SHA-1に基づく出力長何変な暗号ハッシュ関数の設計

許　容碩　　　櫻井　幸一
九州大学 システム情報科学研究院 〒812-8581 福岡市東区箱崎 6-10-1
E-mail: {ysher, sakurai}@tcslab.csce.kyushu-u.ac.jp

A Design of Cryptographic Hash Function Group with Variable Output-Length Based on SHA-1

Yong-Sork HER　　Kouichi SAKURAI
Dept. of Computer Science and Communication Engineering, Kyushu University
6-10-1 Hakozaki, Higashi-ku, Fukuoka, 812-8581 Japan
E-mail: {ysher, sakurai}@tcslab.csce.kyushu-u.ac.jp

Abstract The cryptographic hash function provides the services of information security, authentication, integrity, non-reputation in a branch of information secret. A cryptographic hash function has been developed since MD4 was proposed by Rivest. In present, U.S standard of a hash function is SHA-1 with 160 bits of output length. It is difficult to be sure of a security of a hash function with 160 bits of output length. In this paper, we propose a hash function, namely SHA-V, with variable output-length based on SHA-1, HAVAL and HAS-V. The structure of SHA-V is two parallel lines, denoted as the Left-line and Right-line, consisting of 80 steps each and 3-variable 4 Boolean functions each line. The input length is 1024 bits and the output length is from 128 bits to 320 bits by 32-bit. SHA-V has the most advantage of SHA-1. That is, the message variable creates in combination with input message and step calculations. This new message variable provides the resistance against most of attacks that search the collision resistance by the fabricating of input messages. When we compare SHA-V and HAS-V in side of operation, SHA-V is 10% faster than HAS-V on a Pentium PC.

Keyword Cryptographic hash functions, Cryptography, SHA-1, Collision resistance, MD-4, HAVAL

1. Introduction

Hash functions[2] take a message as input with an arbitrary and produce an output referred to as a hash-code, hash-result, hash-value, or simply hash. In 1990, Rivest proposed the cryptographic hash function MD4[18]. MD4 is a 128-bit hash function and consists of 3 rounds. Den Boer, Bosselaers[4] and Dobbertin [8] proposed an attack method on the each round of MD4. So, it is undesirable to use MD4. But, There are hash functions, MD-5[19], SHA-1[15] and RIPEMD-160[7], that is based on the MD4. We call the customized hash functions based on MD4. It was not known the attack method about these hash functions. As the development of computer technologies, the ability of computer processing has been increased.

As the growth of computer technologies, the hash value has been become longer based on the complexity of calculation. It was known to be desirable that the output lengths of hash functions are more than 160 bits.

The 1st edition of Hash Function Standard-160 of Korea, namely HAS-160[20], was published in 1998 and the revised edition was published in 2000. P.J.Lee *et al.* proposed HAS-V[16] with variable output length based on HAS-160[20] and HAVAL[22].

The important properties that a cryptographic hash function must satisfy are the following [2].

a) Preimage Resistance: it is computationally infeasible to find any input which hashes to that output, i.e., to find any preimage x' such that $h(x')=y$ when given any y for which a corresponding input is not known.

b) Second Preimage Resistance: it is computationally infeasible to find any second input which has the same output as any specified input, i.e., given x, to find a 2nd-preimage $x' \neq x$ such that *h(x)=h(x').*

*c) **Collision Resistance**:* it is computationally infeasible to find any two distinct inputs *x, x'* which hash to the same input, i.e., such that *h(x)=h(x')*.
The attack method of hash functions based on the properties of a cryptographic hash function was introduced by several cryptography-researcher. We introduce cryptographic hash functions in Section 2 and propose SHA-V in Section 3. In Section 4, we compare with HAS-V and we analyze the security against Kuwakado and Tanaka's attack method[9] in section 5. In section 6, we compare to SHA-1 / 256 and SHA-V(160bits, 256bits) and analyze the parallel implementation of SHA-V and HAS-V in Section 7.

2. Introduction of Cryptographic Hash Functions

In this section, we introduce to the cryptographic hash function. <Table 1>[10] shows features of cryptographic hash functions.

2.1. SHA-1 / SHA-256, 384, 512

SHA-1 (Secure Hash Algorithm)[15], based on MD4, was proposed by the NIST (U.S. National Institute for Standards and Technology) for certain federal government applications [2]. The hash-value is 160 bits and five 32-bit chaining variables are used. SHA-1 uses four non-zero additive constants, whereas MD4 used three constants only two of which were non-zero.

