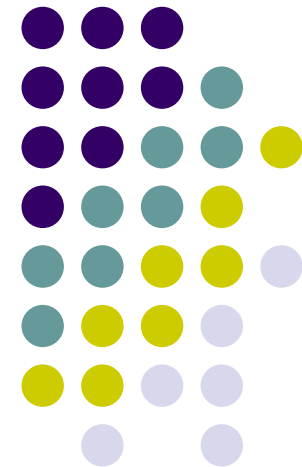


IAS 2008
September 8-10, 2008
Naples, Italy



PHAL-256
Parameterized Hash Algorithm



Przemysław RODWALD, Military Communication Institute, Poland



Janusz STOKŁOSA, Poznań University of Technology, Poland



Agenda



- Introduction
- Hash functions
 - Definition, properties
 - State of art in cryptanalysis
- Parameterized Hash ALgorithm – PHAL-256
 - Iteration schema
 - Compression function
 - Security analysis
 - Efficiency
- Summary

Introduction

Why new hash functions?



- Great progress in cryptanalysis of hash functions
- Several flaws in Merkle-Damgård's construction
- NIST call for SHA-3 (AHS)

<http://www.nist.gov/hash-competition>

NIST - National Institute of Standards and Technology

SHA - Secure Hash Algorithm

AHS - Advanced Hash Standard



Definition

Hash function h -

computationally efficient function mapping an input (message m) of arbitrary finite bitlength, to an output of fixed bitlength n .

$$h : \{0,1\}^* \rightarrow \{0,1\}^n$$



Properties of hashes

- ***Preimage resistance (non-invertibility):***
it is computationally infeasible to find any input which hashes to that output
- ***2nd preimage resistance (weak collision resistance):***
it is computationally infeasible to find any second input (m') which has the same output as any specified input (m)
- ***Collision resistance (strong collision resistance):***
it is computationally infeasible to find any two distinct inputs m and m' which hash to the same output $h(m) = h(m')$



Cryptanalysis

1990

1991

1992

1993

1994

1996

2002

20??



*Xiaoyun
Wang*

RIPEMD-160

SHA-2

SHA-3

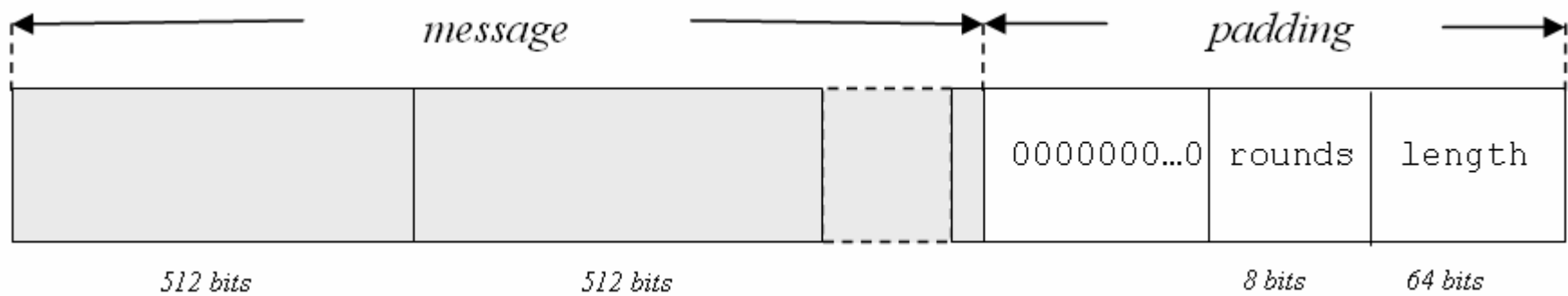
PHAL-256



- Message Padding
- Iteration Schema
- Message Modification
- Constants and Initial value
- