NIST Special Publication 800-38F
DRAFT--August, 2011

Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping

Morris Dworkin

C O M P U T E R S E C U R I T Y

NIST Special Publication 800-38F
DRAFT

Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping

Morris Dworkin

C O M P U T E R S E C U R I T Y

Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899

August 2011

U.S. Department of Commerce
Rebecca M. Blank, Acting Secretary

National Institute of Standards and Technology
Patrick D. Gallagher, Director

Reports on Information Security Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analyses to advance the development and productive use of information technology. ITL's responsibilities include the development of technical, physical, administrative, and management standards and guidelines for the cost-effective security and privacy of sensitive unclassified information in Federal computer systems. This Special Publication 800-series reports on ITL's research, guidance, and outreach efforts in computer security, and its collaborative activities with industry, government, and academic organizations.

Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

National Institute of Standards and Technology Special Publication 800-38F
Natl. Inst. Stand. Technol. Spec. Publ. 800-38F, 29 pages (August 2011)
CODEN: NSPUE2

Acknowledgements

The author wishes to thank his colleagues who reviewed drafts of this publication and contributed to its development, especially Elaine Barker, Meltem Turan, Allen Roginsky, Lily Chen, Donghoon Chang, Sharon Keller, Tim Polk, John Kelsey, and Souradyuti Paul. Heather Pearce worked extensively on an earlier version of this publication. Professor Phillip Rogaway kindly provided Figure 1 from his paper analyzing key wrap modes. The author also gratefully acknowledges the comments from the public and private sectors to improve the quality of this publication.

Abstract

This publication describes cryptographic methods that are approved for "key wrapping," i.e., the protection of the confidentiality and integrity of cryptographic keys. In addition to describing existing methods, this publication specifies two new, deterministic authenticated encryption modes of operation of the Advanced Encryption Standard (AES) algorithm: the AES Key Wrap (KW) mode and the AES Key Wrap With Padding (KWP) mode. The analogous mode with the Triple Data Encryption Algorithm (TDEA) as the underlying block cipher, called TKW, is also specified, to support legacy applications.

KEY WORDS: authenticated encryption; authentication; block cipher; computer security; confidentiality; cryptography; encryption; information security; key wrapping; mode of operation.

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

1 Purpose

This publication is the sixth part in a series of Recommendations regarding the modes of operation of block cipher algorithms. The purpose of this part is to provide approved methods for key wrapping, i.e., the protection of cryptographic keys.

2 Authority

This publication has been developed by the National Institute of Standards and Technology (NIST) in furtherance of its statutory responsibilities under the Federal Information Security Management Act (FISMA) of 2002, Public Law 107-347.

NIST is responsible for developing standards and guidelines, including minimum requirements for federal information systems, but such standards and guidelines shall not apply to national security systems.

This recommendation has been prepared for use by Federal agencies. It may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright. (Attribution would be appreciated by NIST.)

Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and binding on federal agencies by the Secretary of Commerce under statutory authority. Nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other federal official.

Conformance testing for implementations of this Recommendation will be conducted within the framework of the Cryptographic Algorithm Validation Program (CAVP) and the Cryptographic Module Validation Program (CMVP). The requirements of this Recommendation are indicated by the word "shall." Some of these requirements may be out-of-scope for CMVP or CAVP validation testing, and thus are the responsibility of entities using, implementing, installing, or configuring applications that incorporate this Recommendation.

3 Introduction

3.1 Overview

This Recommendation specifies a deterministic authenticated encryption mode of operation of the Advanced Encryption Standard (AES) algorithm. The mode is called AES Key Wrap, abbreviated as KW in this Recommendation. Although KW can be used with any reversible padding scheme, a variant of KW with a padding scheme is also specified to promote interoperability. This variant is called AES Key Wrap With Padding, abbreviated as KWP. The analogue of KW with the Triple Data Encryption Algorithm (TDEA) [5] as the underlying block cipher is also specified, to support legacy applications. This analogue is called Triple DEA Key Wrap, abbreviated as TKW.

KW, KWP, and TKW were designed to protect cryptographic keys. Each provides an option for protecting keys in a manner that is distinct from the methods that protect general data. Segregating keys from general data can provide an extra layer of protection.

Nevertheless, there is no requirement to protect cryptographic keys with a distinct cryptographic method. Previously-approved authenticated encryption modes—as well as combinations of an approved encryption mode with an approved authentication method—are approved for the protection of cryptographic keys, in addition to general data.

Similarly, KW, KWP, and TKW are each approved for the protection of general data, in addition to cryptographic keys.

KW, KWP, and TKW are each robust in the sense that each bit of output can be expected to depend in a nontrivial fashion on each bit of input, even when the length of the input data is greater than one block. This property is achieved at the cost of a considerably lower throughput rate, compared to other authenticated encryption modes, but the tradeoff may be appropriate for some key management applications. For example, a robust method may be desired when the length of the keys to be protected is greater than the block size of the underlying block cipher, or when the value of the protected data is very high.

3.2 Related Specifications

Earlier specifications of key wrap algorithms that are related to KW, KWP, and TKW are discussed in this section.

In 2001, NIST posted a document entitled "AES Key Wrap Specification" on NIST's Computer Security Resource Center web site as an unofficial suggestion for the protection of cryptographic keys. That algorithm is essentially equivalent to KW as specified in this Recommendation.

In 2002, two industry groups published key wrapping specifications that were based on the specification that NIST posted. First, the Internet Engineering Task Force (IETF) developed an essentially equivalent specification in Request For Comments (RFC) 3394 [9]. Second, the Telecommunications Industry Association published a protocol for Digital Radio Over-the-Air-Rekeying [1] containing a "Key Wrap Algorithm" that supported TDEA, in addition to the AES algorithm. Those algorithms are essentially equivalent to KW and TKW as specified in this Recommendation.

In 2008, Accredited Standards Committee X9, Inc., published a key wrap standard for the financial services industry [2]. The variant of KW in that standard, named AESKW, featured a more general framework for formatting the input data, including a padding scheme, as well as an analogue with TDEA as the underlying block cipher.