SHA-256, 384 and 512 [14] are the longer version of the output length based on SHA-1, MD4 and MD5.

2.2. HAS-160

HAS-160 (Hash Algorithm Standard)[20] provides the methods to compress bit strings with arbitrary lengths into a hash code with fixed lengths (160 bits). This hash algorithm inputs messages of the random length with block unit of 512 bits. The length of output is 160 bits and the compression function deals with block of 512-bit unit. It is composed of 4 rounds, 80 steps and 5 words of chaining variables. The number of variable message to apply in each step is 20 words. If it inputs the message *M* of the random length in this hash algorithm, *M* is made up of the multiple of 512-bit through attach processing and divided into the block M_i of 512 bits.

2.3. HAS-V

HAS-V[16] was proposed to meet the needs of various security levels desired among different applications. The length of the hash-code is an important factor directly connected to the security of the hash function. KCDSA (Korea Certificate-based Digital Signature Algorithm) is an example of a cryptographic application where a variable length of hash-code is needed. There exists an optional extension of RIPEMD-128 and RIPEMD-160 to produce 256-bit and 320-bit hash-code. However, these methods do not provide any increase in security level, but merely an increase in the length of the hash-code. This gives a clear motivation to design a new hash function with variable length hash-code, which is both efficient and secure.

<Table 1> Features of cryptographic hash functions

Algorithm	**Endi-aness**	**Message Block Size**	**Digest Size (bits)**	**Word Size (bits)**	**The Number of steps**
MD5	Little	512	128	32	64
RIPEMD-128	Little	512	128	32	2 X 64
RIPEMD-160	Little	512	160	32	2 X 80
SHA-1	Big	512	160	32	80
SHA-256	Big	512	256	32	64
SHA-512	Big	1024	512	64	80
HAS-V	**Little**	**1024**	**128-320**	**64**	**100**
SHA-V	**Big**	**1024**	**128-320**	**64**	**80**

3. SHA-V

We propose SHA-V based on SHA-1, HAVAL and SHA-V. The structure of SHA-V is two parallel lines, denoted as the Left-line and Right-line, consisting of 80 steps each. The input length is 1024 bits and the output length is from 128-bit to 320 bits by 32-bit. SHA-V has the most advantage of SHA-1. That is, the new message variable creates in combination with input message and step calculations. This new message variable provides the resistance against most of attacks that search the collision resistance by the fabricating of input messages.

3.1. Initial Values

The initial values of the chaining are used in SHA-V are given in follow Table 2. It is divided two parallel lines: left-line and right-line. The initial values of left line, $H_0^{(i)}$, $H_1^{(i)}$, $H_2^{(i)}$, $H_3^{(i)}$, $H_4^{(i)}$, are the same SHA-1. The initial values of right line are based on HAS -V[16]. In order to increase the output length, SHA-V has ten 32-bit words.

<Table 2> Initial values of SHA-V

Left-line				
$H_0^{(i)}$	$H_1^{(i)}$	$H_2^{(i)}$	$H_3^{(i)}$	$H_4^{(i)}$
67452301	efcdab89	98badcfe	10325476	c3d2e1f0
Right-line				
$H_5^{(i)}$	$H_6^{(i)}$	$H_7^{(i)}$	$H_8^{(i)}$	$H_9^{(i)}$
8796a5b4	4b5a6978	0f1e2d3c	a0b1c2d3	68794e5f

3.2. Constants

The constants are taken as the integer parts of the number given in <Table 3>. The constants of left-line are based on SHA-1 except K[3]. K[3] is Oxca62c1d6 $(2^{30}\sqrt{10})$ in SHA-1, but K[3] of SHA-V uses the Oxa953fd4e $(2^{30}\sqrt{7})$. The constants of right-line are based on HAS-V, but the constants sequence is different with HAS-V. Also, the constants of HAS-V consist of five 32 bits for 5 rounds. SHA-V consists of 4 round and the constants of four 32 bits. SHA-V can be decreased the computation number than HAS-V.

