2008年度版リストガイド
（メッセージ認証コード）

平成21年3月

独立行政法人情報通信研究機構
独立行政法人情報処理推進機構

目 次

1　メッセージ認証コード

1.1　本文書の位置付け

1.1.1　文書の目的

通信データや蓄積データの偽造、改ざん、破損等を検知し、データの完全性を保証することは電子政府システムにおいて非常に重要である。ブロック暗号やハッシュ関数などの共通鍵暗号系の技術を応用して完全性のチェックを行うメカニズムはメッセージ認証コード MAC (Message Authentication Code) と呼ばれる。

本リストガイドでは、ハッシュ関数ベースのメッセージ認証コードである HMAC、ブロック暗号ベースのメッセージ認証コードである CBC-MAC および CMAC の技術概要および実装仕様を記述する。

1.1.2　対象とする利用目的

メッセージ認証コードは、相手認証、データの完全性保証に利用可能である。メッセージ認証コードは、共通鍵暗号系の技術を応用しており、電子署名とは異なり、事前に鍵の共有が必要である。

1.1.3　本文書の構成

本文書の構成は以下の通りである。2.2 節で用語および記号の定義を記述し、2.3 節でメッセージ認証コードの技術概要について述べ、2.4 節で HMAC, CBC-MAC, CMAC の実装仕様を記述する。

1.2　定義

1.2.1　用語定義

メッセージ認証コード (MAC)　メッセージ生成元の認証およびデータの完全性を検証するメカニズムおよび検証に使われるデータ。認証対象のデータに鍵を作用させ、暗号学的な変換を行うことで生成される。

鍵　メッセージ認証コードの生成および検証に用いるために秘密に保持すべきビット列

完全性　データおよび処理方法が正確であることおよび完全であることを保護すること

ハッシュ関数　任意長のメッセージから、定められた固定長のメッセージに変換する関数であり、暗号学的に一定の性質をもつことが期待されるもの

鍵付きハッシュ関数　鍵を入力に含んだハッシュ関数

繰り返し型ハッシュ関数　ハッシュ関数の代表的な構成法で、圧縮関数と呼ばれる固定長入力固定長出力の関数を繰り返し適用するハッシュ関数

圧縮関数　繰り返し型ハッシュ関数において繰り返し適用される固定長入力固定長出力の関数

ビット　2 進の数。0 または 1

ビット列　0 と 1 からなる順序付けられた列

ブロック暗号　固定長の入力 (64 ビットや 128 ビットなど) をとり、同じ長さの出力を生成する逆変換可能な関数群。鍵によってパラメータ化される。

ブロックサイズ　ブロック暗号における入出力ビット列の長さ

AES　Advanced Encryption Standard

ANSI　American National Standards Institute

CBC　Cipher Block Chainig

CMAC　Cipher-based MAC

FIPS　Federal Information Processing Standard

HMAC　Hash function-based MAC

IEC　International Electrotechnical Commission

ISO　International Organization for Standardization

MAC　Message Authentication Code

NIST　National Institute of Standards and Technology

PRF　Pseudo Random Function

PRP　Pseudo Random Permutation

RFC　Request for Comments

SP　Special Publication

TDEA　Triple Data Encryption Algorithm

1.2.2 記号、および記法の定義

0x	:	16 進記法
$\lceil x \rceil$	:	x 以上の最小の整数
$\|\|$	:	連結
$\oplus$	:	排他的論理和演算
$Enc_K(X)$	:	ブロック暗号 Enc により鍵 K を用いてメッセージブロック X を暗号化した結果
$\mathrm{LSB}_s(X)$	:	ビット列 X の右 s ビットを切り出したビット列
$\mathrm{MSB}_s(X)$	:	ビット列 X の左 s ビットを切り出したビット列
$X \ll 1$	:	ビット列 X を左に 1 ビットシフトし、右にビット 0 を付加したビット列
$lg(x)$	:	x の 2 を底とする対数
0^s	:	s 個のビット"0" からなるビット系列

1.3 技術概要

1.3.1 暗号技術の利用モデル

メッセージ認証コード技術では、メッセージ認証コード (MAC) の生成者および検証者の 2 つのエンティティが存在する。メッセージ認証コードは、これらの 2 つのエンティティ間で事前に共有された鍵を用いて、メッセージの偽造、改ざん、破損等を検知し、メッセージの生成元が確かであること、およびそのデータの完全性を保証する技術である。

