PKCS #3: Diffie-Hellman Key-Agreement Standard

An RSA Laboratories Technical Note
Version 1.4
Revised November 1, 1993*

1. Scope

This standard describes a method for implementing Diffie-Hellman key agreement, whereby two parties, without any prior arrangements, can agree upon a secret key that is known only to them (and, in particular, is not known to an eavesdropper listening to the dialogue by which the parties agree on the key). This secret key can then be used, for example, to encrypt further communications between the parties.

The intended application of this standard is in protocols for establishing secure connections, such as those proposed for OSI's transport and network layers [ISO90a][ISO90b].

Details on the interpretation of the agreed-upon secret key are outside the scope of this standard, as are details on sources of the pseudorandom bits required by this standard.

2. References

X.208	CCITT. *Recommendation X.208: Specification of Abstract Syntax Notation One (ASN.1).* 1988.
X.509	CCITT. *Recommendation X.509: The Directory – Authentication Framework.* 1988.
[DH76]	W. Diffie and M.E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory,* IT-22:644–654, 1976.

*Supersedes June 3, 1991 version, which was also published as NIST/OSI Implementors' Workshop document SEC-SIG-91-19. PKCS documents are available by electronic mail to `<pkcs@rsa.com>`.

[Sch90] C.P. Schnorr. Efficient identification and signatures for smart cards. In G. Brassard, editor, *Advances in Cryptology – CRYPTO '89 Proceedings,* volume 435 of *Lecture Notes in Computer Science,* pages 239–251. Springer-Verlag, New York, 1990.

[ISO90a] ISO. *JTC1/SC6/N6285: Draft Transport Layer Security Protocol.* Draft, November 1990.

[ISO90b] ISO. *JTC1/SC6/N2559: Draft Network Layer Security Protocol.* Draft, September 1990.

3. Definitions

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

`AlgorithmIdentifier`: A type that identifies an algorithm (by object identifier) and any associated parameters. This type is defined in X.509.

ASN.1: Abstract Syntax Notation One, as defined in X.208.

Diffie-Hellman parameters: Prime and base.

Diffie-Hellman: The Diffie-Hellman key-agreement protocol, elsewhere called "exponential key agreement," as defined in [DH76].

4. Symbols and abbreviations

Upper-case italic symbols (e.g., *PV*) denote octet strings; lower-case italic symbols (e.g., *g*) denote integers.

PV	public value	p	prime
PV'	other's public value	x	private value
SK	secret key	x'	other's private value
g	base	y	integer public value
k	length of prime in octets	y'	other's integer public value
l	length of private value in bits	z	integer secret key
mod n	modulo n		

5. General overview

The next four sections specify parameter generation, two phases of Diffie-Hellman key agreement, and an object identifier.

A central authority shall generate Diffie-Hellman parameters, and the two phases of key agreement shall be performed with these parameters. It is possible that more than one instance of parameters may be generated by a given central authority, and that there may be more than one central authority. Indeed, each entity may be its own central authority, with different entities having different parameters. The algorithm identifier for Diffie-Hellman key agreement specifies which Diffie-Hellman parameters are employed.

Two entities intending to agree on a secret key shall employ the first phase independently to produce outputs PV and PV', the public values. The entities shall exchange the outputs.

The entities shall then employ the second phase independently with the other entity's public value as input. The mathematics of Diffie-Hellman key agreement ensure that the outputs SK of the second phase are the same for both entities.

6. Parameter generation

This section describes Diffie-Hellman parameter generation.

A central authority shall select an odd prime p. The central authority shall also select an integer g, the base, that satisfies $0 < g < p$. The central authority may optionally select an integer l, the private-value length in bits, that satisfies $2^{l-1} \leq p$.

The length of the prime p in octets is the integer k satisfying

$$2^{8(k-1)} \leq p < 2^{8k} .$$

Notes.

1. The cost of some methods for computing discrete logarithms depends on the the length of the prime, while the cost of others depends on the length of the private value. The intention of selecting a private-value length is to reduce the computation time for key agreement, while maintaining a given level of security. A similar optimization is suggested by Schnorr [Sch90].

2. Some additional conditions on the choice of prime, base, and private-value length may well be taken into account in order to deter discrete logarithm computation. These security conditions fall outside the scope of this standard.

7. Phase I

This section describes the first phase of Diffie-Hellman key agreement.

The first phase consists of three steps: private-value generation, exponentiation, and integer-to-octet-string conversion. The input to the first phase shall be the Diffie-Hellman parameters. The output from the first phase shall be an octet string PV, the public value; and an integer x, the private value.

This phase is performed independently by the two parties intending to agree on a secret key.