<Table 3> Constants of SHA-V

Left-line			
K[0]	**K[1]**	**K[2]**	**K[3]**
5a837999 $(2^{30}\sqrt{2})$	6ed9eba1 $(2^{30}\sqrt{3})$	8f1bbcdc $(2^{30}\sqrt{5})$	a953fd4e $(2^{30}\sqrt{7})$
Right-line			
K'[0]	**K'[1]**	**K'[2]**	**K'[3]**
7a6d76e9 $(2^{30}\sqrt[3]{7})$	6d703ef3 $(2^{30}\sqrt[3]{5})$	5c4dd124 $(2^{30}\sqrt[3]{3})$	50a28be6 $(2^{30}\sqrt[3]{2})$

3.3. Boolean functions

The Boolean functions to be used each step operations are as follows. It consists of two parallel lines. The Boolean functions of left-line are based on SHA-1 and the Boolean functions of right-line are the reverse order with left-line.

- Left

$f_t(x,y,z) = (x \wedge y) \vee (\sim x \wedge z) \quad (0 \leq t \leq 19)$

$f_t(x,y,z) = x \oplus y \oplus z \quad (20 \leq t \leq 39)$

$f_t(x,y,z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z) \quad (40 \leq t \leq 59)$

$f_t(x,y,z) = x \oplus y \oplus z \quad (60 \leq t \leq 79)$

- Right

$g_t(x,y,z) = x \oplus y \oplus z \quad (0 \leq t \leq 19)$

$g_t(x,y,z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z) \quad (20 \leq t \leq 39)$

$g_t(x,y,z) = x \oplus y \oplus z \quad (40 \leq t \leq 59)$

$g_t(x,y,z) = (x \wedge y) \vee (\sim x \wedge z) \quad (60 \leq t \leq 79)$

3.4. Append Padding Bits

The message words to be used in the compression function are 32 words, or a 1024 bits block, of the input message. The padding bits of SHA-V are based on SHA-1 and HAS-V. The message is padded so that its length is congruent to 952 modulo 1024. Padding is performed by appending a single "1"bit and necessary "0"bits to satisfy the above constraints. The remaining 72 bits, in order to be a multiple of 1024bits, are filled by appending the desired length of the hash-code represented in bytes.

3.5. Output Tailoring

In order to creates the output length with variables, the hash-code is computed as <Table 4>. These methods are based on PMD-V[14]. The output length is 128 + 32 (t-3).

The character of SHA-V is fairly to be computed in each line. For the computation speed, it uses only shift and addition operation. The output length is from 128 bits to 320 bits. For example, in order to create the 128-bit output, it adds to the each two by two in 10 chaining variables.

In the case of 320-bit hash-code, the output length is the same is given as the contents of the 10-chaining variable concatenated, i.e. $H_0^{(i)} \| H_1^{(i)} \| H_2^{(i)} \| H_3^{(i)} \| H_4^{(i)} \| H_5^{(i)} \| H_6^{(i)} \| H_7^{(i)} \| H_8^{(i)} \| H_9^{(i)}$.

4. Estimation of HAS-V and SHA-V

In this paper, we analyze HAS-V[16] and SHA-V through the step computations of two papers. <Figure 1> shows the structure of the proposal SHA-V.

4.1. Endianess

The basic structures of the compression functions of HAS-V and SHA-V are two parallel lines and an input messages are 1024 bits, too. HAS-V favors ' little-endian' architectures such like MD4-family whereas SHA-V favors 'big-endian' architectures such like SHA-family. In the result of implementation, the both methods unify for fairness. Because, Little-endian

<Table 4> The calculation methods of a variable output-length

output length	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	O_9
128-bit	$a+v^{\ll 2}$	$b+v^{\ll 3}$				$f+w^{\ll 2}$	$g+w^{\ll 3}$			
160-bit	a+f	b+g	c+h	d+i	e+j					
192-bit	$a+j^{\ll 2}$	$b+j^{\ll 3}$	$c+j^{\ll 5}$			$f+j^{\ll 2}$	$g+j^{\ll 3}$	$h+j^{\ll 5}$		
224-bit	$a+j^{\ll 2}$	$b+j^{\ll 3}$	$c+j^{\ll 5}$	$d+j^{\ll 7}$	$e+j^{\ll 11}$	$f+j^{\ll 13}$	$g+j^{\ll 17}$			
256-bit	$a+j^{\ll 2}$	$b+j^{\ll 3}$	$c+j^{\ll 5}$	$d+j^{\ll 7}$		$f+e^{\ll 2}$	$g+e^{\ll 3}$	$h+e^{\ll 5}$	$i+e^{\ll 7}$	
288-bit	$a+j^{\ll 2}$	$b+j^{\ll 3}$	$c+j^{\ll 5}$	$d+j^{\ll 7}$	$e+j^{\ll 11}$	$f+j^{\ll 13}$	$g+j^{\ll 17}$	$h+j^{\ll 19}$	$i+j^{\ll 23}$	
320-bit	a	b	c	d	e	f	g	h	i	j

($z=f\oplus g\oplus h\oplus i\oplus j$, $v=c\oplus d\oplus e$, $w=h\oplus i\oplus j$)

architecture is suitable in the Pentium processors and Bit-endian architecture is suitable in the SUN system.