メッセージ認証コード技術では、図 1 に示すように MAC 生成者 (送信者)A および MAC 検証者 (受信者)B は事前に鍵 K を共有し、それぞれ秘密に保持している。送信者は MAC 生成アルゴリズムを用いて、メッセージ M と鍵 K からメッセージ認証コード $T = \mathrm{MAC}(K, M)$ を計算し、メッセージ M とメッセージ認証コード T のペア (M, T) を受信者に送る。受信者は受け取ったメッセージ M' と鍵 K から MAC 検証アルゴリズムを用いて、メッセージ認証コード $T' = \mathrm{MAC}(K, M')$ を計算し、$T' = T$ かどうかを検証する。$T' = T$ であれば VALID, そうでなければ INVALID (改ざん検出信号) を出力する。

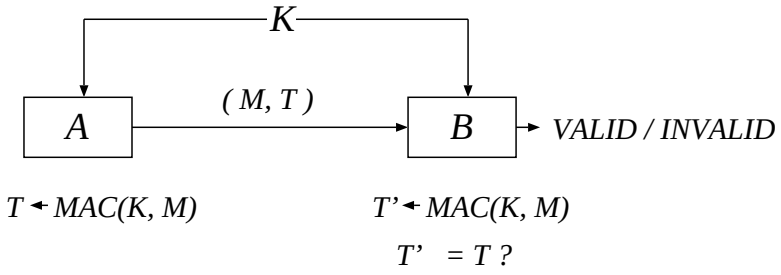

図 1: メッセージ認証コードのモデル

1.3.2 技術の基本的構成

メッセージ認証コード技術の基本的構成として、MAC 生成および MAC 検証がある。これらを実行するアルゴリズムについて説明する。

MAC 生成アルゴリズム　MAC 生成アルゴリズムは、ある正当なエンティティがメッセージ認証コードを生成するために用いるアルゴリズムであり、認証対象のデータ M と鍵 K を入力として、メッセージ認証コード T を出力する。

MAC 検証アルゴリズム　MAC 検証アルゴリズムは、ある正当なエンティティがメッセージ認証コードを検証するために用いるアルゴリズムである。認証対象のデータ M と鍵 K を入力として、メッセージ認証コードを計算し、これが受信されたメッセージ認証コードと一致していれば VALID、一致しなければ INVALID(改ざん検出信号) を出力する。

MAC 生成アルゴリズムに求められるセキュリティ要件として、適応的選択平文攻撃に対して存在的偽造困難であることが求められる。すなわち、攻撃者が MAC 生成アルゴリズムに自由にアクセスして、攻撃者に都合のよいメッセージに対する MAC を入手できたとしても、メッセージ認証コード生成アルゴリズムにより MAC が生成されたことのないメッセージに対して正当な MAC を偽造することができないことを示す必要がある。

MAC 生成アルゴリズムは、擬似ランダム性を満たしているものが多い。これにより、MAC 生成アルゴリズムが、擬似ランダム関数として鍵導出等の用途に用いられることもある[1]。

また、MAC の生成と検証は、電子署名とは異なり、同じ共通鍵を使って行われる。これにより電子署名のような否認不可性は持たない。

1.3.3　評価観点と比較

本リストガイドでは、メッセージ認証コードとして HMAC, CBC-MAC, CMAC の仕様を示す。本節では、それぞれの技術概要および主な特徴を示し、比較を行う。

HMAC　HMAC は、ハッシュ関数ベースのメッセージ認証コードであり、SHA-2 などのハッシュ関数を用いて、安全で効率のよい鍵付きハッシュ関数 (keyed hash function) を構成できる。HMAC アルゴリズムでは、ハッシュ関数 2 回分の計算量で MAC を計算することができる。

HMAC の安全性は、HMAC で使われているハッシュ関数内の圧縮関数が擬似ランダム関数 (PRF) であれば HMAC も擬似ランダム関数 (PRF) であり、理想的なメッセージ認証コードとしての安全性を満たしていることが Bellare により示されている [B06]。

