a nonzero field element $z \in GF(q)$; or "error"

Operation. The shared secret value z shall be computed by the following or an equivalent sequence of steps:

1. Convert x into an octet string o using FE2OSP.
2. Convert x' into an octet string o' using FE2OSP.
3. Compute an octet string $s=H(o,o')$
4. Convert s into an integer i using OS2IP
5. Compute an octet string $s'=H(o',o)$
6. Convert s' into an integer i' using OS2IP
7. Compute an elliptic curve point $P=(u+i)\ U$
8. Compute an elliptic curve point $Q=i'W'+V'$
9. Let $z = \hat{e}(P,Q)$.
10. If $z = 1$, then terminate
11. Output z as the shared secret value.

Conformance region recommendation. A conformance region should include:

— at least one valid set of EC domain parameters q, a, b, r and g
— at least one valid private key U for each set of domain parameters
— all valid key pairs (u, V) associated with the same set of domain parameters as s
— all valid public keys w' and v' associated with the same set of domain parameters as s

NOTES

1—This primitive does not address small subgroup attacks, which may occur when the public keys W' and V' are not valid and when Weil pairing is used. To prevent them, a key agreement scheme should validate the public keys W' and V' before executing this primitive.

3.2.2 IBSVDP-SCK

IBSVDP-SCK is Identity Based Secret Value Derivation Primitive, SCK version. It is based on the work of [Smart2002, ChenKudla2002]. This primitive derives a shared secret value from a domain public key, one party's two key pairs and another party's two public keys. If two parties correctly execute this primitive, they will produce the same output. This primitive can be invoked by a scheme to derive a shared secret key; specifically, it may be used with the scheme IBKAS-SCK. It assumes that the input keys are valid.

Input:

— the EC domain parameters q, a, b, r and G associated keys R, d, x, Q, E (the domain parameters shall be the same for these keys)
— a modified pairing function $\hat{e}(.,.)$
— a domain public key R which is an elliptic curve point
— the party's own identity based private key d which is an elliptic curve point
— the party's own second private key x, which is an integer
— the other party's identity based public key Q which is an elliptic curve point
— the other party's second public key E which is an elliptic curve

Assumptions: The domain public key R, private key d, private key x, public keys Q and E, and EC domain parameters q, a, b, r and G are valid; all the keys are associated with the domain parameters.

Output: the derived shared secret value, which is a nonzero field element $z \in GF(q)$; or "error".

Operation. The shared secret value z shall be computed by the following or an equivalent sequence of steps:

1. Compute an elliptic curve point $A=xQ$,
2. Compute a pairing $t_1 = \hat{e}(A,R)$,
3. Convert t_1 into an octet string o_1 using FE2OSP,
4. Compute a pairing $t_2 = \hat{e}(d,E)$,
5. Convert t_2 into an octet string o_2 using FE2OSP,
6. Compute an elliptic curve point $B = xE$,
7. Convert B into an octet string o using EP2OSP --- **this primitive should have been specified somewhere**,
8. Let $z = o \,||\, o_1 \,||\, o_2$, ------ **|| denotes for concatenation, how to address this operation should have been specified somewherer?**
9. If $z = I$, then terminate,
10. Output z as the shared secret value.

Conformance region recommendation. A conformance region should include:

— at least one valid set of EC domain parameters q, a, b, r and g
— at least one valid private key d for each set of domain parameters
— valid private key x associated with the same set of domain parameters as s
— all valid public keys Q and E associated with the same set of domain parameters as s

NOTES

1—This primitive does not address small subgroup attacks, which may occur when the public keys Q and E are not valid. To prevent them, a key agreement scheme should validate the public keys Q and E before executing this primitive.

4. Identity Based Key Agreement Schemes

4.1 General Model

In a key agreement scheme, each party combines its own private key(s) with the other party's public key(s) to come up with a secret key. Other (public or private) information known to both parties may also enter the scheme as *key derivation parameters*. If the parties use the corresponding keys and identical key derivation parameters, and the scheme is executed correctly, the parties will arrive at the same secret key (see note 1, below). A key agreement scheme can allow two parties to derive shared secret keys without any prior shared secret.

A key agreement scheme consists of a key agreement operation, along with supporting key management. Domain parameter and key pair generation for the key agreement schemes are specified further in Section?). A key agreement operation has the following form for all the schemes:

1. Establish one or more sets of valid domain parameters with which the parties' key pairs shall be associated.
2. Obtain one's valid private keys for the operation, associated with the domain parameters established in Step 1.
3. Establish one or more valid private keys for the operation, associated with the domain parameters established in Step 1
4. Obtain other party's identity based public key and one or more purported public keys for the operation.
5. *(Optional.)* Depending on the cryptographic operations in Step 5, choose an appropriate method to validate the public keys and the domain parameters. If any validation fails, output "invalid" and stop.
6. Apply certain cryptographic operations to the private and public keys to produce a shared secret value.
7. For each shared secret key to be agreed on, establish or agree on key derivation parameters and derive a shared secret key from the shared secret value and the key derivation parameters using a key derivation function.