4.2. Boolean functions

When we compare SHA-V and HAS-V, it is the same in the critical path length (CPL) (<<<, f(), +). <Table 5>[1] shows the CPL of a step for each hash functions.

<Table 5> Lower bound on the CPL of a step

Algorithm	Operations in CPL	min.CPL
MD4,RIPEMD,RIPEMD-128	f(), + , <<<	3
MD5, RIPEMD-160	f(), + , <<<, +	4
SHA-1	+, <<<	2

4.3.Step Operations and Comparison of Speed Performance with HAS-V

The step operation of HAS-V consists of 3 additions, 2 circular shifts and a Boolean function. Since the Boolean function consists of 4 unit operations, a single step operation will consist of 9 unit operations, assuming both addition and circular shift to be equivalent to unit operation. The total number of unit operations for generating the extra messages is

2(lines) X 5(rounds)X4(messages)X 3 (unit operations) = 120 (unit operations).

Therefore the number of unit operations to hash 1024 -bit block is given as follows[16].

1 (block) X 200(steps) X 9 (step operations) + 120 (message expansion) = 1920 (unit operations)

In the case of RIPEMD-160, the total number of unit operation to hash 1024 bit block is given below.

2(block) X 160(steps) X 9 (Step operation) = 2880 (unit operations)

This is about 33% more operation than HAS-V. This fact can also be seen in Table.5 where HAS-V is 31% faster than RIPEMD-160 on a Pentium PC[16].

In the case of SHA-V, the step operations of SHA-V consist of 80 steps each line, a single step operation will consist of 10 unit operations, have not the message expansion. Therefore the number of unit operations to hash 1024 bit block is given as follows.

1 (block) X 160(steps) X 10 (step operations) = 1600 (unit operations)

Let assume the speed of HAS-V and SHA-V using a proportional expression. HAS-V is about 17% more operation than SHA-V. SHA-V is 10% faster than HAS-V on a Pentium PC. But, these values are the computed values.

5. The Security of SHA-V against KUWAKADO and TANAKA's Attack Methods

We analyze the security of SHA-V against Kuwakado and Tanaka's attack method[9] of hash functions. In Kuwakado and Tanaka's paper, they proposed an efficient algorithm for finding preimages for the reduced MD4 that consists of the first round and the third round. We apply these attack methods to the SHA-V. We reduce the first, the third and the fifth round of SHA-V.

■ **Algorithm**

- Input : Messages A, B, C, D, and E
- Output : Message words m_0, m_1, ..., m_{19}

① Choose a_0 and e_1 randomly

② Set $b_4 \leftarrow$0xffffffff, $c_3 \leftarrow$0xffffffff, and $d_2 \leftarrow$ 0xffffffff.

③ According to steps 0,1,2,3,and 4 compute m_0, m_1, m_2, m_3 m_4 from a_{-5}, b_{-1}, c_{-2}, d_{-3}, e_{-4}, a_0 , b_4, c_3, d_2, and e_1.

Left-Line

```
Message padding  $M \leftarrow$ SHA-V-PAD
Global  $X_0, \ldots, X_{79}$
$M = M_1 \| M_2 \| M_3 \| \cdots \| M_n$,where each M is a 1024-bit block
$H_0 \leftarrow$ 67452301
$H_1 \leftarrow$ efcdab89
$H_2 \leftarrow$ 98badcfe
$H_3 \leftarrow$ 10325476
$H_4 \leftarrow$ c3d2e1f0
for  $i \leftarrow 1$ to n
$M_1 = W_0 \| W_1 \| W_2 \| \cdots \| W_{15}$,where each $W_i$ is a word
for t $\leftarrow$ 16 to 79
Do $W_t \leftarrow ROTL^1 (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16})$
$A \leftarrow H_0$
$B \leftarrow H_1$
$C \leftarrow H_2$
$D \leftarrow H_3$
$E \leftarrow H_4$
for t $\leftarrow$ 0 to 79
Temp $\leftarrow ROTL^1(A) + f_t(B,C,D) + E + W_t + K_t$
$E \leftarrow D$
$D \leftarrow C$
$C \leftarrow ROTL^{30}(B)$
$B \leftarrow A$
$A \leftarrow$ temp
$H'_0 \leftarrow H_0 + A$
$H'_1 \leftarrow H_1 + B$
$H'_2 \leftarrow H_2 + C$
$H'_3 \leftarrow H_3 + D$
$H'_4 \leftarrow H_4 + E$
Return $(H'_0 \| H'_1 \| H'_2 \| H'_3 \| H'_4)$
```

Right-Line

```
Message padding  $M \leftarrow$ SHA-V-PAD
Global  $X_0, \ldots, X_{79}$
$H_5 \leftarrow$ 8796a5b4
$H_6 \leftarrow$ 4b5a6978
$H_7 \leftarrow$ 0f1e2d3c
$H_8 \leftarrow$ a0b1c2d3
$H_8 \leftarrow$ 68794e5f
for  $i \leftarrow 1$ to n
$M'_1 = W'_0 \| W'_1 \| W'_2 \| \cdots \| W'_{15}$,where each $W'_i$ is a word
for t $\leftarrow$ 16 to 79
Do $W'_t \leftarrow ROTL^1 (W'_{t-3} \oplus W'_{t-8} \oplus W'_{t-14} \oplus W'_{t-16})$
$A' \leftarrow H_5$
$B' \leftarrow H_6$
$C' \leftarrow H_7$
$D' \leftarrow H_8$
$E' \leftarrow H_9$
for t $\leftarrow$ 0 to 79
Temp$\leftarrow ROTL^1(A') + f_t(B',C',D') + E' + W'_t + K_t$
$E' \leftarrow D'$
$D' \leftarrow C'$
$C' \leftarrow ROTL^{30}(B')$
$B' \leftarrow A'$
$A' \leftarrow$ temp
$H'_5 \leftarrow H_5 + A'$
$H'_6 \leftarrow H_6 + B'$
$H'_7 \leftarrow H_7 + C'$
$H'_8 \leftarrow H_8 + D'$
$H'_9 \leftarrow H_9 + E'$
Return $(H'_5 \| H'_6 \| H'_7 \| H'_8 \| H'_9)$
```

<Figure 1> The structure of SHA-V

$m_{18} \leftarrow (a_0 >>5) - (a_{-5} + F(b_{-1}, c_{-2}, d_{-3}, e_{-4}))$

$m_0 \leftarrow (e_1 >>11) - (e_{-4} + F(a_0, b_{-1}, c_{-2}, d_{-3}))$

$m_1 \leftarrow (d_2 >>7) - (d_{-3} + F(e_1, a_0, b_{-1}, c_{-2}))$

$m_2 \leftarrow (c_3 >>13) - (c_{-2} + F(d_2, e_1, a_0, b_{-1}))$

④ After setting a_5 to Ox 00000000, compute m_{19} as

$m_{19} \leftarrow (a_5 >>) - (a_0 \quad F(b_4, c_3, d_2, e_1))$

That is

$m_{19} \leftarrow (a_5 >>) - (a_0 + \neg e_1)$

⑤ m_{19} = ? Considering operations from Step 6 to step 19. We observe that d, c, b ?

⑥ Set $a_{95} \leftarrow A - a_{-5}$, $b_{99} \leftarrow B - b_{-1}$, $c_{98} \leftarrow C - c_{-2}$, $d_{97} \leftarrow D - d_{-3}$, $e_{99} \leftarrow E - e_{-4}$

⑦ By executing from step 99 to step 84 in reverse order, compute a_{80}, b_{84}, c_{83}, d_{32}, e_8.For example, at step 99, we have $b_{94} \leftarrow (b_{99} >>14) - (K(c_{98}, d_{97}, e_{96}, a_{95}) + m_0 + t_5)$

⑧ Focusing on steps 95, 90,85,80, compute m_{17}, m_{16}, m_{19}, m_{18}. For example, the formula for m_{17} is given as

$m_{17} \leftarrow (a_{95} >>8) - (a_{90} + K(b_{94}, c_{93}, d_{92}, e_{91}) + t_5)$

At this time we have determined the values of all m (i=0,1,..., 19)

⑨ Froe steps 80,15,10, and 5, compute a_{15}, a_{10}, a_5, and a'_0 as follows.

$a_{15} \leftarrow (a_{80} >>5) - (K(b_{79}, c_{78}, d_{77}, e_{76}) + t_1)$

$a_5 \leftarrow (a_{10} >> 7) - (F(b_{14}, c_{13}, d_{12}, e_{11}) + t_1)$
$a_{10} \leftarrow (a_{15} >> 8) - (F(b_{14}, c_{13}, d_{12}, e_{11}) + t_1)$
$a'_0 \leftarrow (a_5 >> 6) - (F(b_4, c_3, d_2, e_1) + t_1)$

We note that all variables, except for a_0, are consistent.

⑩ If a'_0 is equal to a_0, which was determined at A-1, then output m_i (i=0,1,...,19) and terminate. Otherwise, go to A-1.

■ **The results of attack**

Dobbertin[6] has proposed an algorithm for finding preimages of MD4. Dobbertin's algorithm based on the inverted steps and used G functions of Boolean functions of MD4. That is, $G(x,y,z) = (x, y, z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z)$, ($\wedge$:AND, $\vee$:OR)

G has a character as follow:

$G(x, x, y) = G(x, y, x) = G(y, x, x) = x$

Dobbertin used this point. Kuwakado and Tanaka's used the same character although Boolean function is differ.

$F(x,y,z) = (x \wedge y) \vee (\neg x \wedge z)$.
F(w,ffffffff,ffffffff) = fffffff
F(00000000, w, fffffff) = fffffff
(ffffffff, 00000000, w) = 00000000

In order to find Primages in a reduced version of SHA-V, we must find the relationship in the Boolean functions as the upper conditions. In the case of SHA-V, See section3.3.

The Boolean functions of SHA-V have each three Boolean-variable. Although two variables are same, output is not a fixed value. That is not correspond with the previously conditions of Kuwakado and Tanaka's attack. We could not suppose message digest through input values (ex. a_{-5}, b_{-1}, c_{-2}, d_{-3}, e_{-4}). For example, in the next Eq. let the constant values are as follows.

$b_4 \leftarrow$0xffffffff, $c_3 \leftarrow$0xffffffff, and $d_2 \leftarrow$0xffffffff,
a_{-5}=0x67452301, b_{-1}=0x efcdab89, c_{-2}= 0x 98badefe,
d_{-3}= 0x 10325476, e_{-4}=0x c3d2e1f0.

And then, we cannot the calculations about m_{18}.

$m_{18} \leftarrow (a_0 >> 5) - (a_{-5} + F(b_{-1}, c_{-2}, d_{-3}, e_{-4}))$
$= (a_0 >> 5) - (a_{-5} + \text{0x 18315c7e})$
$= (a_0 >> 5) - (\text{0x67452301} + \text{0x 18315c7e})$
$= (a_0 >> 5) - (\text{0x 7f7f7f7f})$

In the previously ⑩, it must be same the value of a'_0 and a_0. Because we don't know the a_0 and cannot compare a'_0, the next Equation is impossible because the value of a_0 must be used to the next Equation. This problem is caused by F functions. The HAS-V employs 4-variable Boolean functions. The computations of Boolean functions require about 3 or 4 unit operations. It cannot find the previously conditions of MD4's attack method.

6. Compare to SHA-V and SHA-1/ 256

6.1. SHA-1 vs. SHA-V (160-bit)

We compare to the SHA-1 and SHA-V with 160 bits of output length as following <Table 6>.

<Table 6> Features of SHA-1 and SHA-V (160 bits)

	SHA-1	SHA-V(160-bit)
Endianess	Big	Big
Message Block Size	512	1024
Digest Size(Bits)	160	160
Word Size (bits)	32	64
The number of steps	80	80 ×2

■ **A Number of Computation and Speed**

The step operation of SHA-1 consists of 4 additions, 1 circular shifts and a Boolean function. The Boolean function consists of 3 unit operations and the step operations consists 80 steps. That is

1 (block) X 80(steps) X 10 (step operation)
= 800 (unit operations)

The SHA-V (160-bit) consists of 4 additions, 1 circular shifts and a Boolean function. The Boolean functions consist of 3 unit operations of 2 lines and the step operations consist of 80 steps of 2 lines.