しかしながら、実際に利用されるハッシュ関数には擬似ランダム関数 (PRF) とは言えない特性があり、例えば、ハッシュ関数 MD5 が HMAC で利用された場合 (HMAC-MD5 と表記される)、鍵が導出できることが示されている。

HMAC 内で使用するハッシュ関数は、プロトコルで特定のアルゴリズムが指定[2]されていなければ、電子政府推奨暗号リストに記載されたものを使用することが望ましい。

---

[1]例えば、HMAC は TLS プロトコルにおける key generation (RFC2246) や IPsec プロトコルの IKE (Internet Key Exchange) における key derivation (RFC2409) に、AES ベースの CMAC は IPsec プロトコル IKE2 (Internet Key Exchange version 2) (RFC4306) における擬似ランダム関数としてさまざまな目的で用いられている (RFC4615)。

[2]例えば、HMAC-SHA-1 は、SSH プロトコル (RFC4253) において必須実装となっている。

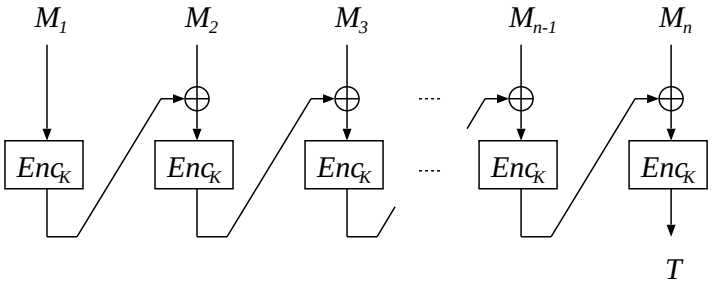

図 2: CBC-MAC のアルゴリズム

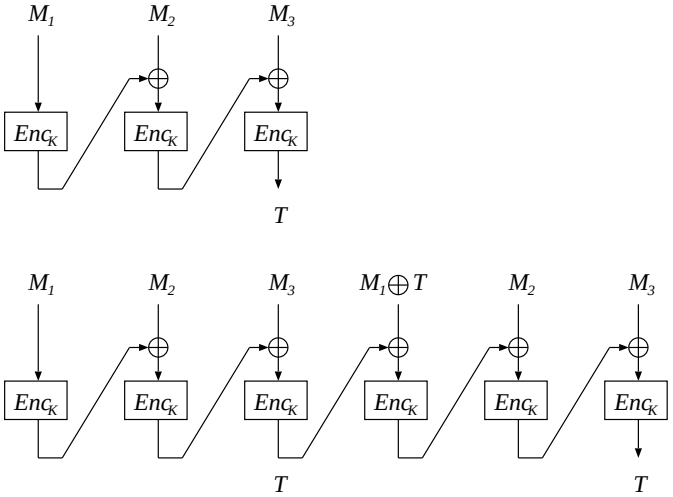

図 3: CBC-MAC の偽造の例

CBC-MAC　CBC-MAC は、ブロック暗号ベースのメッセージ認証コードであり、AES などのブロック暗号を用いて実現できる。CBC-MAC の基本的なアルゴリズムは、初期ベクトルを $IV = 0$ として CBC モードで認証対象のデータを暗号化し、暗号文の最終ブロックを MAC として利用するものである (図 2 参照)。

CBC-MAC の安全性については、メッセージ長が固定の場合、ブロック暗号が理想的な擬似ランダム置換 (PRP) であれば、CBC-MAC は擬似ランダム関数 (PRF) であり、MAC として安全であることが証明されている [BKR94]。しかしながらメッセージ長が可変の場合、容易に MAC の偽造ができることが知られている [MOV96]。

図 3 に CBC-MAC の偽造の例を示す。例えば攻撃者が $(M_1||M_2||M_3)$ という 3 ブロックメッセージに対して鍵 K を用いて生成された MAC T を入手できた時、攻撃者は、別の例えば 6 ブロックメッセージ $(M_1||M_2||M_3||(M_1 \oplus T)||M_2||M3)$ に対する正当な MAC $(= T)$ を知ることが可能となる。

