

## Triple-DES and RC2 Key Wrapping

<[draft-ietf-smime-key-wrap-00.txt](#)>

### Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#). Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at  
<http://www.ietf.org/shadow.html>.

To view the entire list of current Internet-Drafts, please check the "lid-abstracts.txt" listing contained in the Internet-Drafts Shadow Directories on ftp.is.co.za (Africa), ftp.nordu.net (Northern Europe), ftp.nis.garr.it (Southern Europe), munnari.oz.au (Pacific Rim), ftp.ietf.org (US East Coast), or ftp.isi.edu (US West Coast).

### Abstract

This document specifies the algorithm for wrapping one Triple-DES [3DES] key with another Triple-DES key and the algorithm for wrapping one RC2 [RC2] key with another RC2 key. These key wrap algorithms were originally published in [section 12.6 of RFC 2630](#) [CMS]. They are republished since these key wrap algorithms have been found to be useful in contexts beyond those supported by [RFC 2630](#).

This draft is being discussed on the "ietf-smime" mailing list. To join the list, send a message to <ietf-smime-request@imc.org> with the single word "subscribe" in the body of the message. Also, there

is a Web site for the mailing list at <<http://www.imc.org/ietf-smime/>>.

---

INTERNET DRAFT

September 2001

## 1 Introduction

Management of symmetric cryptographic keys often leads to situations where one symmetric key is used to encrypt (or wrap) another. Key wrap algorithms are commonly used in two situations. First, key agreement algorithms (such as Diffie-Hellman [DH-X9.42]) generate a pairwise key-encryption key, and a key wrap algorithm is used to encrypt the content-encryption key or a multicast key with the pairwise key-encryption key. Second, a key wrap algorithm is used to encrypt the content-encryption key, multicast key, or session key in a locally generated storage key-encryption key or a key-encryption key that was distributed out-of-band.

This document specifies the algorithm for wrapping one Triple-DES key with another Triple-DES key [3DES] and specifies the algorithm for wrapping one RC2 key with another RC2 key [RC2]. Encryption of a Triple-DES key with another Triple-DES key uses the algorithm specified in [section 3](#). Encryption of a RC2 key with another RC2 key uses the algorithm specified in [section 4](#). Both of these algorithms rely on the key checksum algorithm specified in [section 2](#). Triple-DES and RC2 content-encryption keys are encrypted in Cipher Block Chaining (CBC) mode [MODES].

In this document, the key words MUST, MUST NOT, REQUIRED, SHOULD, SHOULD NOT, RECOMMENDED, and MAY are to be interpreted as described by Scott Bradner in [STDWORDS].

The same key wrap algorithm is used for both Two-key Triple-DES and Three-key Triple-DES keys. When a Two-key Triple-DES key is to be wrapped, a third DES key with the same value as the first DES key is created. Thus, all wrapped Triple-DES keys include three DES keys. However, a Two-key Triple-DES key MUST NOT be used to wrap a Three-key Triple-DES key that is comprised of three unique DES keys.

RC2 supports variable length keys. RC2 128-bit keys MUST be used as

key-encryption keys; however, the wrapped RC2 key MAY be of any size.

## [2](#) Key Checksum

The key checksum algorithm is used to provide a key integrity check value. The algorithm is:

1. Compute a 20 octet SHA-1 [SHA1] message digest on the key that is to be wrapped.
2. Use the most significant (first) eight octets of the message digest value as the checksum value.

## [3](#) Triple-DES Key Wrapping and Unwrapping

This section specifies the algorithms for wrapping and unwrapping one Triple-DES key with another Triple-DES key [3DES].

### [3.1](#) Triple-DES Key Wrap

The Triple-DES key wrap algorithm encrypts a Triple-DES key with a Triple-DES key-encryption key. The Triple-DES key wrap algorithm is:

1. Set odd parity for each of the DES key octets comprising the Three-Key Triple-DES key that is to be wrapped, call the result CEK.
2. Compute an 8 octet key checksum value on CEK as described above in [Section 2](#), call the result ICV.
3. Let CEKICV = CEK || ICV.
4. Generate 8 octets at random, call the result IV.
5. Encrypt CEKICV in CBC mode using the key-encryption key. Use the random value generated in the previous step as the initialization vector (IV). Call the ciphertext TEMP1.
6. Let TEMP2 = IV || TEMP1.
7. Reverse the order of the octets in TEMP2. That is, the most significant (first) octet is swapped with the least significant (last) octet, and so on. Call the result TEMP3.
8. Encrypt TEMP3 in CBC mode using the key-encryption key. Use an initialization vector (IV) of 0x4adda22c79e82105. The ciphertext is 40 octets long.

Note: When the same Three-Key Triple-DES key is wrapped in different key-encryption keys, a fresh initialization vector (IV) must be generated for each invocation of the key wrap algorithm.