In 2009, a different padding scheme was specified in RFC 5649 [4], referencing elements of the specification in [9]. The resulting algorithm, called AES Key Wrap With Padding, is essentially equivalent to KWP as specified in this Recommendation.

The differences between KW, KWP, and TKW as specified in this Recommendation and their earlier specifications are described in Appendix B.

4 Definitions and Notation

4.1 Definitions

approved	FIPS-approved or NIST-recommended: an algorithm or technique that is either 1) specified in a FIPS or a NIST Recommendation, or 2) adopted in a FIPS or a NIST Recommendation.
authenticated encryption function	A function that encrypts plaintext into ciphertext and provides a means for the associated authenticated decryption function to verify the authenticity of either the plaintext or the ciphertext.
authenticated decryption function	A function that decrypts purported ciphertext into corresponding plaintext and verifies the authenticity of the data. The output is either the plaintext or an indication that the plaintext is not authentic.
authenticity	The property that data actually originated from its purported source.
bit	A binary digit: `0` or `1`.
bit string	A finite, ordered sequence of bits.
block	For a given block cipher, a bit string whose length is the block size of the block cipher.
block cipher	A permutation, determined from a specified family by the choice of a parameter called the key, on bit strings of a fixed length. The permutation is called the forward cipher function, and its inverse is called the inverse cipher function.
block cipher mode of operation (mode)	A function, or a pair of related functions, e.g., for encryption, authentication, or authenticated encryption, of which a block cipher is the main component.
block size	For a given block cipher and key, the fixed length of the input (or output) bit strings.

collision	In a given context, the equality of two values, usually out of a large number of possible values.
ciphertext	The confidential form of the plaintext that is the output of the authenticated encryption function.
exclusive-OR	The bitwise addition, modulo 2, of two bit strings of equal length.
integrity check value	A fixed string that is prepended to the plaintext within the authenticated encryption function of a key wrap algorithm, in order to be verified within the authenticated decryption function.
key encryption key	The block cipher key that determines the authenticated encryption and authenticated decryption functions of a key wrap algorithm. May be called a key wrapping key in other documents.
key wrap algorithm	A deterministic, symmetric-key authenticated encryption algorithm, determined by the choice of a key encryption key, that is intended for the protection of cryptographic keys. Consists of two functions: authenticated encryption and authenticated decryption.
least significant bit(s)	The right-most bit(s) of a bit string.
most significant bit(s)	The left-most bit(s) of a bit string.
mode	See “block cipher mode of operation.”
octet	A string of 8 bits.
plaintext	Usable data that is formatted as input to a mode.
prerequisite	A required input to an algorithm that has been established prior to the invocation of the algorithm.
semiblock	Given a block cipher, a bit string whose length is half of the block size.
sequence of n semiblocks	For a given block size and a positive integer, n, a string that can be represented as the concatenation of n semiblocks.

shall	Is required to. Requirements apply to conforming implementations.
should	Is recommended to.
unwrapping function	The inverse of the wrapping function.
valid length	A length for a plaintext string, or the corresponding length for a ciphertext string, that is allowed for an implementation of KW, KWP, or TKW.
wrapping function	The keyed, length-preserving permutation that is applied to an enlarged form of the plaintext within the authenticated encryption function to produce the ciphertext.

4.2 Acronyms

AES	Advanced Encryption Standard.
FIPS	Federal Information Processing Standard.
CAVP	Cryptographic Algorithm Validation Program.
CMVP	Cryptographic Module Validation Program.
FISMA	Federal Information Security Management Act.
ICV	integrity check value.
IETF	Internet Engineering Task Force.
ITL	Information Technology Lab.
KW	AES Key Wrap.
KWP	AES Key Wrap with Padding.
KEK	key encryption key.
NIST	National Institute of Standards and Technology.

RFC	Request For Comment.
TDEA	Triple Data Encryption Algorithm.
TKW	TDEA Key Wrap.

4.3 Variables

C The ciphertext.

ICV1 The 64-bit default ICV for KW: `0xA6A6A6A6A6A6A6A6`.

ICV2 The 32-bit default ICV for KWP: `0xA65959A6`.

ICV3 The 32-bit default ICV for TKW: `0xA6A6A6A6`.

P The plaintext.

4.4 Operations and Functions

0^s The bit string that consists of s consecutive '0' bits.

$\text{CIPH}_K(X)$ The output of the forward cipher function of the block cipher under the key K applied to the block X.

$\text{CIPH}^{-1}_K(X)$ The output of the inverse cipher function of the block cipher under the key K applied to the block X.

int(X) The integer for which the bit string X is the binary representation.

len(X) The bit length of the bit string X.

$\text{LSB}_s(X)$ The bit string consisting of the s right-most bits of the bit string X.

$\text{MSB}_s(X)$ The bit string consisting of the s left-most bits of the bit string X.

TW(S) The output of the wrapping function for TKW applied to the string S.

$\text{TW}^{-1}(C)$ The output of the unwrapping function for TKW applied to the string C.

W(S) The output of the wrapping function for KW and KWP applied to the bit string S.

$\text{W}^{-1}(C)$ The output of the unwrapping function for KW and KWP applied to the bit string C.

$\lceil x \rceil$	The least integer that is not less than the real number x.
$[x]_s$	The binary representation of the non-negative integer x as a string of s bits, where $x<2^s$.
$X \oplus Y$	The bitwise exclusive-OR of bit strings X and Y whose bit lengths are equal.
$X \| Y$	The concatenation of bit strings X and Y.
0x	The marker for the beginning of a hexadecimal representation of a bit string.

4.5 Examples of Basic Operations and Functions on Strings

In this publication, the `new courier` font indicates the '`0`' bit, the '`1`' bit, and any hexadecimal symbols: `0`, `1`, …, `9`, `A`, `B`, `C`, `D`, `E`, `F`.

The beginning of a hexadecimal representation of a string is marked by '0x.' For example, 0x`A659`=`1010011001011001`.