CBC-MAC は従来は広く利用されていたが、近年ではこのような安全性上の問題点から規格等で新しく採用されることは少ない。また、CBC-MAC が規定されていた FIPS 113 [FIPS113] も 2008 年に廃止されている。CBC-MAC は、ISO 16609 [ISO16609] で TDEA を用いた CBC-MAC が規定されているなど金融業界で利用されているため、互換性維持の観点から本リストガイドに記述しているが、新規の利用には推奨しない。

CMAC　CMAC は、CBC-MAC の安全性上の問題点を解決するために提案された方式である。CBC-MAC の改良方式はいくつか存在し、例えば ISO/IEC 9797-1 [ISO/IEC 9797-1] では 6 種類の CBC-MAC ベースの MAC が規定されている。このうち CMAC が鍵長や安全性証明の観点で優れていることから NIST SP800-38B [SP800-38B] に採用されている。

CMAC は、CBC-MAC と同様、ブロック暗号を用いてメッセージ認証コードを構成する方式である。ブロック暗号は、電子政府推奨暗号リストに記載された方式から選ぶことが望ましい。ブロック暗号は、128 ビットブロック暗号、64 ビットブロック暗号のいずれを用いることも可能であるが、64 ビットブロック暗号を用いる場合は、種々の安全性上の検討を行う必要がある。詳細は 1.4.4 節、1.4.5 節を参照のこと。

メッセージ認証コード HMAC, CBC-MAC, CMAC の比較を表 1 に示す。

表 1: メッセージ認証コードの比較

アルゴリズム	HMAC	CBC-MAC	CMAC
プリミティブ	ハッシュ関数	ブロック暗号	ブロック暗号
計算コスト	ハッシュ関数 2 回	ブロック暗号 $\left\lceil \frac{\text{メッセージ長}}{\text{ブロックサイズ}} \right\rceil$ 回	ブロック暗号 $\left\lceil \frac{\text{メッセージ長}}{\text{ブロックサイズ}} \right\rceil + 1$ 回
安全性	特に問題なし	安全であるためには、メッセージ長の扱いに強い条件がかかる. この条件が達成されない限り 安全でなく、偽造が可能.	特に問題なし
パラメータ 鍵長 MAC 長	 ハッシュ長/2 バイト以上 ハッシュ長	 ブロック暗号の鍵長 ブロック暗号のブロック長	 ブロック暗号の鍵長 ブロック暗号のブロック長
標準化状況	FIPS 198 (2002) ISO/IEC 9797-2 (2002) RFC 2104 (1997) ANS X9.71 (2000)	FIPS 113 (2008 廃止) ISO/IEC 9797-1 (1999) ISO 16609 (2004) ISO/TR 19038 (2005) ANS X9.19 (1998)	NIST SP 800-38B (2005) ISO/IEC 9797-1 (改訂中) RFC 4493 (2006)

1.4　実装仕様

1.4.1　HMAC の仕様

HMAC に関する定義　HMAC の実装仕様の説明で用いる記号を定義する。

- B　：ハッシュ関数への入力のブロックサイズ (バイト)
- H　：HMAC 内で使うハッシュ関数
- $ipad$　：内部パッド。1 バイトデータ `0x36` を B 個連結した定数
- K　：送信者と受信者で共有される鍵
- K_0　：鍵 K から事前処理により生成される B バイトの鍵
- L　：ハッシュ関数の出力のブロックサイズ (バイト)
- $opad$　：外部パッド。1 バイトデータ `0x5c` を B 個連結した定数
- T　：MAC 値
- $Tlen$　：MAC のサイズ (ビット)
- M　：MAC に入力されるデータ. 認証対象のデータで鍵データは含まない。 M のビット長 (メッセージサイズ) $Mlen$ は $0 \leq Mlen < 2^B - 8B$。

HMAC の鍵　鍵 K の長さは $L/2$ バイト以上でなければならない。例えば、HMAC 内で用いるハッシュ関数が SHA-256 であれば、$L = 256/8 = 32$ バイトであるので、HMAC の鍵長は 128 ビット以上でなければならない。鍵長の上限はないが、鍵長が B バイト以上の場合は鍵データのハッシュをとった値を利用する。参考までに、ISO/IEC 9797-2 では、HMAC の鍵長をハッシュ長 (L バイト) 以上ブロック長 (B バイト) 以下と規定している。

HMAC の鍵は、適切な鍵生成法を用いてランダムに選び、適切な秘密レベルで管理される必要がある。また、鍵は同じものを使い続けるのではなく、定期的に変更されなければならない。