NOTES

1—*(Key confirmation.)* By the definition of a key agreement scheme, if the correct keys are used and computation is performed properly, the shared secret keys computed by the two parties will also be the same. However, to verify the identities of the parties and to ensure that they indeed possess the same key, the parties may need to perform a key confirmation protocol. See Annex D.5.1.3 of IEEE 1363-2000 for more details.

2—*(Repeated use of a key pair.)* A given public/private key pair may be used by either party for any number of key agreement operations, depending on the implementation.

3—*(Key derivation parameters.)* Depending on the key derivation function, there may be security-related constraints on the set of allowed key derivation parameters. The interpretation of these parameters is left to the implementation. For instance, it may contain key-specific information, protocol-related public information, and supplementary, private information. For security, the interpretation should be unambiguous. See Annex D.5.1.4 of IEEE 1363-2000 for further discussion.

4—*(Attributes of the shared secret key.)* The attributes of the shared secret key depend on the particular key agreement scheme used, the attributes of the public/private key pairs, the nature of the parameters to the key derivation function, and whether or not key confirmation is performed. See Annex D.5.1 of IEEE 1363-2000 for further discussion of the attributes of the shared secret key.

5—*(Error conditions.)* The two parties may produce errors under certain conditions, such as the following:

— private key not found in step 2
— public key not found in step 4
— public key not valid in step 5

— private key or public key not supported in step 6
— key derivation parameter not supported in step 7

Such error conditions should be detected and handled appropriately by an implementation, but the specific methods for detecting and handling them are outside of the scope of this standard.

4.2 IBKAS-IDAK

IBKAS-IDAK is Identity Based Key Agreement Scheme, IDAK version. Each party contributes two key pairs.

4.2.1 Scheme Options

The following options shall be established or otherwise agreed upon between the parties to the scheme:

— a secret value derivation primitive, which shall be
  — IBSVDP-IDAK
— a key derivation function, which should be KDF1 (Section 13.1 of IEEE 1363-2000) or a function designated for use with IBKAS-IDAK in an addendum to this standard

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

4.2.2 Key Agreement Operation

A sequence of shared secret keys K_1, K_2, …, K_t shall be generated by each party by performing the following or an equivalent sequence of steps:

1. Obtain the valid set of EC domain parameters with which the parties' two identity based key pairs shall be associated.
2. Obtain the identity based private key w and select a valid key pair (u, v) for the operation, associated with the parameters established in Step 1.
3. Obtain the other party's identity based public key w' and the purported public key v' for the operation, associated with the parameters established in Step 1.
4. Compute a shared secret value z from the selected private keys w and u and the other party's two public keys w' and v' with the selected secret value derivation primitive.
5. Convert the shared secret value z to an octet string Z using FE2OSP.
6. For each shared secret key to be agreed on:
   a. Establish or otherwise agree on key derivation parameters P_i for the key.
   b. Derive a shared secret key K_i from the octet string Z and the key derivation parameters P_i with the selected key derivation function (see Section 9.4.1).

Conformance region recommendation. A conformance region should include:
— at least one valid set of domain parameters
— at least one valid private key w for each set of domain parameters
— all valid key pairs (u, v) associated with the same set of domain parameters as w
— all valid public keys w' and v' associated with the same set of domain parameters as s; if key validation is performed, invalid public keys w' and v' that are appropriately handled by the implementation may also be included in the conformance region
— a range of key derivation parameters P

4.3 IBKAS-SCK

IBKAS-SCK is Identity Based Key Agreement Scheme, SCK version. Security of this scheme was proved in [ChenChengSmart07]. Each party contributes two key pairs.

4.3.1 Scheme Options

The following options shall be established or otherwise agreed upon between the parties to the scheme:

- a secret value derivation primitive, which shall be
  - IBSVDP-SCK
- a key derivation function, which should be KDF1 (Section 13.1 of IEEE 1363-2000) or a function designated for use with IBKAS-SCK in an addendum to this standard

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

4.3.2 Key Agreement Operation

A sequence of shared secret keys K_1, K_2, …, K_t shall be generated by each party by performing the following or an equivalent sequence of steps:

1. Obtain the valid set of EC domain parameters with which the parties' two identity based key pairs shall be associated.
2. Obtain the identity based private key d and select a valid key pair (x, $E = xP$) **--- Should this operation be addressed by using another primitive?** for the operation, associated with the parameters established in Step 1.
3. Obtain the other party's identity based public key Q and the purported public key E' for the operation, associated with the parameters established in Step 1.
4. Compute a shared secret value z from the selected private keys d and x and the other party's two public keys Q and E' with the selected secret value derivation primitive --- **this primitive hasn't been specified**.
5. Convert the shared secret value z to an octet string Z using FE2OSP.
6. For each shared secret key to be agreed on:
   a. Establish or otherwise agree on key derivation parameters P_i for the key.
   b. Derive a shared secret key K_i from the octet string Z and the key derivation parameters P_i with the selected key derivation function (see Section 9.4.1).

Conformance region recommendation. A conformance region should include:

- at least one valid set of domain parameters
- at least one valid private key d for each set of domain parameters
- all valid key pairs (x, E) associated with the same set of domain parameters as s
- all valid public keys Q and E' associated with the same set of domain parameters as s; if key validation is performed, invalid public keys Q and E' that are appropriately handled by the implementation may also be included in the conformance region
- a range of key derivation parameters P