Given a real number x, the ceiling function, denoted $\lceil x \rceil$, is the least integer that is not less than x. For example, $\lceil 2.1 \rceil = 3$, and $\lceil 4 \rceil = 4$.

Given a positive integer s, 0^s denotes the string that consists of s '`0`' bits. For example, 0^8 = `00000000`.

The concatenation operation on bit strings is denoted ||. For example, `001` || `10111` = `00110111`.

Given bit strings of equal length, the exclusive-OR (XOR) operation, denoted ⊕, specifies the addition, modulo 2, of the bits in each bit position. For example, `10011` ⊕ `10101` = `00110`.

Given a bit string X, the bit length of X is denoted len(X). For example, len(`00010`)=5.

Given a bit string X and a non-negative integer s such that len(X)≥s, the functions $\text{LSB}_s(X)$ and $\text{MSB}_s(X)$ return the s least significant (right-most) bits and the s most significant (left-most) bits, respectively, of X. For example, LSB_3(`11011010`)=`010`, and MSB_4(`11011010`)=`1110`.

Given a positive integer s and a non-negative integer x that is less than 2^s, the integer-to-string function, denoted $[x]_s$, is the binary representation of x as a string of bit length s with the least significant bit on the right. For example, for the (base 10) integer 39, the binary representation (base 2) is `100111`, so $[39]_8$ = `00100111`.

Given a (non-empty) bit string X, the string-to-integer function, denoted int(X), is the integer x such that $[x]_{\text{len}(X)}=X$. In other words, int(X) is the non-negative integer less than $2^{\text{len}(X)}$ whose binary representation is X. For example, int(`00011010`) = 26.

5 Common Elements

The common elements of KW, KWP, and TKW are introduced in the following three sections, including the associated notation and requirements. The underlying block cipher and key are discussed in Sec. 5.1. The authenticated encryption and authenticated decryption functions and their core transformations, the wrapping and unwrapping functions, are discussed in Sec. 5.2. Limits on the length of the data to be protected are discussed in Sec. 5.3.

5.1 The Underlying Block Cipher and Key

The transformations of the variants of KW feature a block cipher as the main component; thus, each variant is a mode of operation (mode, for short) of the block cipher. The mode key is exactly the key for the underlying block cipher. This key is called the key encryption key (KEK), denoted K.

For any given KEK, the underlying block cipher of the mode is a permutation, i.e., an invertible function, on bit strings of a fixed length. The strings are called blocks, and the length of a block is called the block size. The permutation is also called the forward cipher function, denoted CIPH_K, to distinguish it from its inverse, called the inverse cipher function, denoted CIPH^{-1}_K.

For KW and KWP, the underlying block cipher shall be approved, including the length of the KEK, and the block size shall be 128 bits. Currently, the AES algorithm is the only block cipher that fits this profile. For TKW, the underlying block cipher is specified to be TDEA, and the length of the KEK may be any key size for which TDEA is approved; see [6]. The choice of the key size affects the security of the algorithms against brute force search, but the key size will not be explicitly indicated in the specifications. Methods for generating cryptographic keys are discussed in [7]; the goal is to select the keys uniformly at random, i.e., for each possible key to occur with equal probability.

The KEK shall be secret, i.e., disclosed only to parties that are authorized to know the protected information. Compliance with this requirement is the responsibility of the entities using, implementing, installing, or configuring applications that incorporate this Recommendation. The management of KEKs is outside the scope of this publication.

5.2 The Authenticated Encryption and Authenticated Decryption Functions

For a given KEK and block cipher, KW, KWP, and TKW are each comprised of two related functions: authenticated encryption and authenticated decryption. The authenticated encryption function takes an input string, called the plaintext, denoted P, and returns a longer output string, called the ciphertext, denoted C. The authenticated encryption function expands the data so that only a small fraction of strings of any given length are ciphertexts.

The authenticated decryption function takes an input string, called the purported ciphertext, and returns either 1) an output string or 2) a special symbol, denoted FAIL. In the first case, the

output string is the unique plaintext that corresponds to the purported ciphertext, which confirms that it is a genuine ciphertext.

For KW, the authenticated encryption function and the authenticated decryption function are denoted KW-AE and KW-AD; for KWP, the functions are denoted KWP-AE and KWP-AD; for TKW, the functions are denoted TKW-AE and TKW-AD.

Note that, although the KEK is a parameter for each of these six functions, in the specifications of these functions in this Recommendation, the KEK is considered to be a prerequisite, i.e., an input that has been established prior to the invocation of the function, and is omitted from the notation. Similarly, the choice of the block cipher for the KW and the KWP functions is a prerequisite that is omitted from the notation.

The authenticated encryption and authenticated decryption functions of KW and KWP are based on a keyed permutation, called the wrapping function, denoted W, and its inverse, called the unwrapping function, denoted W^{-1}. The analogous keyed permutations for TKW are denoted TW and TW^{-1}.

Within the authenticated encryption function, the wrapping function is applied to an enlarged plaintext string to produce the ciphertext. Each key wrap variant enlarges the plaintext by prepending a fixed string called the integrity check value (ICV); for KWP-AE, the enlarged plaintext also includes a 32-bit encoding of the octet length of the plaintext and possibly some "zero" octets as padding.

In each key wrap variant, the authenticated decryption function applies the unwrapping function to the purported ciphertext and then verifies whether the output string is the result of properly enlarging a plaintext string.

A useful unit of length for describing these functions is half the block size, i.e., 64 bits for KW and KWP, and 32 bits for TKW. A string of this length is called a semiblock, and, for a positive integer n, a string that can be represented as the concatenation of n semiblocks is called a sequence of n semiblocks.

For each key wrap variant, the wrapping function and the unwrapping function are defined on sequences of three or more semiblocks. The length of the output is the same as the length of the input.

KW-AE (or TKW-AE) is defined on any sequence of two or more semiblocks. For KWP-AE, the domain of possible inputs is extended to nonempty octet strings; the 32-bit encoding of the octet length of the plaintext within KWP-AE, however, imposes a mild restriction, namely, KWP-AE is only defined when the octet length of the plaintext is less than 2^{32}.