HMAC 出力の切捨て (truncation)　MAC 生成アルゴリズムにおいて、出力の一部を切捨てた短いビット列が MAC として使われることがしばしばある。本ガイドラインでは HMAC の切捨て (truncation) を行う場合、出力の左 $Tlen$ ビットを MAC とすることとし、切り捨てた後の MAC サイズ $Tlen$ を以下のように定める。

$$32 \leq Tlen \leq 8L$$

但し、プロトコルやアプリケーションで MAC の生成回数が制限されるようになっていない場合は、$Tlen$ は $4L$ ビット以上でなければならない。

HMAC 生成　HMAC による、メッセージ M に対する MAC 生成手順は以下の通りである。

$$\mathrm{HMAC}(K, M, Tlen) = H((K_0 \oplus opad)||H((K_0 \oplus ipad)||M))$$

HMAC($K, M, Tlen$)

前提条件　H: ハッシュ関数

入力　K: HMAC の鍵
M: 認証対象のメッセージ
$Tlen$: MAC のサイズ

出力　T: $Tlen$ ビットの MAC 値

手順
1. 鍵 K の長さが B と等しければ、$K_0 = K$ とし、手順 4. に進む。
2. 鍵 K の長さが B より大きければ、K のハッシュ値を計算して $(B - L)$ 個の 0 を付加した B バイトデータを K_0 とし、手順 4. に進む。
   $K_0 = H(K)||00...00$
3. 鍵 K の長さが B より小さければ、K の末尾に 0 を付加して B バイトとしたデータを K_0 とし、手順 4. に進む。
4. K_0 と $ipad$ の排他的論理和 $K_0 \oplus ipad$ を計算する。
5. 手順 4. で生成したデータに M を付加して $(K_0 \oplus ipad)||M$ を計算する。
6. 手順 5. で生成したデータのハッシュ値 $H((K_0 \oplus ipad)||M)$ を計算する。
7. K_0 と $opad$ の排他的論理和 $K_0 \oplus opad$ を計算する。
8. 手順 7. で生成したデータに手順 6. で生成したデータを付加して $(K_0 \oplus opad)||H((K_0 \oplus ipad)||M)$ を計算する。

9. 手順 8. で生成したデータのハッシュ値 $H((K_0 \oplus opad)||H((K_0 \oplus ipad)||M))$ を計算する。
10. 手順 9. で生成したデータの左 $Tlen$ ビットを MAC 値 T とする。

HMAC 検証　HMAC による、MAC 検証手順は以下の通りである。

HMAC-VER(K, M, T')

前提条件　H: ハッシュ関数

入力　K: 鍵
M: 認証対象のメッセージ
T': 受け取った MAC 値

出力　VALID または INVALID

手順
1. 1.4.1 節の HMAC 生成手順により、メッセージ M から MAC T を生成する。
2. $T = T'$ なら VALID を、そうでなければ INVALID を出力する。

1.4.2 CBC-MAC の仕様

CBC-MAC に関する定義　CBC-MAC の実装仕様の説明で用いる記号を定義する。

b	:	ブロックサイズ (ビット)
K	:	ブロック暗号の鍵
M	:	メッセージ
M_i	:	フォーマットされたメッセージの i 番目のメッセージブロック
M_n^*	:	フォーマットされたメッセージの最終のメッセージブロック。ブロックサイズに満たない場合もある。
$Mlen$	:	メッセージサイズ (ビット)
n	:	フォーマットされたメッセージに含まれるブロック数
T	:	MAC 値
$Tlen$	:	MAC のサイズ (ビット)

CBC-MAC 生成　CBC-MAC による、メッセージ M に対する MAC 生成手順は以下の通りである。

CBC-MAC(K, M, $Tlen$)

前提条件　Enc: b ビットブロック暗号

入力　K: 鍵
M: メッセージ長 $Mlen$ のメッセージ
$Tlen$: MAC のサイズ

出力　T: $Tlen$ ビットの MAC 値

手順
1. $n = \lceil Mlen/b \rceil$ とする。$n = Mlen/b$ ならば $n = n + 1$ とする。
2. メッセージ M を b ビットのブロック単位で分割し、$M = M_1||M_2||\ldots||M_{n-1}||M_n^*$ とする。

