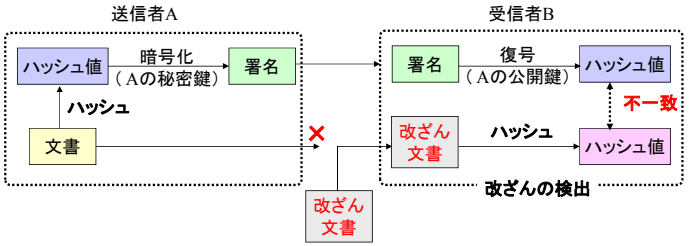
8. メッセージ認証とハッシュ

ディジタル署名

メッセージ完全性チェックの方法

1. **ディジタル署名**
2. **メッセージ認証**

ディジタル署名：公開鍵暗号によるメッセージ完全性チェックとメッセージ保証者の特定



ディジタル署名でのハッシュ

文書全体への署名でなく、ハッシュ値に対し、署名を行う理由

署名の対象となるメッセージの長さに制限があり（RSA暗号の場合の鍵長n）、
文書の分割署名が必要

⇩

○分割文書を並び換えられ、違った内容にされる恐れがある
（改ざんの検出が不可）

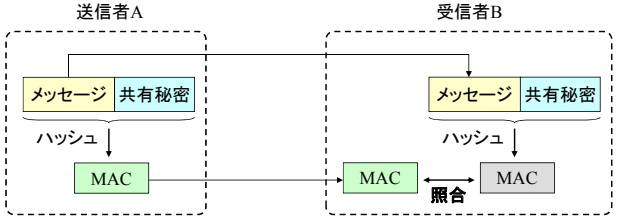
○署名の数が増え、処理時間がかかる

○文書と同じ大きさの署名になり、送信時間がかかる

メッセージ認証

メッセージ認証コード（MAC:Message Authentication Code）
メッセージ完全性コード（MIC:Message Integrity Code）

・メッセージと共有秘密情報を連結し、ハッシュ化
・メッセージ完全性チェックとメッセージ保証者の特定
・ディジタル署名より処理が高速



MAC：鍵付きハッシュ（Keyed hash）ともいう

☆

ハッシュ

ハッシュ（メッセージダイジェスト）関数

元のメッセージよりサイズが小さく、元のメッセージのダイジェスト情報（元のメッセージの内容を反映した短縮版）を出力する関数

ハッシュ関数に対する条件

（1）一方向性

・メッセージmからハッシュ値 h(m) を計算するのは容易
・h(m) が与えられた時、ハッシュ値がh(m)となるmを見つけるのが困難

（2）衝突回避性

・同じハッシュ値になる2つの異なるメッセージを見つけるのが困難
・元のメッセージが 1ビット違うだけでも、全く異なるハッシュ値 が生成される

ハッシュ関数の分類

（1）ブロック暗号を利用

（2）専用のハッシュ関数を設計

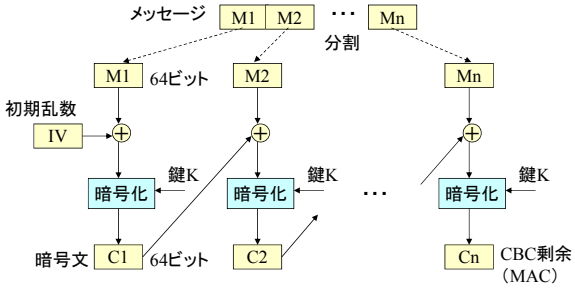
5

ブロック暗号の利用

ブロック暗号を利用したメッセージ認証（ハッシュ関数）

メッセージをCBCモードの共通鍵で暗号化し、最終結果（CBC剰余）をMAC（ハッシュ値）とする

・共通鍵（K）を知っている人しか、MACを作成できない
・メッセージの一部が変更されると、MACが変更になる



6

専用のハッシュ関数

1. MD2, MD4, MD5（MD: Message Digest）

Rivestが考案

MD2（RFC1319）
バイト単位の任意の長さのメッセージを入力し、バイト単位の操作を行い、128ビットのハッシュ値を出力

MD4（RFC1320）
ビット単位の任意の長さのメッセージを入力し、ワード（4バイト）単位の操作を行い、128ビットのハッシュ値を出力

MD5（RFC1321）
MD4の強化版

2. SHA-1, SHA-2(SHA: Secure Hashed Algorithm)

・米国NIST（連邦標準技術局）が提案した規格
・SHS(Secure Hash Standard)ともいう
・基本的な方法はMD4, MD5と同じ
・2^{64}ビット未満の長さのメッセージを入力
・SHA-1：160ビットのハッシュ値を出力
・SHA-2：出力ハッシュ値長に応じ、SHA-224, SHA-256, SHA-384, SHA-512