The authenticated decryption function for each key wrap variant is defined on any sequence of three or more semiblocks.

5.3 Limits on Data Length

Mandatory limits on plaintext lengths for each key wrap variant, and the corresponding limits on ciphertext lengths, are described in Sec. 5.3.1. Additional, implementation-specific limits on the data lengths are discussed in Sec. 5.3.2.

5.3.1 Mandatory Limits

KWP is only defined when the length of the plaintext is less than 2^{32} octets, i.e., when the plaintext is a sequence of fewer than 2^{29} semiblocks. KW can accept longer inputs; nevertheless, the plaintext for KW-AE shall be limited to a sequence of fewer than 2^{54} semiblocks. The plaintext for TKW-AE shall be limited to a sequence of fewer than 2^{28} semiblocks. Similarly, the ciphertext for KW-AD shall be limited to a sequence of no more than 2^{54} semiblocks, and the ciphertext for TKW-AD shall be limited to a sequence of no more than 2^{28} semiblocks. The motivation for these restrictions is discussed in Appendix A4. This information, along with the minimum lengths from Sec. 5.2, is summarized in Table 1:

Table 1: Summary of Limits on Data Length

Algorithm	Plaintext	Ciphertext	Reason for Upper Bounds
KW	$2 \le P < 2^{54}$ semiblocks	$3 \le C \le 2^{54}$ semiblocks	requirement—see Appendix A4
KWP	$1 \le P < 2^{32}$ octets	$2 \le C \le 2^{29}$ semiblocks	undefined on other lengths
TKW	$2 \le P < 2^{28}$ semiblocks	$3 \le C \le 2^{28}$ semiblocks	requirement—see Appendix A4

Compliance with these requirements is the responsibility of the entities using, implementing, installing, or configuring applications that incorporate this Recommendation.

5.3.2 Implementation-Specific Limits

A plaintext length, or the corresponding ciphertext length, that is allowed for an implementation of a key wrap algorithm is called a valid length.

The set of valid plaintext lengths for the authenticated encryption function may be restricted to any subset of the lengths described in Table 1. Any restricted set of plaintext lengths corresponds to a restricted set of ciphertext lengths. In particular, for KW and TKW, the ciphertext is one semiblock longer than the plaintext; for KWP, the ciphertext is one semiblock longer than the padded plaintext.

The definition of the set of valid plaintext lengths is a prerequisite for the authenticated encryption function. The corresponding set of valid ciphertext lengths shall be a prerequisite for the authenticated decryption function. Note that these sets, like the other prerequisites for these functions, must be agreed upon in order for two communicating implementations to interoperate properly.

There is a requirement in [6] that, in order to keep a Restricted status for TKW with 2-key TDEA as the underlying block cipher, the number of invocations of TK-AE shall not exceed 2^{20} for a given KEK; the Restricted status is described in that publication. There is no requirement to limit

the number of invocations for KW-AE or KWP-AE. Considerations for limiting the number of invocations of the authenticated decryption function are discussed in Appendix A3.

6 Specifications of KW and KWP

6.1 W and W^{-1}

Algorithm 1 below specifies the wrapping function, W, for KW-AE (see Sec. 6.2) and KWP-AE (see Sec. 6.3), with a given block cipher and KEK.

Algorithm 1: W(S)

Prerequisites:
approved block cipher, CIPH, with a 128-bit block size;
key, K, for CIPH.

Input:
a sequence of n semiblocks S, for $n \geq 3$.

Steps:

1. Initialize variables.
   a) Let $S_1, S_2, \ldots, S_n$ be the semiblocks such that $S = S_1 \| S_2 \| \ldots \| S_n$.
   b) Let $A^0 = S_1$.
   c) For $i = 2, \ldots, n$: let $R_i^0 = S_i$.
   d) Let $s = 6(n-1)$.
2. Calculate the intermediate values. For $t = 1, \ldots, s$, update the variables as follows:
   a) $A^t = \text{MSB}_{64}(\text{CIPH}_K(A^{t-1} \| R_2^{t-1})) \oplus [t]_{64}$;
   b) For $i = 2, \ldots, n-1$: $R_i^t = R_{i+1}^{t-1}$;
   c) $R_n^t = \text{LSB}_{64}(\text{CIPH}_K(A^{t-1} \| R_2^{t-1}))$.
3. Output the results:
   a) Let $C_1 = A^s$.
   b) For $i = 2, 3, \ldots, n$: $C_i = R_i^s$.
   c) Return $C_1 \| C_2 \| \ldots \| C_n$.

Figure 1 below illustrates the wrapping function applied to a sequence of four semiblocks, i.e., $W(S_1 \| S_2 \| S_3 \| S_4) = C_1 \| C_2 \| C_3 \| C_4$. Each "wire" carries a semiblock, and each of the eighteen numbered rectangles represents an invocation of the underlying block cipher with the KEK. On the left side of these rectangles, the input block's most significant 64 bits enter the top wire, while on the right side of these rectangles, the output block's most significant 64 bits exit the bottom wire; this convention makes for fewer wire crossings.