3. 最終メッセージブロックを $M_n = M_n^*||10^j$ とする。但し、$j = nb - Mlen - 1$ である。
4. $C_0 = 0^b$ とする。
5. $i = 1$ から n に対し、$C_i = Enc_K(C_{i-1} \oplus M_i)$ を計算する。
6. $T = \mathrm{MSB}_{Tlen}(C_n)$ とする。
7. T を出力する。

CBC-MAC 検証　CBC-MAC による、MAC 検証手順は以下の通りである。

CBC-MAC-VER(K, M, T')

前提条件　Enc: b ビットブロック暗号

入力　K: 鍵
M: 認証対象のメッセージ
T': 受け取った MAC 値

出力　VALID または INVALID

手順
1. 1.4.2 節の CBC-MAC 生成手順により、メッセージ M から MAC T を生成する。
2. $T = T'$ なら VALID を、そうでなければ INVALID を出力する。

1.4.3　CMAC の仕様

CMAC に関する定義　CMAC の実装仕様の説明で用いる記号を定義する。

b	:	ブロックサイズ (ビット)
R_b	:	副鍵生成に用いる定数。$R_{128} = 0^{120}10000111$, $R_{64} = 0^{59}11011$.
K	:	ブロック暗号の鍵
$K1, K2$	:	K から導出する 2 つの副鍵
M	:	メッセージ
M_i	:	フォーマットされたメッセージの i 番目のメッセージブロック
M_n^*	:	フォーマットされたメッセージの最終のメッセージブロック。ブロックサイズに満たない場合もある。
$Mlen$	:	メッセージサイズ (ビット)
n	:	フォーマットされたメッセージに含まれるブロック数
T	:	MAC 値
$Tlen$	:	MAC のサイズ (ビット)

CMAC の鍵　CMAC の鍵は、CMAC 内で利用されるブロック暗号の鍵として利用される。鍵は一様かつランダムに生成されなければならない。鍵は機密性を保つように管理し、他の用途と共用してはいけない。

CMAC 副鍵生成　CMAC では、事前に鍵 K から副鍵 2 つの $K1, K2$ が生成される。副鍵のサイズはともにブロックサイズ (b ビット) である。これらの値は鍵が変わらない限り固定の値である。よって事前に計算して保持しておき、繰り返し使うことができる。

副鍵生成の際の各種中間データ、特に $Enc_K(0^b)$ は秘密に管理しておかなければならない[3]。

CMAC の鍵 K から副鍵 $K1, K2$ を以下の手順で生成する。

SUBK(K)

前提条件 Enc: b ビットブロック暗号

入力 K: CMAC の鍵

出力 $K1, K2$: 副鍵

手順
1. $Enc_K(0^b)$ を計算し、L とする。
2. L の最上位ビットが 0 なら $K1 = L \ll 1$ とする。
   L の最上位ビットが 1 なら $K1 = (L \ll 1) \oplus R_b$ とする。
3. $K1$ の最上位ビットが 0 なら $K2 = K1 \ll 1$ とする。
   $K1$ の最上位ビットが 1 なら $K2 = (K1 \ll 1) \oplus R_b$ とする。
4. $K1$, $K2$ を出力する。

手順 2, 3 は有限体 GF(2^b) 上の乗算で、2 ($=0^{b-2}10$) を乗じることに対応している。R_b は GF(2^b) 上の既約多項式に対応し、それぞれ R_{128} は $u^{128} + u^7 + u^2 + u + 1$, R_{64} は $u^{64} + u^4 + u^3 + u + 1$ に対応している。

CMAC 生成　CMAC による、メッセージ M に対する MAC 生成手順は以下の通りである。ある鍵 K に対して CMAC で生成される全ての MAC のサイズ $Tlen$ は固定でなくてはならない。

CMAC($K, M, Tlen$)

前提条件 Enc: b ビットブロック暗号

入力 K: 鍵
M: メッセージ長 $Mlen$ のメッセージ
$Tlen$: MAC のサイズ

出力 T: $Tlen$ ビットの MAC 値

手順
1. 1.4.3 節の副鍵生成手順により、鍵 K から副鍵 $K1, K2$ を生成する。
2. $Mlen = 0$ なら $n = 1$ とし、$Mlen \geq 0$ なら $n = \lceil Mlen/b \rceil$ とする。
3. メッセージ M を b ビットのブロック単位で分割し、$M = M_1 || M_2 || \ldots || M_{n-1} || M_n^*$ とする。