7.1 Private-value generation

An integer x, the private value, shall be generated privately and randomly. This integer shall satisfy $0 < x < p-1$, unless the central authority specifies a private-value length l, in which case the integer shall satisfy $2^{l-1} \leq x < 2^l$.

7.2 Exponentiation

The base g shall be raised to the private value x modulo p to give an integer y, the integer public value.

$$y = g^x \bmod p,\ \ 0 < y < p\ .$$

This is the classic discrete-exponentiation computation.

7.3 Integer-to-octet-string conversion

The integer public value y shall be converted to an octet string PV of length k, the public value. The public value PV shall satisfy

$$y = \textbf{Error!}\ , \qquad (1)$$

where $PV_1, \ldots, PV_k$ are the octets of PV from first to last.

In other words, the first octet of PV has the most significance in the integer and the last octet of PV has the least significance.

8. Phase II

This section describes the second phase of Diffie-Hellman key agreement.

The second phase consists of three steps: octet-string-to-integer conversion, exponentiation, and integer-to-octet-string conversion. The input to the second phase shall be the Diffie-Hellman parameters; an octet string PV', the other entity's public value; and the private value x. The output from the second phase shall be an octet string SK, the agreed-upon secret key.

This phase is performed independently by the two parties intending to agree on a secret key, after the parties have exchanged public values resulting from the first phase.

8.1 Octet-string-to-integer conversion

The other entity's public value PV' shall be converted to an integer y', the other entity's integer public value. Let $PV'_1, \ldots, PV'_k$ be the octets of PV' from first to last. Then the other entity's integer public value y' shall satisfy

$$y' = \textbf{Error!} \, .$$

In other words, the first octet of PV' has the most significance in the integer and the last octet of PV' has the least significance.

8.2 Exponentiation

The other entity's integer public value y' shall be raised to the private integer x modulo p to give an integer z, the integer secret key.

$$z = (y')^x \bmod p, \ \ 0 < z < p \, .$$

This is the classic discrete-exponentiation computation.

Note. The integer secret key z satisfies

$$z = (y')^x = (g^{x'})^x = (g^x)^{x'} = y^{x'} \bmod p \, ,$$

where x' is the other entity's private value. This mathematical relationship is the reason the two entities arrive at the same key.

8.3 Integer-to-octet-string conversion

The integer secret key z shall be converted to an octet string SK, the secret key, of length k. The secret key SK shall satisfy

$$z = \textbf{Error!} \, ,$$

where $SK_1, \ldots, SK_k$ are the octets of SK from first to last.

In other words, the first octet of SK has the most significance in the integer and the last octet of SK has the least significance.

9. Object identifier

This standard defines two object identifiers: `pkcs-3` and `dhKeyAgreement`.

The object identifier `pkcs-3` identifies this standard.

```
pkcs-3 OBJECT IDENTIFIER ::=
  { iso(1) member-body(2) US(840) rsadsi(113549)
      pkcs(1) 3 }
```

The object identifier `dhKeyAgreement` identifies the Diffie-Hellman key agreement method defined in Sections 7 and 8.

```
dhKeyAgreement OBJECT IDENTIFIER ::= { pkcs-3 1 }
```

The `dhKeyAgreement` object identifier is intended to be used in the `algorithm` field of a value of type `AlgorithmIdentifier`. The `parameters` field of that type, which has the algorithm-specific syntax `ANY DEFINED BY algorithm`, would have ASN.1 type `DHParameter` for this algorithm.

```
DHParameter ::= SEQUENCE {
  prime INTEGER, -- p
  base INTEGER, -- g
  privateValueLength INTEGER OPTIONAL }
```

The fields of type `DHParameter` have the following meanings:

- `prime` is the prime *p*.
- `base` is the base *g*.
- `privateValueLength` is the optional private-value length *l*.

Revision history

Versions 1.0–1.2

Versions 1.0–1.2 were distributed to participants in RSA Data Security, Inc.'s Public-Key Cryptography Standards meetings in February and March 1991.

Version 1.3

Version 1.3 is part of the June 3, 1991 initial public release of PKCS. Version 1.3 was published as NIST/OSI Implementors' Workshop document SEC-SIG-91-19.

Version 1.4

Version 1.4 incorporates several editorial changes, including updates to the references and the addition of a revision history. The following substantive changes were made:

- Section 6: Parameter generation is modified to allow central authority to select private-value length in bits.
- Section 7.1: Private-value generation is modified to handle private-value length.
- Section 9: Optional `privateValueLength` field is added to `DHParameter` type.

Author's address

RSA Laboratories (415) 595-7703
100 Marine Parkway (415) 595-4126 (fax)
Redwood City, CA 94065 USA `pkcs-editor@rsa.com`