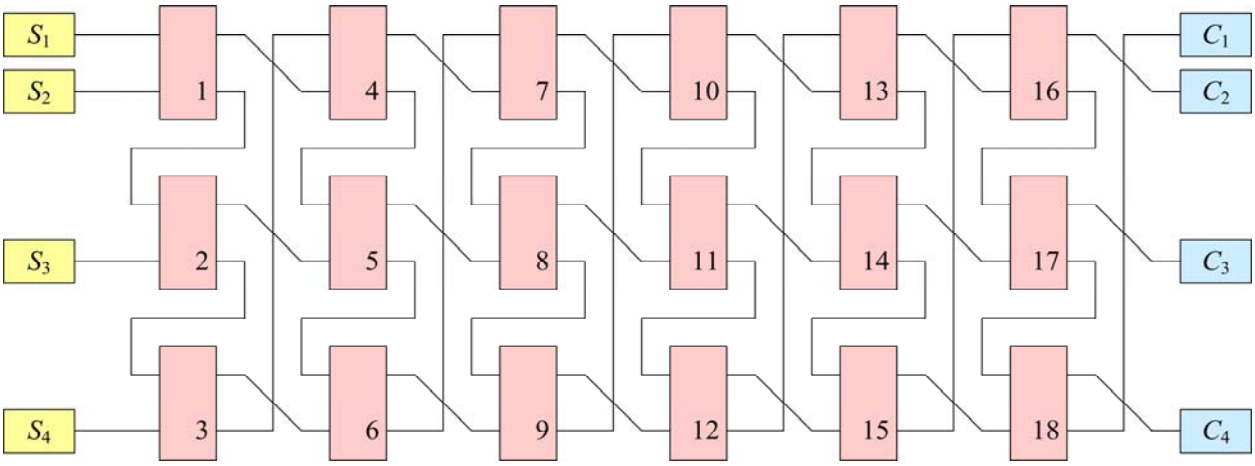

Figure 1: Illustration of the wrapping function, W

Figure 2 below illustrates the assignment of intermediate values within Step 2 of Algorithm 1. The dashed lines indicate the assignments of new values to the n semiblock variables. The variable that indexes the iterations, t, increases from 1 to $6(n-1)$.

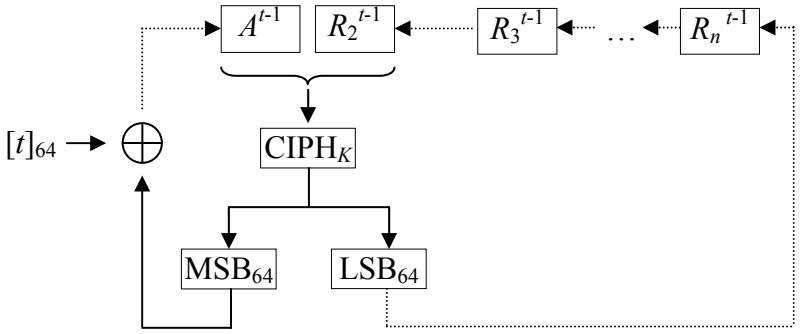

Figure 2: Illustration of an iteration within Step 2 of Algorithm 1

Algorithm 2 below specifies the unwrapping function, W^{-1}, for KW-AD (see Sec. 6.2) and KWP-AD (see Sec. 6.3), with a given block cipher and KEK.

Algorithm 2: $W^{-1}(C)$

Prerequisites:
inverse cipher function, CIPH^{-1}, of an approved block cipher with a 128-bit block size;
key, K, for CIPH.

Input:
a sequence of n semiblocks, C, for $n \geq 3$.

Steps:

1. Initialize the variables.
   a) Let $s = 6(n-1)$.
   b) Let $C_1, C_2, \ldots, C_n$ be the semiblocks such that $C = C_1 \| C_2 \| \ldots \| C_n$.
   c) Let $A^s = C_1$.
   d) For $i = 2, \ldots, n$: let $R_i^s = C_i$.
2. Calculate the intermediate values. For $t = s, s-1, \ldots, 1$, update the variables as follows:
   a) $A^{t-1} = \text{MSB}_{64}(\text{CIPH}^{-1}_K((A^t \oplus [t]_{64}) \| R_n^t))$;
   b) $R_2^{t-1} = \text{LSB}_{64}(\text{CIPH}^{-1}_K((A^t \oplus [t]_{64}) \| R_n^t))$;
   c) For $i = 3, \ldots, n$, $R_i^{t-1} = R_{i-1}^t$.
3. Output the results:
   a) Let $S_1 = A^0$.
   b) For $i = 2, \ldots, n$: $S_i = R_i^0$.
   c) Return $S_1 \| S_2 \| \ldots \| S_n$.

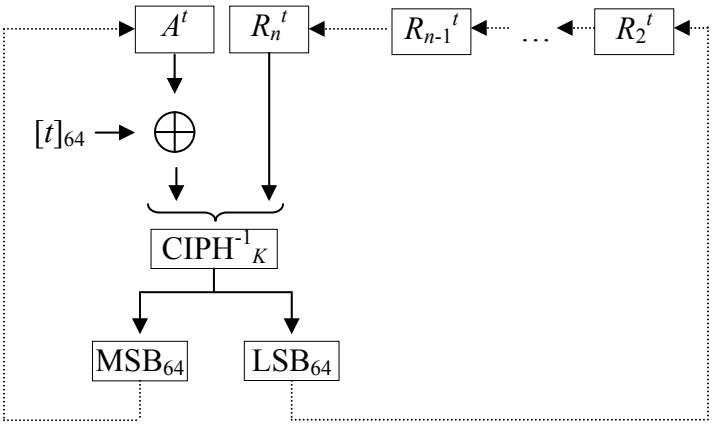

Figure 3: Illustration of an iteration within Step 2 of Algorithm 2

Figure 3 above illustrates the assignment of intermediate values within Step 2 of Algorithm 2. The dashed lines indicate the assignments of new values to the n semiblock variables. The variable that indexes the iterations, t, decreases from $6(n-1)$ to 1. The input to the inverse cipher function is the concatenation of the two semiblocks that are indicated by the incoming arrows.

6.2 KW

Algorithm 3 below specifies the authenticated encryption function for KW for a given block cipher and KEK. The wrapping function, W, specified in Algorithm 1 above, is invoked in Step 3 with the same block cipher and KEK as prerequisites.

Algorithm 3: KW-AE(*P*)

Prerequisites:
approved block cipher, CIPH, with a 128-bit block size;
key encryption key, *K*;
definition of the set of valid plaintext lengths.

Input:
plaintext *P*, with valid length.

Output:
ciphertext C.

Steps:
1. Let *ICV*1=0x`A6A6A6A6A6A6A6A6`.
2. Let *S*=*ICV*1 || *P*.
3. Return *C*=W(*S*).

Algorithm 4 below specifies the authenticated decryption function for KW for a given block cipher and KEK. The unwrapping function, W^{-1}, specified in Algorithm 2 above, is invoked in Step 4 with the same block cipher and KEK.

