

ハッシュ関数の構成法に関する一考察

AMSL⁺ 井上徹、 Toru INOUE

⁺ (高度移動通信セキュリティ技術研究所)

現在使われているハッシュ関数にはSHA, SHA-1, MD4, MD5などがある。これらのハッシュ関数用の特定のアルゴリズムを用いる方式には演算スピードはそれなりに高速であるが、いくつかの攻撃（攻撃）が発表されている方式もある。^{1) 2) 3)} 一方、ブロック暗号を圧縮関数として繰り返し用いてハッシュ関数を得る方法に Davies-Meyer の方法がある。^{4) 5)} 安全性改善策として2次元 Cellular Automaton を使う方法も報告されている。⁶⁾ また種々の解析結果も報告されている。⁷⁾ しかし一般の64ビットブロック暗号をそのまま用いるとバースデー攻撃により平均 2^{32} の試行で衝突 (collision) が生じてしまい、これは今日では安全性が高いとは言えない。ブロック暗号をハッシュ関数に用いる利点としてブロック暗号の安全性が保証されていればそれを用いたハッシュ関数も安全性が保証されると言われている。^{8) 9)} また、単一のハッシュモードを多重の (multiple) ハッシュモードに拡張しても安全性は保たれる。⁹⁾ さらにデータの暗号化部で使う暗号アルゴリズムのハードウェアまたはソフトウェア資源が共用で

きるメリットがある。本稿では Knudsen らの方法を大幅に改良し、効率の良いハッシュ値を得る方法を提案する。

1. はじめに

本稿では安全性が保証されたブロック暗号を用いることとし、以下の仮定をおくことにする。

- 1) 単一のハッシュモードは安全 (secure) である。
- 2) 単一のハッシュモードを解読 (break) する短絡 (short cut) が存在しない。
- 3) ブロック暗号の安全性が保証されていればそれを用いたハッシュ関数も安全性が保証される。
- 4) 文書は十分に長いとする。

しかしながら、一般の 64 ビット暗号をそのまま用いたのでは衝突 (バースデーアタックまたは collision) が平均 2^{32} 回の試行で生じ、これは今日の暗号事情では決して安全性が高いとはいえない。⁹⁾

誤り訂正符号を用いて衝突耐性を向上させる手法に Knudsen L. and Preneel B. の手法がある。^{8) 9)} 彼らの手法は 4 元 (n, k, d) 線形符号を用いて衝突を $2^{m/2}$ から $2^{(d-1)m/2}$ に改善している。しかし彼らは非 2 元 (non-binary) 符号しか構成に用いておらず、安全性の面でもハードウェア構成の面でも改良の余地がある。筆者は BCH 符号のような 2 元符号を用いた構成法を試み、その有用性を示す。更に畳み込み符号を用いた手法も考察し、ブロック符号と同じ性能条件で設計するとき、拘束長 N の畳み込み符号

はブロック符号を用いた時の $1/N$ の数の Davies-Meyer 関数で装置化できることを示す。

2. ハッシュ関数

任意の長さの文書のある決められた長さに圧縮するためには暗号論的ハッシュ関数がもちいられる。衝突とは $x \neq x'$ なる場合に $h(x) = h(x')$ となることである。衝突に対するハッシュ関数の耐性の定義には、弱い耐性と強い耐性がある。弱い耐性 (Weak collision resist) とはプレイメージ耐性 (pre-image resistance) あるいはセカンドプレイメージ耐性 (2nd-preimage resistance) と呼ばれるものである。

プレイメージ耐性とは与えられたハッシュ値 $H = h(x)$ に対して $h(x')$ なる x' を見つけることがどのくらい困難かということである。

セカンドプレイメージ耐性とは与えられた文書入力 x に対してハッシュ値 $H = h(x) = h(x')$ を持つ第2の文書入力 x' を見つけることがどの程度困難かということである。つまり、文書 x とそのハッシュ値 $H = h(x)$ があつたとき、同じハッシュ値 $H = h(x)$ になる別の文書 x' を見つけることがいかなる文書 x に対しても困難であり、平均 W 回の試行を必要とすればそのハッシュ関数 h のセカンドプレイメージ耐性は W であるという。本稿では以後プレイメージとセカンドプレイメージとは数値として同じであるので同等に扱い区別しない。

強い耐性 (strong collision resist) とは、衝突耐性 (collision resistance) またはバースデーアタックに対する耐性と呼ばれるものである。 $h(x) =$

$h(x')$ となるいかなる文書入力ペア $(x, x'; x \neq x')$ を見つけることがどの程度困難かということである。すなわち、ハッシュ値が同じになる異なる文書の組を見つけるために、平均 W 回の試行を必要とするならば、そのハッシュ関数のバースデーアタックに対する耐性は W であるという。通常単に衝突耐性と言うときはこの強い耐性を言う。

m -ビットブロック暗号 (DESなどは64-ビットブロック暗号である) に基づくハッシュ関数のハッシュレート (hash rate) とは、1回の暗号化または復号化で処理される m -ビットメッセージブロックの数で定義される。

$E_K(\cdot)$ を m -ビットブロック暗号の暗号アルゴリズムとし、その m -ビット鍵を K とする。FEALのように64ビットブロック暗号で64ビット鍵を用いる暗号が該当する。圧縮関数 h を Davies-Meyer 関数と呼ぶ。

『仮定1: 関数 h に対するコリジョン (collision) を見つけるには安全なブロック暗号を用いている限り (m -ビットブロックの) 約 $2^{m/2}$ 回の暗号化が、又関数 h に対するプレイメージ (preimage) を見つけるには約 2^m 回の暗号化が必要である。』^{8) 9)}

メッセージ M_i 、ハッシュ (hash) 値 H_i およびひとつ前の (previous) ハッシュ値 H_{i-1} の間には

$$H_i = h(M_i, H_{i-1}) = E_{M_i}(H_{i-1}) + H_{i-1}$$

が成り立つ。ここで $+$ は法2加算である。 H_i は文書の始まりから時刻 $i-1$ までのハッシュ値と時刻 i のメッセージとの累積加算値である。

$E_{K'}(\cdot)$ を、 $a > 0$ なる $a \times m$ -ビット鍵 K を使う m -ビットブロック暗

号アルゴリズムとする。 h_1, h_2, \dots, h_n を鍵を互いに異なる値を取ることにより互いに異なる Davies-Meyer 関数にする。多重デヴィスマイヤー (multiple Davies-Meyer) 関数は r 個の m -ビットメッセージ入力をアフィン変換して n 個のペア (X_i, Y_i) へ写像して入力する。出力は H_1, H_2, \dots, H_n の接続となる。コリジョン (collision) またはプレイメージ (preimage) アタックにおいて入力ブロックを構成する (X_i, Y_i) が元のペアと異なるなら $h(X_i, Y_i)$ は積極的 (active) と呼ばれる。又、2つの関数 $h_i(X_i, Y_i)$ と $h_j(X_j, Y_j)$ は独立に (independently) 攻撃可能という時は、関数 h_i の変数パラメータ (X_i, Y_i) が変わっても、関数 h_j の変数パラメータ (X_j, Y_j) は変わらないような関数をいう。

仮定 2 : 多重デヴィスマイヤー (multiple Davies-Meyer) の圧縮関数に対するコリジョン (collision) あるいはプレイメージ (preimage) が見つかったとする。 P を active な関数の数とし、 $P - v$ を独立に攻撃可能な関数の最大数とする。このコリジョン (collision) あるいはプレイメージ (preimage) が起こるためには少なくとも $2^{vm/2}$ 、あるいは 2^{vm} 回の暗号化がそれぞれ必要である。』^{8) 9)}