[3] ANS X9.24 Annex C に、$Enc_K(0^b)$ (但し Enc は DES で $b = 64$) の左 16 ビットまたは 24 ビットを鍵の ID (KCV, Key Check Value) として利用する用法が記載されている。KCV は金融システムにおいて利用されているが、KCV との併用は、CMAC のみならず、いくつかの CBC-MAC ベースの MAC アルゴリズムの安全性を低下させるため、注意が必要である。

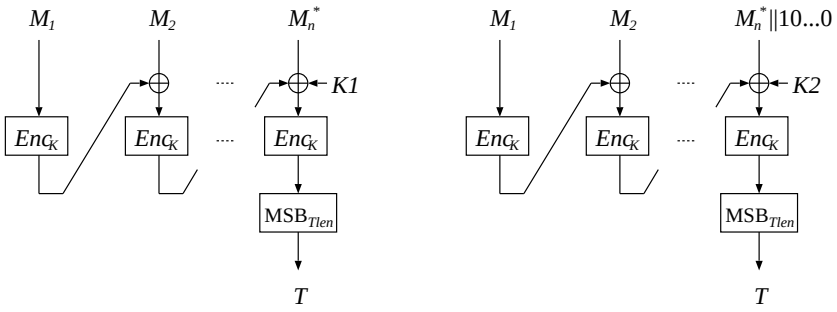

図 4: CMAC

4. M_n^* がブロック長を満たす場合、最終メッセージブロックを $M_n = K1 \oplus M_n^*$ とし、M_n^* がブロック長に満たない場合、最終メッセージブロックを $M_n = K2 \oplus (M_n^* || 10^j)$ とする。但し、$j = nb - Mlen - 1$ である。
5. $C_0 = 0^b$ とする。
6. $i = 1$ から n に対し、$C_i = Enc_K(C_{i-1} \oplus M_i)$ を計算する。
7. $T = \mathrm{MSB}_{Tlen}(C_n)$ とする。
8. T を出力する。

上記では、手順 3, 4 で事前にメッセージ全体のフォーマットを行い、その後 CBC モードを行う手順を示したが、1 ブロックずつ逐次的に実行する実装も可能である。

CMAC 検証　CMAC による、MAC 検証手順は以下の通りである。

CMAC-VER(K, M, T')

前提条件　Enc: b ビットブロック暗号

入力　K: 鍵
M: 認証対象のメッセージ
T': 受け取った MAC 値

出力　VALID または INVALID

手順
1. 1.4.3 節の CMAC 生成手順により、メッセージ M から MAC T を生成する。
2. $T = T'$ なら VALID を、そうでなければ INVALID を出力する。

1.4.4　MAC 長について

MAC 長 $Tlen$ は重要なセキュリティパラメータである。MAC 長 $Tlen$ の選択基準として、文献 [SP800-38B] の Appendix A に記載されている以下の指針が参考になる。

MAC 検証において、攻撃者は鍵 K を知らなくても単にメッセージ M に対する MAC を推測する推測攻撃 (guessing attack) は可能であり、攻撃者が MAC 値をランダムに選んだ場合、MAC 長が $Tlen$ の場合、推測攻撃の成功確率は $1/2^{Tlen}$ である。$Tlen$ が大きいほどこの攻撃は防御可能となる。また、MAC 検証失敗の際に何度も

試行を許すと、この攻撃の成功確率は高まるため、MAC 検証依頼の回数を制限することも検討すべきである。

ほとんどのアプリケーションにおいて $Tlen$ の値は少なくとも 64 ビットあれば十分であると考えられるが、64 ビット未満とする場合は注意が必要である。

MAC 出力の切捨て (truncation) を行う場合も、MAC 長は実用的な範囲でできるだけ長い方が望ましい。MAC 長の最小値として 32 ビットまで許すが、MAC 検証依頼の回数が制限される環境やアプリケーションに限るべきである。

$Tlen$ は以下の 2 つの値を用いて定量化することができる。

- $Risk$: 正しくないメッセージが検証プロセスをパスしてしまう最大許容確率
- $MaxInvalids$: ある鍵の使用期間内に、全ての実装における全ての検証プロセスにおいて INVALID (改ざん検出) を出力させる回数制限

とすると、$Tlen$ は以下の不等式を満たす必要がある。

$$Tlen \geq lg(MaxInvalids/Risk)$$

1.4.5 一つの鍵で MAC 生成を行うメッセージ数について

一つの鍵で MAC 生成を行うメッセージ数も、安全性上、重要なパラメータである。この選択基準として、文献 [SP800-38B] の Appendix B に記載されている以下の指針が参考になる。

MAC 長が $Tlen$ ビットの場合、$2^{Tlen/2}$ 個のメッセージに対して MAC 値が同じ値をとる (collision が生じる) ことが期待される。攻撃者がこの collision を利用して MAC の検証を不正にパスするのを防ぐ簡単かつ実用的な方法は、ある鍵に対して MAC を生成するメッセージ数を制限することである。一般的な用途での推奨値は、128 ビットブロック暗号を用いた CMAC の場合、一つの鍵で MAC を生成できるのは 2^{48} 個まで、64 ビットブロック暗号を用いた CMAC の場合は 2^{21} 個までとする。この場合、128 ビットブロック暗号を用いた CMAC で collision が起きる確率は 10 億分の 1、64 ビットブロック暗号を用いた CMAC で collision が起きる確率は 100 万分の 1 となる。

さらに高い安全性が求められる用途での推奨値は、128 ビットブロック暗号を用いた CMAC の場合、一つの鍵で MAC を生成できるのは 2^{48} メッセージブロック (2^{48} Gbyte) まで、64 ビットブロック暗号を用いた CMAC の場合は 2^{21} メッセージブロック (16 Mbyte) までとすべきである。

参考文献

[B06] M. Bellare, "New proofs for NMAC and HMAC: Security without collision-resistance," *Advances in Cryptology — CRYPTO '06, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20–24, 2006. Proceedings*, ed. C. Dwork, pp. 602–619, Lecture Notes in Computer Science vol. 4117, Springer-Verlag, 2006.

[BKR94] M. Bellare, J. Kilian, and P. Rogaway, "The Security of Cipher Block Chaining," *Advances in Cryptology — CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 1994. Proceedings*, ed. Y.G. Desmedt, pp. 341–358, Lecture Notes in Computer Science vol. 839, Springer-Verlag, 1994.

[FIPS113] Federal Information Processing Standards Publication 113, Computer Data Authentication, National Institute of Standards and Technology.

[FIPS198] Federal Information Processing Standards Publication 198, The Keyed-Hash Message Authentication Code (HMAC), National Institute of Standards and Technology, March 6, 2002.

[FIPS198-1] Federal Information Processing Standards Publication 198-1, The Keyed-Hash Message Authentication Code (HMAC), National Institute of Standards and Technology, July, 2008.

[ISO16609] ISO 16609: 2004, Banking – Requirements for message authentication using symmetric techniques.

[ISO/IEC 9797-1] ISO/IEC 9797-1: 1999, Information technology – Security techniques – Message Authentication Codes (MACs) – Part 1:Mechanisms using a block cipher.

[MOV96] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone, *Handbook of Applied Cryptography,* CRC Press, 1996.

[SP800-38B] NIST Special Publication 800-38B, Morris Dworkin, Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, National Institute of Standards and Technology, May, 2005.

発行日　2009年5月14日　第1版　第1刷
発行者
・〒184-8795
東京都小金井市貫井北四丁目2番1号
独立行政法人 情報通信研究機構
（情報通信セキュリティ研究センター セキュリティ基盤グループ）
NATIONAL INSTITUTE OF
INFORMATION AND COMMUNICATIONS TECHNOLOGY
4-2-1 NUKUI-KITAMACHI,KOGANEI
TOKYO,184-8795 JAPAN

・〒113-6591
東京都文京区本駒込二丁目28番8号
独立行政法人 情報処理推進機構
（セキュリティセンター 暗号グループ）
INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN
2-28-8 HONKOMAGOME,BUNKYO-KU
TOKYO,113-6591 JAPAN