Algorithm 4: KW-AD(*C*)

Prerequisites:
approved block cipher, CIPH, with a 128-bit block size;
key encryption key, *K*;
definition of the set of valid ciphertext lengths.

Input:
purported ciphertext, *C*.

Output:
Plaintext *P* or indication of inauthenticity, *FAIL*.

Steps:
1. If the length of *C* is not valid, then return FAIL and stop.
2. Let *n* be the number of semiblocks in *C*.
3. Let *ICV*1=0x`A6A6A6A6A6A6A6A6`.
4. Let $S = W^{-1}(C)$.
5. If $MSB_{64}(S) \neq ICV1$, then return FAIL and stop.
6. Return $LSB_{64(n-1)}(S)$.

6.3 KWP

Algorithm 5 below specifies the authenticated encryption function for KWP for a given block cipher and KEK. The wrapping function, W, specified in Algorithm 1 above, is invoked in Step 5 with the same block cipher and KEK.

Algorithm 5: KWP-AE(*P*)

Prerequisites:
approved block cipher, CIPH, with a 128-bit block size;
key encryption key, *K*;
definition of the set of valid plaintext lengths.

Input:
plaintext *P*, with valid length.

Output:
ciphertext C.

Steps:
1. Let *ICV2*=0x`A65959A6`.
2. Let *padlen*=$8\cdot\lceil \text{len}(P)/64 \rceil - \text{len}(P)/8$.
3. Let *PAD*=$0^{8padlen}$.
4. Let $S = ICV2 \,||\, [\text{len}(P)/8]_{32} \,||\, P \,||\, PAD$.
5. If len(*P*)≤64, then return $C = \text{CIPH}_K(S)$; if len(*P*)>64, then return $C = \text{W}(S)$.

Algorithm 6 below specifies the authenticated decryption function for KWP for a given block cipher and KEK. The unwrapping function, W^{-1}, specified in Algorithm 2 above, is invoked in Step 4 with the same block cipher and KEK.

Algorithm 6: KWP-AD(*C*)

Prerequisites:
approved block cipher, CIPH, with a 128-bit block size;
key encryption key, *K*;
definition of the set of valid ciphertext lengths.

Input:
purported ciphertext, *C*.

Output:
Plaintext *P* or indication of inauthenticity, *FAIL*.

Steps:
1. Let *n* be the number of semiblocks in *C*.
2. If *n* is not valid, then return FAIL and stop.
3. Let *ICV*2=0x`A65959A6`.
4. If *n*=2, then let $S = \text{CIPH}^{-1}_K(C)$; if *n*>2, then let $S = \text{W}^{-1}(C)$.
5. If $\text{MSB}_{32}(S) \neq ICV2$, then return FAIL and stop.
6. Let *Plen*=int($\text{LSB}_{32}(\text{MSB}_{64}(S))$).
7. Let *padlen*=8(*n*-1)-*Plen*.
8. If *padlen*<0 or *padlen*>7, then return FAIL and stop.

9. If $\text{LSB}_{8padlen}(S) \neq 0^{8padlen}$, then return FAIL and stop.
10. Return $\text{MSB}_{8Plen}(\text{LSB}_{64(n-1)}(S))$.

7 Specification of TKW

7.1 TW and TW^{-1}

Algorithm 7 below specifies the wrapping function, TW, for the authenticated encryption function of TKW (see Sec. 7.2) with a given KEK, K. In this algorithm, CIPH_K denotes the forward cipher function of TDEA with K as the key.

Algorithm 7: TW(S)

Prerequisites:
key, K, for TDEA.

Input:
sequence of n semiblocks S, for $n \geq 3$.

Steps:
1. Initialize the variables.
   a) Let $S_1, S_2, \ldots, S_n$ be the semiblocks such that $S = S_1 \| S_2 \| \ldots \| S_n$.
   b) Let $A^0 = S_1$.
   c) For $i = 2, \ldots, n$, let $R_i^0 = S_i$.
   d) Let $s = 6(n-1)$.
2. Calculate the intermediate values. For $t = 1, \ldots, s$, update the variables as follows:
   a) $A^t = \text{MSB}_{32}(\text{CIPH}_K(A^{t-1} \| R_2^{t-1})) \oplus [t]_{32}$;
   b) For $i = 2, \ldots, n-1$: $R_i^t = R_{i+1}^{t-1}$;
   c) $R_n^t = \text{LSB}_{32}(\text{CIPH}_K(A^{t-1} \| R_2^{t-1}))$.
3. Output the results:
   a) Let $C_1 = A^s$.
   b) For $i = 2, \ldots, n$: $C_i = R_i^s$.
   c) Return $C_1 \| C_2 \| \ldots \| C_n$.

Algorithm 8 below specifies the unwrapping function, TW^{-1}, for the authenticated decryption function of TKW (see Sec. 7.2) with a given KEK, K. In this algorithm, CIPH^{-1}_K denotes the inverse cipher function of TDEA with K as the key..

Algorithm 8: $\text{TW}^{-1}(C)$

Prerequisites:
key, K, for TDEA.

Input:
sequence of n semiblocks C, for $n \geq 3$.

Steps:

1. Initialize the variables.
    a) Let $s = 6(n-1)$.
    b) Let $C_1, C_2, \ldots, C_n$ be the semiblocks such that $C=C_1 \| C_2 \| \ldots \| C_n$.
    c) Set $A^s = C_1$.
    d) For $i = 2, \ldots, n$: $R_i^s = C_i$.
2. Calculate the intermediate values. For $t = s, s-1, \ldots, 1$, update the variables as follows:
    a) $A^{t-1} = \text{MSB}_{32}(\text{CIPH}^{-1}_K((A^t \oplus [t]_{32}) \| R_n^t))$;
    b) $R_2^{t-1} = \text{LSB}_{32}(\text{CIPH}^{-1}_K((A^t \oplus [t]_{32}) \| R_n^t))$;
    c) For $i = 3, \ldots, n$: $R_i^{t-1} = R_{i-1}^t$.
3. Output the results:
    a) Let $S_1 = A^0$.
    b) For $i = 2, \ldots, n$: $S_i = R_i^0$.
    c) Return $S_1 \| S_2 \| \ldots \| S_n$.