3. 誤り訂正符号を用いた構成法

3.1. Knudsen L, and Preneel B による構成法

Knudsen L, and Preneel B による誤り訂正符号を用いた構成法を紹介しよう。

[定理 3] : 長さ n 、情報シンボル数 k 、最小距離 d 、 $(a + 1)k > n$ 但

し $a \geq 1$ 且つ $m \gg \log_2 n$ なる $GF(2^{a+1})$ 上の (n, k, d) 符号で入力ブロックを符号化したとする。すると仮定 2 が成り立つ限り圧縮関数に対するコリジョン (collision) を見つけることは少なくとも $2^{(d-1)m/2}$ 回の暗号化が、あるいはプレイメージ (preimage) を見つけるには少なくとも $2^{(d-1)m}$ 回の暗号化が必要である。9)』

(証明): 入力は $(a+1)k$ 個の m ビットブロックからなる。 n 個の連鎖値 H_{i-1}^1 から H_{i-1}^n までと r 個のメッセージブロック M^1 から M^r までのブロックからなる。ここで $r = 2k - n > 0$ である。入力ブロックの $(a+1)$ 個の引き続くビットは $GF(2^{a+1})$ の k 個の要素を与えるように構成される。

これら要素は (n, k, d) 符号を使い $GF(2^{a+1})$ の n 個の元となる。 n 個の関数の 1 つに対して $(a+1)$ ビットの入力を表す。個々のビットは $GF(2^{a+1})$ の上のベクトル空間として $GF(2^{a+1})$ の元を表している。

この構成は $v = d - 1$ なる値に対して仮定 2 に対する条件が満足される。

$(a+1)k$ 個の入力の線形変換によって圧縮関数を得ることが可能となる。ここで最初の k 個の関数がお互いに独立である。符号語は最後の $n - k$ 個の関数の少なくとも $d - 1$ 個が最初の関数の入力に依存している。 $n - k \geq d - 1$ だから定理 3 はおのずと証明される。(証明終)

この hash 関数は $n \cdot m$ ビットの内部メモリーが必要で hash rate は $(a+1)(k/n) - 1$ である。すなわち、例えば距離 3 の誤り訂正符号を使う

ことによってコリジョン (collision) を $2^{m/2}$ から 2^m に、あるいはプレイメージ (preimage) を 2^m から 2^{2m} に改善し、安全な hash 関数を簡単に構成できる。

3.2. ブロック符号による構成法

また新たに BCH 符号などの 2 元符号を用いる方式¹⁰⁾ が提案され、1 つの m ビット入力に 2 元符号 (binary code) の 2 つの符号語を割り当てる構成にして 2 元符号を使えるような構成にしている。

4. 提案方式

前項 (3. 2. 項) の改善策はガロア体の符号を 2 元の符号に置き換えてはいるものの、初期値を除くと 1 単位時間前のハッシュ値だけで構成される n 個の m -ブロックから構成されている 1 列目の BCH 符号語は前の時刻のハッシュ値だけを符号化する事になり、ここには新しいメッセージの情報は入力されない。従って、この列が攻撃 (アタック) されることはなく、攻撃に対する符号化をする必要もない。符号化は 2 列目の n 個の m -ブロックだけを行い攻撃から守ればよい。ベクトル X_i は暗号器が完全 (乱数オラクルとみなせる) であれば乱数と見做せる。関数 h_i の Y_i だけ誤り訂正符号化する。

このように構成しても圧縮関数に対するコリジョン (collision) を見つけることは少なくとも $2^{(d-1)m/2}$ 回の暗号化が、あるいはプレイメージ (preimage) を見つけるには少なくとも $2^{(d-1)m}$ 回の暗号化が必要である。

4.1. ブロック符号を用いた方法

〔定理 4〕: k 個のメッセージブロックを ECC 符号化して n 個の m -ビットブロックを作り、それを n 個の previous hashed values と並べて、 $n \times 2$ 個の m -ビットブロックにして n -multiple Davies-Meyer 関数に入力するとする。 n 個のブロックに collision または preimage が起こったとし P 個の Davies-Meyer 関数が積極的ならば $P - (d - 1)$ 個のメッセージ入力ベクトル Y_i が独立で $d - 1$ 個のメッセージ入力ベクトル Y_i が従属である。』