### [3.2](#) Triple-DES Key Unwrap

The Triple-DES key unwrap algorithm decrypts a Triple-DES key using a Triple-DES key-encryption key. The Triple-DES key unwrap algorithm is:

1. If the wrapped key is not 40 octets, then error.
2. Decrypt the wrapped key in CBC mode using the key-encryption key. Use an initialization vector (IV) of 0x4adda22c79e82105. Call the output TEMP3.
3. Reverse the order of the octets in TEMP3. That is, the most significant (first) octet is swapped with the least significant (last) octet, and so on. Call the result TEMP2.
4. Decompose TEMP2 into IV and TEMP1. IV is the most significant (first) 8 octets, and TEMP1 is the least significant (last) 32 octets.

5. Decrypt TEMP1 in CBC mode using the key-encryption key. Use the IV value from the previous step as the initialization vector. Call the ciphertext CEKICV.
6. Decompose CEKICV into CEK and ICV. CEK is the most significant (first) 24 octets, and ICV is the least significant (last) 8 octets.
7. Compute an 8 octet key checksum value on CEK as described above in [Section 2](#). If the computed key checksum value does not match the decrypted key checksum value, ICV, then error.
8. Check for odd parity each of the DES key octets comprising CEK. If parity is incorrect, then error.
9. Use CEK as a Triple-DES key.

### [3.3](#) Triple-DES Key Wrap Algorithm Identifier

Some security protocols employ ASN.1 [X.208-88, X.209-88], and these protocols employ algorithm identifiers to name cryptographic algorithms. To support these protocols, the Triple-DES key wrap algorithm has been assigned the following algorithm identifier:

id-alg-CMS3DESwrap OBJECT IDENTIFIER ::= { iso(1) member-body(2)

```
us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) alg(3) 6 }
```

The AlgorithmIdentifier parameter field MUST be NULL.

## [4](#) RC2 Key Wrapping and Unwrapping

This section specifies the algorithms for wrapping and unwrapping one RC2 key with another RC2 key [RC2].

### [4.1](#) RC2 Key Wrap

The RC2 key wrap algorithm encrypts a RC2 key with a RC2 key-encryption key. The RC2 key wrap algorithm is:

1. Let the RC2 key be called CEK, and let the length of CEK in octets be called LENGTH. LENGTH is a single octet.
2. Let LCEK = LENGTH || CEK.
3. Let LCEKPAD = LCEK || PAD. If the length of LCEK is a multiple of 8, the PAD has a length of zero. If the length of LCEK is not a multiple of 8, then PAD contains the fewest number of random octets to make the length of LCEKPAD a multiple of 8.
4. Compute an 8 octet key checksum value on LCEKPAD as described above in [Section 2](#), call the result ICV.
5. Let LCEKPADICV = LCEKPAD || ICV.
6. Generate 8 octets at random, call the result IV.
7. Encrypt LCEKPADICV in CBC mode using the key-encryption key. Use the random value generated in the previous step as the

- initialization vector (IV). Call the ciphertext TEMP1.
8. Let TEMP2 = IV || TEMP1.
  9. Reverse the order of the octets in TEMP2. That is, the most significant (first) octet is swapped with the least significant (last) octet, and so on. Call the result TEMP3.
  10. Encrypt TEMP3 in CBC mode using the key-encryption key. Use an initialization vector (IV) of 0x4adda22c79e82105.

Note: When the same RC2 key is wrapped in different key-encryption keys, a fresh initialization vector (IV) must be generated for each invocation of the key wrap algorithm.

### [4.2](#) RC2 Key Unwrap

The RC2 key unwrap algorithm decrypts a RC2 key using a RC2 key-encryption key. The RC2 key unwrap algorithm is:

1. If the wrapped key is not a multiple of 8 octets, then error.
2. Decrypt the wrapped key in CBC mode using the key-encryption key. Use an initialization vector (IV) of 0x4adda22c79e82105. Call the output TEMP3.
3. Reverse the order of the octets in TEMP3. That is, the most significant (first) octet is swapped with the least significant (last) octet, and so on. Call the result TEMP2.
4. Decompose the TEMP2 into IV and TEMP1. IV is the most significant (first) 8 octets, and TEMP1 is the remaining octets.
5. Decrypt TEMP1 in CBC mode using the key-encryption key. Use the IV value from the previous step as the initialization vector. Call the plaintext LCEKPADICV.
6. Decompose the LCEKPADICV into LCEKPAD, and ICV. ICV is the least significant (last) octet 8 octets. LCEKPAD is the remaining octets.
7. Compute an 8 octet key checksum value on LCEKPAD as described above in [Section 2](#). If the computed key checksum value does not match the decrypted key checksum value, ICV, then error.
8. Decompose the LCEKPAD into LENGTH, CEK, and PAD. LENGTH is the most significant (first) octet. CEK is the following LENGTH octets. PAD is the remaining octets, if any.
9. If the length of PAD is more than 7 octets, then error.
10. Use CEK as an RC2 key.