7.2 TKW

Algorithm 9 below specifies the authenticated encryption function for TKW for a given TDEA key. The wrapping function, TW, specified in Algorithm 7 above, is invoked in Step 3 with the same key.

Algorithm 9: TKW-AE(*P*)

Prerequisites:
KEK, *K*;
definition of the set of valid plaintext lengths.

Input:
plaintext *P*, with valid length.

Output:
ciphertext C.

Steps:

1. Let *ICV*3=0x`A6A6A6A6`.
2. Let $S = ICV3 \| P$.
3. Return C=TW(S).

Algorithm 10 below specifies the authenticated decryption function for TKW for a given TDEA key. The unwrapping function, TW^{-1}, specified in Algorithm 8 above, is invoked in Step 4 with the key.

Algorithm 10: TKW-AD(*C*)

Prerequisites:
TDEA with key, *K*;
definition of the set of valid ciphertext lengths.

Input:
purported ciphertext, *C*.

Output:
Plaintext *P* or indication of inauthenticity, *FAIL*.

Steps:

1. If the length of *C* is not valid, then return FAIL and stop.
2. Let n+1 be the number of semiblocks in *C*.
3. Let *ICV*3=0x`A6A6A6A6`.
4. Let $S = TW^{-1}(C)$.
5. If $MSB_{32}(S) \neq ICV3$, then return FAIL and stop.
6. Return $LSB_{32n}(S)$.

8 Conformance

An implementation may claim conformance to one or more of the following six functions: KW-AE, KW-AD, KWP-AE, KWP-AD, TKW-AE, and TKW-AD. TKW-AE and TKW-AD are approved to support legacy systems but should not be used for new applications.

The associated wrapping and unwrapping functions, W, W^{-1}, TW, and TW^{-1}, are not approved for use independently of these six functions.

Implementations of the authenticated encryption and authenticated decryption functions may restrict the lengths of the plaintext, as discussed in Sec. 5.3.2, as long as at least one bit length is supported. Any such restrictions may affect interoperability.

For every algorithm that is specified in this Recommendation, a conforming implementation may replace the given set of steps with any mathematically equivalent set of steps. In other words, different procedures that produce the correct output for any input are permitted.

Appendix A: Some Security Considerations

A1 Equality of Plaintexts

Each key wrap algorithm in this Recommendation is deterministic: for a given block cipher and KEK, any invocation of the authenticated encryption function on a given plaintext produces the same ciphertext. It follows that any pair of ciphertexts reveals whether the corresponding plaintexts are equal.

Therefore, ideally, each plaintext should be encrypted only once. If a system encrypts any plaintext more than once, then one method for ensuring that the ciphertexts are different would be to prepend each plaintext with a fixed-length nonce before invoking the authenticated encryption function. Upon authenticated decryption, the nonce would be discarded.

A2 Implied Strength of Protected Keys

The disclosure of a KEK potentially compromises any data (i.e., any key) that the KEK protects. Therefore, the cryptographic strength of the protected key is limited implicitly by the length of the KEK, which determines the resistance of the KEK to brute force search. This illustrates the general principle that the KEK should be at least as strong cryptographically as any key that it protects.

A3 Authentication Assurance

The expansion of the plaintext within the authenticated encryption function provides the mechanism whereby assurance of the authenticity of the data can be obtained upon the execution of the authenticated decryption function. The nature of this assurance depends on the output of the authenticated decryption function:

- If the output is a plaintext, i.e., not FAIL, then the design of the mode provides strong, but not absolute, assurance that this plaintext is authentic, i.e., that the ciphertext, as received, was generated by a party with access to the KEK.

- If the output is *FAIL*, then it is certain that the ciphertext is not authentic.

In the first case, the assurance is not absolute because forgeries are possible, in principle. In other words, an adversary, i.e., a party without access to the key or to the authenticated encryption function, may be able to produce a genuine ciphertext, for example, by a lucky guess.

In particular, if the adversary chooses a string at random with a valid ciphertext length, the probability that the string will be a genuine ciphertext is exactly 1 in 2^{64} for KW, and approximately 1 in 2^{64} for KWP. The probability for TKW is greater, 1 in 2^{32}, so TKW is significantly more vulnerable to forgeries. For this reason, TKW should not be used for new applications.

Given repeated attempts, of course, the adversary can increase the probability that a randomly-generated string will eventually be accepted as a valid ciphertext. The system or protocol that implements the authenticated decryption function should monitor and, if necessary, limit the number of unsuccessful verification attempts for each key.

A4 Forgeries of Extremely Long Messages

The motivation for the limits on the length of the plaintext in Sec. 5.3 is the following observation on the unwrapping function, due to John Kelsey of NIST: if S is extremely long, about 2^{67} semiblocks, and if T and U are any sequences of semiblocks of equal length, it is likely that

$$\mathrm{MSB}_{64d}(\mathrm{W}^{-1}(S||T)) = \mathrm{MSB}_{64d}(\mathrm{W}^{-1}(S||U)) \quad (1)$$

for some positive integer d.

Equation 1 will hold if six suitable intermediate values coincide within the two invocations of the unwrapping function in the equation. Each pair of coinciding values is called a collision.

In particular, let S be a sequence of m semiblocks, and let T and U be distinct semiblocks. Let A^1, A^2, …, A^{6m} be the semiblocks defined within Algorithm 2 for $\mathrm{W}^{-1}(S||T)$, and let B^1, B^2, …, B^{6m} be the corresponding semiblocks for $\mathrm{W}^{-1}(S||\underline{U})$. Let $i1$, $i2$, …, $i6$ be indices that satisfy the following "collision conditions":

$0 < i1 < m$,	$A^{i1}=B^{i1}$,
$i1+n < i2$,	$A^{i2}=B^{i2}$,
$i2+n < i3$,	$A^{i3}=B^{i3}$,
$i3+n < i4$,	$A^{i4}=B^{i4}$,
$i4+n < i5$,	$A^{i5}=B^{i5}$,
$i5+n < i6 \leq 6m$, and	$A^{i6}=B^{i6}$.