(証明): ベクトル Y_i の n ビットは少なくとも最初の k ビットが互いに独立である。最後の $n - k$ ビットの少なくとも $d - 1$ 個が最初の入力に依存している。 $n - k \geq d - 1$ だから定理 4 はおのずと証明される。(証明終)

以上の議論は定常状態で入力ベクトル X_i が乱数オラクルの出力とみなせる限り成り立つ。しかし、初期状態においては鍵入力、メッセージ入力ともに任意の値を取りうる。従って、初期入力ベクトルについては Knudsen, L. and Preneel, B. に基づいて入力せねばならない。

〔仮定 2 の拡大 (extension)〕: 初期状態: Multiple-Davies-Meyer の圧縮関数に対するコリジョンあるいはプレイメージが見つかったとする。 P を active な関数の数、 $P - v$ を独立に攻撃可能な関数の数とする。 f_1, \dots, f_n を同時に衝突が起きている Davies-Meyer 関数とする。個々の関数には

$$f_i(X_i, Y_i) = E_{X_i}(Y_i) + Y_i$$

が成り立つ。これらは異なる値を「 $\log_2 n$ 」個の鍵を固定して得られる。 2^k 個の m ビット初期値入力をもつ圧縮関数を考えそれがアフィン変換により n 個のペア (X_i, Y_i) に写像されるすべての圧縮関数の出力が 2^k 個のブロックに依存する。すなわち変換マトリクスは階数 2^k を持つ。

定常状態： k 個の m ビット入力を持つ初期値状態を除く定常状態において k 個の m ビット入力を持つ圧縮関数のアフィン変換により n 個のベクトル Y_i に写像されるすべての圧縮関数の出力が k 個のブロックに依存する。すなわち変換マトリクスは階数 k を持つ。

同時衝突が f_1, \dots, f_n に起こった時、すなわち 2 つの入力値 $\{Z_i\}$ と $\{Z_i'\}$ の異なった組がすべての n 個の関数に等しい出力を与えるとする。この時 $P \leq n$ で $Z_i \neq Z_i'$ に依存する P 個の関数の組を $\{f_j\}$ とする。関数 $\{f_j\}$ のマトリクスの階数は $P - v$ となる。これら P 個の関数 $\{f_j\}$ に対する同時衝突は少なくとも $2^{v \cdot m} / 2$ 回の暗号化が、また、pre-image に対しては $2^{v \cdot m}$ 回の暗号化が必要である。従って鍵入力 m ビットのうち k ビットを予め誤り訂正符号で符号化し n ビットにして初期値 X_0 を与え、残り $n - k$ ビットを初期値 Y_0 のうち $n - k$ ビットに配置する。すなわち初期状態のメッセージビットは $r = 2^k - n$ ビットとなる。2 番目以降のメッセージビットは $r = k$ ビットとなる。

すなわち $d - 1$ 個の Y_i ベクトルだけが従属であることが要求されるので定常状態においてはベクトル Y_j だけを誤り訂正符号化すればよい。

[補題 5]：符号長 n 、情報記号数 k 、最小距離 d の 2 元符号を考える。デ

ビスマイヤー (Davies-Meyer) の段数を n とする。時刻を i と仮定すると前の時刻 $i - 1$ の n 個のハッシュ値 (previous hash value) はすべて 1 列目の n 個の m -ブロックに戻されるから、メッセージブロックのため使われるブロック数は k となる。Hash rate (ハッシュ率) は k / n となる。』

これは元々誤り訂正符号の持っている符号化率に等しい。すなわち効率が大幅に改善されることになる。運用上の注意としては初期値に $n \times m$ ビットの鍵の初期値が必要である。