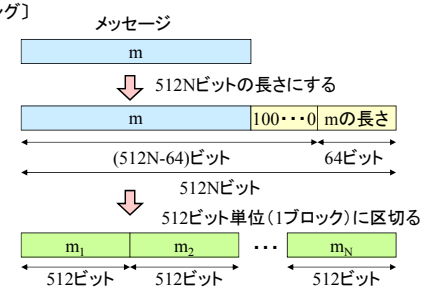
7

SHA-1

入力：2^{64}ビット未満の長さのビット列 m

出力：160ビット長のmの圧縮値 h(m)

〔パディング〕



8

SHA-1のアルゴリズム

パディングされたメッセージ m_1 m_2 ・・・ m_N

$H_0 \leftarrow h_0$, $H_1 \leftarrow h_1$, $H_2 \leftarrow h_1$, $H_3 \leftarrow h_3$, $H_4 \leftarrow h_4$　（h_0, h_1, h_1, h_3, h_4：初期値）

i = 1 ～Nまで繰り返し

ステップ1）m_i ⇨ W_0 W_1 ・・・ W_{79}

ステップ2）$A \leftarrow H_0$, $B \leftarrow H_1$, $C \leftarrow H_2$, $D \leftarrow H_3$, $E \leftarrow H_4$

ステップ3）t = 0～79まで繰り返し

$A \leftarrow g_A(W_t, K_t, A, f_t(B,C,D), E)$
$B \leftarrow g_B(A)$
$C \leftarrow g_C(B)$
$D \leftarrow g_D(C)$
$E \leftarrow g_E(D)$

ステップ4）$H_0 \leftarrow H_0+A$, $H_1 \leftarrow H_1+B$, $H_2 \leftarrow H_2+C$, $H_3 \leftarrow H_3+D$, $H_4 \leftarrow H_4+E$

ハッシュ値 H_0 H_1 H_2 H_3 H_4 160ビット

9

SHA-1のアルゴリズム

〔初期値〕
h_0 = 0x 67452301
h_1 = 0x efcdab89
h_2 = 0x 98badcfe
h_3 = 0x 10325476
h_4 = 0x c3d2e1f0
} 32ビット長

〔加算定数〕
K_t = 0x 5a827999（t = 0, ・・・, 19）
K_t = 0x 6ed9eba1（t = 20, ・・・, 39）
K_t = 0x 8f1bbcdc（t = 40, ・・・, 59）
K_t = 0x ca62c1d6（t = 60, ・・・, 79）
} 32ビット長

t：回数

〔関数〕

$f_t(x,y,z) = (x \wedge y) \vee (\neg x \wedge y)$　（t = 0, ・・・, 19）
$f_t(x,y,z) = x \oplus y \oplus z$　（t = 20, ・・・, 39）
$f_t(x,y,z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z)$　（t = 40, ・・・, 59）
$f_t(x,y,z) = x \oplus y \oplus z$　（t = 60, ・・・, 79）

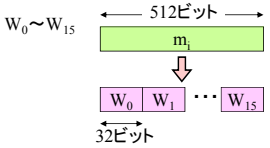
x, y, z, $f_t(x,y,z)$ はそれぞれ32ビット長

10

SHA-1のアルゴリズム

ステップ1）m_i に対し、W_0～W_{79}を生成

W_0～W_{15}

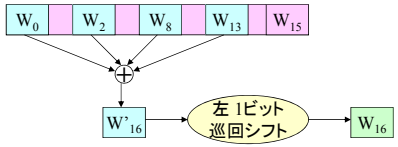


512＝32×16

W_{16}～W_{79}

$Wt = (W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3}) <<< 1$　（t = 16, ・・・, 79）

X <<< n：Xに対する左 n ビット巡回シフト



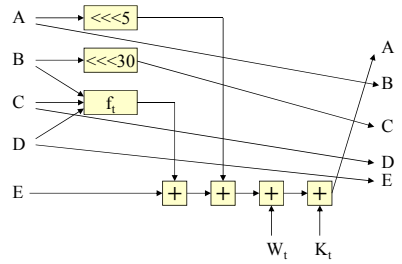
11

SHA-1のアルゴリズム

ステップ3）t = 0 から79まで、以下を繰り返す

$Temp \leftarrow (A<<<5) + f_t(B,C,D) + E + W_t + K_t$
$E \leftarrow D$
$D \leftarrow C$
$C \leftarrow B<<<30$
$B \leftarrow A$
$A \leftarrow Temp$

+ は 加算で、桁溢れは無視



参考：現代暗号、岡本龍明、山本博資、産業図書、1997.6.30、pp.192-195

12