These conditions imply that collisions will occur at many other intermediate values, e.g., $A^j = B^j$ for each index j such that $5m<j<i6$, or $4m<j<i5$, etc; collisions will also occur at these indices for the R values that are defined in Algorithm 2. For $i1$, these colliding R values are part of the output of the unwrapping function, as described in Equation 1 with $d = i1$.

The first semiblock of the output of the unwrapping function determines whether a purported ciphertext will pass the integrity check within the authenticated decryption function. Therefore, if $S||T$ is a given ciphertext for KW, then the modified ciphertext $S||U$ will also pass the integrity check provided that the collision conditions are satisfied. In other words, if six suitable collisions occur, $S||U$ will be a successful forgery. Moreover, the first $i1$-1 semiblocks of the resulting plaintext will be identical to the plaintext from which $S||T$ was generated.

For KWP, $S||U$ would also be a successful forgery if the collision conditions are satisfied and if no padding octets were appended to the plaintext during the generation of $S||T$.

For TKW, an analogue of the above analysis applies, adapted to the smaller semiblock size.

The length of the ciphertext affects the probability that some set of indices will satisfy the collision conditions. One can estimate this probability for a fixed value of m by modeling some of the individual collisions as independent events that occur with probability 2^{-64}**.** In particular, beginning at index $6m$, one analyzes the probability of a collision at exactly m indices, as follows:

- At index j, if $A^j \neq B^j$, or if $j<m$, one next considers the index j-1.
- At index j, if $A^j = B^j$ and $j>m$, one next considers the index j-m-1.

If this procedure identifies collisions at six or more indices, then the first six collisions will satisfy the collision conditions. Conversely, if there are six indices that satisfy the collision conditions, then this procedure will identify collisions at six or more indices. Consequently, for KW, the probability, E, that the forgery attack succeeds in this model is

$$E = 1 - \sum_{i=0}^{5} \binom{m}{i} \left(\frac{1}{2^{64}}\right)^i \left(\frac{2^{64}-1}{2^{64}}\right)^{m-i} . \tag{2}$$

If $2^{64}/m$ is sufficiently large, then

$$E \approx \frac{1}{720} \cdot \left(\frac{m}{2^{64}}\right)^6 . \tag{3}$$

Consequently, if $m<2^{54}$, as required in Sec. 5.3.1, then $E<2^{-64}$.

For TKW, the analogous conclusion is that if $m<2^{28}$, as required in Sec. 5.3.1, then $E<2^{-32}$.

A5 Additional Analysis

Within [8], Rogaway and Shrimpton provide an analysis of AESKW as specified in [2], much of which also applies to the key wrap variants in this Recommendation. Among other criticisms, the authors emphasize the lack of a proof that the underlying structure of KW meets the goal of deterministic authenticated encryption that they formalize in the paper. Nevertheless, the authors expect that the AES Key Wrap achieves this property, possibly even in a particularly strong manner, i.e., with "beyond-birthday-phenomenon security."

Appendix B: Comparison with Earlier Specifications

This appendix describes the technical differences between KW, KWP, and TKW as specified in this Recommendation and their earlier specification documents discussed in Sec. 3.2.

The maximum plaintext lengths for KW and TKW in Sec 5.3.1, and the explicit option for implementation-defined subsets of valid lengths for KW, KWP, and TKW in Sec. 5.3.2 are unique to this Recommendation. In particular, KW-AE, KWP-AE, and TKW-AE are undefined on plaintexts of invalid length, while KW-AD, KWP-AD, and TKW-AD check whether the length of the purported ciphertext is valid.

This Recommendation also differs slightly in its support of underlying block ciphers. The AES Key Wrap, both in the original "AES Key Wrap Specification" that was posted on NIST's Computer Security Resource Center web and in [9], is defined only for the AES algorithm; however, KW and KWP will also support any 128-bit block cipher that is approved in the future. The Key Wrap algorithm in [1] supports a wider choice of block ciphers, encompassing a variant that is equivalent to TKW, but also allowing TDEA with key lengths that are not supported in TKW.

Otherwise, the specification of KW in this Recommendation is equivalent to the original "AES Key Wrap Specification" and to the specification in [9], and almost equivalent to the key wrap specification in [1] with the AES algorithm as the underlying block cipher. The specification in [1] supports the appending of random padding bits to plaintexts that are not sequences of semiblocks, when the length of plaintext is fixed. This amounts to a reversible padding scheme that is different than the padding scheme defined in [4] and later adopted for KWP.

References

[1] ANSI/TIA-102.AACA-1-2002: *Project 25 – Digital Radio Over-the-Air-Rekeying (OTAR) Protocol: Addendum 1 – Key Management Security Requirements for Type 3 Block Encryption Algorithms*, Telecommunications Industry Association, November, 2002.

[2] ANS X9.102-2008, *Symmetric Key Cryptography For the Financial Services Industry—Wrapping of Keys and Associated Data*, Accredited Standards Committee X9, Inc., June, 2008.

[3] Federal Information Processing Standards (FIPS) Publication 197, *Advanced Encryption Standard*, U.S. DoC/NIST, November 26, 2001.

[4] R. Housley and M. Dworkin, *Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm*, RFC 5649, August, 2009.

[5] NIST Special Publication 800-67 Version 1.*1: Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher*, May, 2008.

[6] NIST Special Publication 800-131A: *Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths*, January, 2011.

[7] NIST Special Publication 800-133: *Recommendation for Cryptographic Key Generation*, draft, August 2011.

[8] P. Rogaway and T. Shrimpton, *Deterministic Authenticated-Encryption: A Provable-Security Treatment of the Keywrap Problem*, EUROCRYPT 2006. LNCS vol. 4004, 373-390, Springer, 2006.

[9] J. Schaad and R. Housley, *Advanced Encryption Standard (AES) Key Wrap Algorithm*, RFC 3394, September, 2002.