新しい構成法は 2 元符号の 1 つの符号語によって n 個のペアー入力の片側である m ビットブロックに入力される。

2 元符号を使った例と Knudsen らの結果とを比較のため表でしめす。

表 1. 2 元の符号による方法と従来法との比較

field	t	code	rate	collision	memory
GF(2)	1	(7, 4, 3)	4/7=0.571	2^m	7m
GF(2)	1	(15, 11, 3)	11/15=0.733	2^m	15m
GF(2)	2	(15, 7, 5)	7/15=0.467	2^{2m}	15m
GF(2)	2	(25, 15, 5)	15/25 =0.60	2^{2m}	25m
GF(2)	2	(30, 20, 5)	20/30 =0.666	2^{2m}	30m
	2	(62, 54, 5)	54/62 =0.871	2^{2m}	62m
GF(2 ²)	1	(5, 3, 3)	1/5=0.20	2^m	5m
GF(2 ⁴)	1	(6, 4, 3)	1/4=0.25	2^m	6m
GF(2 ²)	1	(8, 5, 3)	1/4=0.25	2^m	8m

Non-binary の符号は Knudsen L. and Preneel B. ⁸⁾ ⁹⁾ らの結果による。

この方法は Knudsen L. and Preneel B. と比べてフィードバックされたハッシュ値を誤り訂正符号化しない分だけ効率 (hash rate) が改善されており、

BCH 符号などの 2 元符号を使うため既存の高速ハードウェアなどが流用で

きる。更に collision を $m = 64$ として $2^{2m} = 2^{128}$ に一回の割合に抑える要望があれば $d = 5$ の BCH 符号を選べば良い。

以上の説明からわかるように本稿によれば 2 元符号を使うため、ガロア体の演算をしなくて済み、且つ、より安全性の高いハッシュ値を得ることができる。

4.2. 畳み込み符号を用いた構成法¹¹⁾

ブロック符号では誤り訂正能力を大きくするには符号長が一般に大きくなり演算が複雑になる傾向があった。又、符号長分だけのデビスマイヤーの段数が必要になるため多くのデビスマイヤー関数を装置化しておく必要がある。

本稿ではデビスマイヤー (Davies-Meyer) の基本構成である段数入力を畳み込み符号の小ブロック長 n_0 に選ぶことにより又、拘束長 N とするとき N 段に分けて入力することによりデビスマイヤーのサイズをブロック符号を使った時の符号長 n から畳み込み符号の小ブロック長 n_0 まで小さくできる構成法を示す。すなわち、デビスマイヤー (Davies-Meyer) 関数の基本構成の段数は $1/N$ まで減少する。

構成は畳み込み符号器と多段デビスマイヤーハッシュ関数と N 段に接続されたハッシュ値を蓄積する FIFO メモリーによって特徴付けられる。畳み込み符号は大きく分けて 2 種類の符号器がある。例で説明しよう。畳み込み符号の小ブロック長 n_0 ビット、小情報記号ビット k_0 、拘束長 N 段とするとき (n_0, k_0, N) 畳み込み符号と呼ぶ。定理 4 を畳み込み符号にも適用しよう。

【定理6】： n_0 個の previous hashed values と k_0 個のメッセージブロックを畳み込み符号化して n_0 個の m -ビットブロックを作り $n_0 \times 2$ 個の m -ビットブロックを n_0 -multiple Davies-Meyer 関数に入力するとする。拘束長 N とするとき $N \times n_0$ 個のブロックに collision または preimage が起こったとし P 個の Davies-Meyer 関数が積極的ならば $P - d + 1$ 個の入力ベクトルが独立で $d - 1$ 個の入力ベクトルが従属である。』

運用上の注意としては $n_0 \times m$ ビットの鍵初期値が必要である。

畳込み符号を用いると多重デビスマイヤー関数のサイズは $1/N$ 倍と小さくなるが、拘束長 N 、全長 $N \times n_0 \times m$ ビットのハッシュ値を得るためにはデータの最後に $(N - 1) \times k_0 \times m$ ビットの0を入力してデータを完全に符号器から出力させる必要がある。したがって符号化のため $(N - 1) \times m$ ビットの遅延が起こる。

提案方式による一般的なハッシュ関数は多重デビスマイヤー関数の入力側で $n_0 \times 2$ の2次元配置を構成して第1の列の n_0 の m -ビットブロックに1単位時刻前の n_0 個のハッシュ値をフィードバックし新しいメッセージ k_0 個の m -ビットブロックを符号化して n_0 個の m -ビットブロックにしたのち、入力する方式となる。