1 (block) X 160(steps) X 10 (step operation)+5(output message expansion)
= 1605 (unit operations)

That is, SHA-1 is faster at computation about double than SHA-V(160-bit)

■ **Security Aspect**

The existed attacks on MD4 and MD5 were based on the weakness of simple Boolean function and the rather straightforward usage of the message words. A single message word of the input is only used once in every round of MD4 and MD5[11]. We discuss the security of

SHA-1 and SHA-V(160-bit) through such aspects.

- Input messages and Padding

The message block length of SHA-1 is 512 bits and SHA-V(160-bit) has 1024 bits(See Table 6). If SHA-1 and SHA-V(160-bit) have an input message of same length, the expansion function of SHA-V(160-bit) is large than SHA-1. It means that the diffusion of input messages in the SHA-V(160-bit) is more effective and provides the high security against attacks.

- Boolean functions

The existed attack methods on the cryptographic hash functions employ 3-variable Boolean functions. The SHA-1 has 3-variable Boolean functions. So, we cannot convince the security of SHA-1. The SHA-V(160-bit) has 3-varible Boolean functions of 2 lines. That is equal to the twice computation. Although SHA-V(160-bit) has 3-varible Boolean functions, it is equal to the twice computation on the same input message in other kind of Boolean functions.

6.2. SHA-256 vs. SHA-V (256-bit)

In this section, we compare SHA-256[14] and SHA-V(256-bit) as following <Table 7>.

<Table 7> Features of SHA-256 and SHA-V (256 bits)

	SHA-1	SHA-V(160-bit)
Endianess	Big	Big
Message Block Size	512	1024
Digest Size (Bits)	256	256
Word Size (bits)	32	64
The number of steps	64	80 ×2

■ **A Number of Computation and Speed**

The step operation of SHA-256 consists of 7 additions, 2 summations and the step operations consist of 64 steps. That is

1 (block) X 64(steps) X 15 (step operations)
= 960 (unit operations)

SHA-V (256-bit) consists of 4 additions, 1 circular shift and a Boolean function. The Boolean functions consist of 3 unit operations of 2 lines and the step operations consist of 80 steps of 2 lines.

1 (block) X 160(steps) X 10 (step operation) +16 (output message expansion)
= 1616 (unit operations)

That is, SHA-256 is 40% faster at computation about double than SHA-V(256-bit)

■ **Security Aspect**

- Input messages and Padding

The message block-length of SHA-256 is 512 bits and SHA-V(256-bit) has 1024 bits(See table8). The security effect is same as SHA-1 and SHA-V(160-bit)

- Functions

The SHA-256 used six logical functions which consist of XOR, AND, OR, Shift and Rotation. Each of these functions operates on 32-word and produces a 32-bit word as output. The SHA-V(256-bit) has 3-varible Boolean functions of 2 lines. For 256 bits of output in the SHA-V(256-bit), it has additionally 8 shift operations and 8 additions. Although SHA-256 is faster than SHA-V(256-bit), SHA-V(256-bit) is better in the security point of view than SHA-256 when it compares the number and units of computation.

7. Analysis of a Parallel Implementation

In this section, we introduce the analysis of a parallel implementation based on the Nakajima and Matsui's method[10], A.Bosselaers, R.Govaerts and J. Vandewalle's method[1].

■ **Processor Aspect**

The different of CISC(register-memory architecture), including the Intel 80x86 family, and RISC(register-register architecture), including SPARC, Aplha, and MIPS, is the number of on-chip general-purpose registers [1][16]. Pentium processor has only 7 general-propose registers on its chip. In case of SHA-V and HAS-V, a twin structure was employed and the number of chaining variables used in the step operation was chosen to be 5.

■ **Hardware parallelism**

The basic implementation technique to improve CPU performance is pipelining. A pipeline is organized in a number of stages, each of which executes part of a CPU instruction. To enhance performance even further two approaches are available: increase the number of pipeline stages, or use a number of parallel pipelines[1].

The SHA-1 has a natural algorithm parallelism in its compression function.[1][16]. The updated chaining variable in not used in the Boolean function of the next step operation, which has the effect of reducing the critical path length (CPL). This mechanism has been employed in the step operation of HAS-V and SAH-V[16]. The critical path of SHA-1 is shorter than any hash functions (see Table5). This result can apply to the SHA-V because Boolean functions and operations of SHA-V are based on SHA-1.