#### [4.3](#) RC2 Key Wrap Algorithm Identifier

Some security protocols employ ASN.1 [X.208-88, X.209-88], and these protocols employ algorithm identifiers to name cryptographic algorithms. To support these protocols, the RC2 key wrap algorithm has been assigned the following algorithm identifier:

```
id-alg-CMSRC2wrap OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) alg(3) 7 }
```

The AlgorithmIdentifier parameter field MUST be RC2wrapParameter:

```
RC2wrapParameter ::= RC2ParameterVersion
```

RC2ParameterVersion ::= INTEGER

The RC2 effective-key-bits (key size) greater than 32 and less than 256 is encoded in the RC2ParameterVersion. For the effective-key-bits of 40, 64, and 128, the rc2ParameterVersion values are 160, 120, and 58 respectively. These values are not simply the RC2 key length. Note that the value 160 must be encoded as two octets (00 A0), because the one octet (A0) encoding represents a negative number.

## References

- 3DES      American National Standards Institute. ANSI X9.52-1998, Triple Data Encryption Algorithm Modes of Operation. 1998.
- CMS      Housley, R., "Cryptographic Message Syntax", [RFC 2630](#), June 1999.
- DES      American National Standards Institute. ANSI X3.106, "American National Standard for Information Systems - Data Link Encryption". 1983.
- DH-X9.42    Rescorla, E. Diffie-Hellman Key Agreement Method. [RFC 2631](#). June 1999.
- DSS      National Institute of Standards and Technology. FIPS Pub 186: Digital Signature Standard. 19 May 1994.
- MODES      National Institute of Standards and Technology. FIPS Pub 81: DES Modes of Operation. 2 December 1980.
- RANDOM    Eastlake, D., S. Crocker, and J. Schiller. Randomness Recommendations for Security. [RFC 1750](#). December 1994.
- RC2      Rivest, R. A Description of the RC2 (r) Encryption Algorithm. [RFC 2268](#). March 1998.
- SHA1      National Institute of Standards and Technology. FIPS Pub 180-1: Secure Hash Standard. 17 April 1995.
- STDWORDS    Bradner, S. Key Words for Use in RFCs to Indicate Requirement Levels. [RFC2119](#). March 1997.

- X.208-88 CCITT. Recommendation X.208: Specification of Abstract Syntax Notation One (ASN.1). 1988.
- X.209-88 CCITT. Recommendation X.209: Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1). 1988.

## Security Considerations

Implementations must protect the key-encryption key. Compromise of the key-encryption key may result in the disclosure of all keys that have been wrapped with the key-encryption key, which may lead to the disclosure of all traffic protected with those wrapped key.

Implementations must randomly generate initialization vectors (IVs) and padding. The generation of quality random numbers is difficult. [RFC 1750](#) [RANDOM] offers important guidance in this area, and Appendix 3 of FIPS Pub 186 [DSS] provides one quality PRNG technique.

If the key-encryption key and wrapped key are associated with different symmetric encryption algorithms, the effective security provided to data encrypted with the wrapped key is determined by the weaker of the two algorithms. If, for example, data is encrypted with 168-bit Triple-DES and that Triple-DES key is wrapped with a 40-bit RC2 key, then at most 40 bits of protection is provided. A trivial search to determine the value of the 40-bit RC2 key can recover Triple-DES key, and then the Triple-DES key can be used to decrypt the content. Therefore, implementers must ensure that key-encryption algorithms are as strong or stronger than content-encryption algorithms.

These key wrap algorithms specified in this document have been reviewed for use with Triple-DES and RC2, and they have not been reviewed for use with other encryption algorithms. Similarly, the key wrap algorithms make use of CBC mode [MODES], and they have not been reviewed for use with other cryptographic modes.

## Acknowledgments

This document is the result of contributions from many professionals. I appreciate the hard work of all members of the IETF S/MIME Working Group. I extend a special thanks to Carl Ellison, Peter Gutmann, Bob Jueneman, Don Johnson, and Burt Kaliski for their support in defining these algorithms.



---

INTERNET DRAFT

September 2001

## Author Address

Russell Housley  
RSA Laboratories  
918 Spring Knoll Drive  
Herndon, VA 20170  
USA

rhousley@rsasecurity.com

## Full Copyright Statement

Copyright (C) The Internet Society (2001). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. In addition, the ASN.1 module presented in [Appendix A](#) may be used in whole or in part without inclusion of the copyright notice. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process shall be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns. This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