ワイナーアッシュ符号¹³⁾を使った場合とCSOC符号(Convolutional Self-Orthogonal Codes; 自己直交畳込み符号)¹³⁾を使った例を表2, 3に示す。

表2 ワイナーアッシュ (Wyner-Ash) 符号による構成例

符号 (W-S)	N	Hash rate	collision	Memory(Hash)
(4, 3, 3)	2	3/4	2^m	8 m
(8, 7, 3)	2	7/8	2^m	16 m

表3 CSOC符号による構成例

符号 (CSOC)	N	Hash rate	collision	Memory(Hash)
(3, 2, 3)	3	2/3	2^m	9 m
(3, 2, 5)	14	2/3	2^{2m}	42 m

5. まとめ

誤り訂正符号を用いてハッシュ関数を構成する Knudsen L. and Preneel B. を改良して

2元の符号を用いる方法、

畳み込み符号を用いる方法

を開発し、その構成を示した。近年ブロック暗号の鍵長の安全性が真剣に議論されるようになり、128ビット以上の鍵を持つ暗号が主流となりつつある。¹⁴⁾このような要請と合わせ既存の手法を組み合わせて安全なハッシュ関数を求める研究がもっと活発になされるべきである。

参考文献

- 1) Dobeberlin H., "Cryptanalysis of MD4, : Fast Software Encryption, LNCS 1039, D. Goldman, Ed., Springer-Verlag, 1996, pp.53-69.
- 2) Dobeberlin H., :Cryptanalysis of MD5 compress, Presented at the rump session

of EUROCRYPT'96, May 1996.

- 3) Kuwakado Hidenori and Tanaka Hatsukazu, "On the One-wayness of the reduced MD4 compression function", The 1999 Symposium on Cryptography and Information Security", pp.247-250. Kobe, Japan, January 26-29, 1999.
- 4) Davies D.W. and Price W.L.,: Digital Signature-An Update,: Proceedings of International Conference on Computer Communications, Sydney, Oct 1984, North Holland: Elsevier, 1985, pp.843-847.
- 5) Matyas S.M., Meyer C. H. and Oseas J.,: Generating Strong One-Way Functions with Cryptographic Algorithm, IBM Technical Disclosure Bulletin, v. 27, n. 10A, Mar. 1985, pp.5658-5659.
- 6) Hirose Shouichi and Yoshida Susumu, "A one-way hash function based on a two-dimensional cellular automaton", The 20-th Symposium on Information Theory and Its Applications (SITA97) pp.213-216, Matsuyama, Japan, December 2-5, 1997.(In Japanese)
- 7) Morita Hikaru, Odagi Hideo and Ohta Kazuo, "Collision search of a hash function by using random mapping", IEICE Trans. Fundamentals, vol. E81-A, No.1, pp.35-40, January 1998.
- 8) Knudsen L.R., and Preneel B.,: Hash functions based on block ciphers and quaternary codes, Advances in Cryptology, Proc. Asiacrypto'96, LNCS 1163, K. Kim, T. Matsumoto, Eds., Springer-Verlag, 1996, pp.77-90
- 9) Knudsen L. and Preneel B. "Fast and secure hashing based on codes,

Crypto'97, LNCS1294, pp.485-498, 1997.

10)INOUE Toru and TANAKA Hatsukazu, :A Note on Hash Functions Based on Block Ciphers and Error Correction Code, International Symposium on Information Theory and its Applications, pp.127-129., Mexico City MEXICO, October 14-16,1998

11)井上徹、“畳み込み符号を用いたハッシュ関数の一構成法”、信学技報 I T 9 8 - 5 5 , pp.1-6,(1999-01)。

12)Lin, S,: An Introduction to Error Correction Codes, Englewood Cliffs., N., J., 1970,Chapter 10.

13)川村信一：A E S 参加報告、信学会技術報告、ISEC98-41,pp.39-43, November 1998.