8. Conclusions

Hash functions play an important role in a branch of information secret. The hash algorithm provides the services of information security, authentication, integrity, non-reputation and so on. As the growth of computer technologies, the hash value has been become longer based on the complexity of calculation. It was known to be desirable that the output le ngths of hash functions are more than 160 bits.

We propose a hash function with variable output length. This SHA-V can be the selective using of output length by the program. Therefore, we can expect a lot of the practice. In the future, we must verify more security test of SHA-V and speed test in the hardware/software.

References

[1] A.Bosselaers, R.Govaerts and J.Vandewalle "SHA: A Design for Parallel Architecture?" Advances in Cryptology-Eurocrypt'97, pp348-3621,997.

[2] A.J. Menzs, P.C. Van Oorshot, S.A.Vanstone "Handbook of Applied Cryptography" CRC Press, 1997.

[3] Antoon Bosselaers " The Hash Function RIPEMD 160"http://www.esat.kuleuven.ac.be/~bosselae/ripemd160.html

[4] B.den Boer and A.Bosselaers " An attack on the last two rounds of MD4" Advances in Cryptology-Crypto'91, LNCS576, Springer-Verlag, pp 194-203, 1992.

[5] B.Schneier " Applied Cryptography" WILEY Press, Vol 2, 1996.

[6] C.H. Lim, N.K.Park, E.J. Lee, P.J.Lee " The proposal of the new hash function possible to select the output length", preprint, 1997 (Korean).

[7] H.Dobbertin, A. Bosselaers, B. Preneel " RIPEMD-160 : A strengened Version of RIPEMD" Fast Software Encryption, LNCS 1039, Springer-Verlag, pp71-82, 1996.
http://www.esat.kulenven.ac.be/~cosicart/pdf

[8] H.Dobbertin "Cryptanalysis of MD4" Fast Software Encryption, LNCS 1039, Springer-Verlag, pp53-69, 1996.

[9] H.Kuwakado, H.Tanaka " New Algorithm for Finding Preimage in a Reduced Version of the MD4 Compression Function " IEICE TRANS, Fundamentals, Vol.E83-A, No.1, Jan,2000.

[10] J.Nakajima, M.Matsui "Performance Analysis and Parallel Implementation of Dedicated Hash Functions" EUROCRYPT 2002, LNCS 2332, pp165-180, 2002.

[11] M.J.B.Robshaw " On Recent Results for MD2, MD4, MD5 " April, 1996.

[12] M.S.Lee " Modern Cryptography " KYOU Press, 1999.

[13] National Security Research Institute " Modern Cryptology" KYUNG-MOON Press, 2000.

[14] NIST "Descriptions of SHA-256, SHA-384, and SHA-512"2001.
http://csrc.nist.gov/cryptval/shs.html

[15] NIST "Secure Hash Standard" FIPS PUB180-1, May, 1993.
http://www.itl.nist.gov/fipsubs/fip180-1.htm

[16] N.K.Park, J.H.Hwang, P.J.Lee " HAS-V : A New Hash Function with Variable Output Length" SAC2000, LNCS2012, Springer-Verlag, pp.202-216, 2001.

[17] P.Sarkar, P,J, Schellenberg "A Parallelizable Design Principle for Cryptography Hash Functions" Indocrypt 2001, LNCS 2247, pp40-49, 2001.

[18] R.Rivest, "The MD4 Message Digest Algorithm" Advances in Cryptology-Crypto'90, LNCS 537, Springer-Verlag, pp303-311, 1991.

[19] R.Rivest "The MD5 Message Digest Algorithm" RFC 1321, Internet Activities Board, Internet Privacy Task Force, Apr, 1992.

[20] Telecommunication Technology Association " Hash Function Standard – Part 2 : Hash Function Algorithm Standard (HAS-160) " TTA, KO-12.0011/R1, Dec 2000.

[21] Y.S.HER, K.SAKURAI " Design and Analysis Cryptographic Hash Function for The Next Generation " IWIE2002, Proc. Pp168-173, May 2002.

[22] Y.Zheng, J.Pieprzyk and J.Sebbery, "HAVAL-A – One-Way Hashing Algorithm with Variable Length of Output" Advances in Cryptography-AUSCRYPT '92,LNCS718, Springer-Verlag, pp83-104, 1993.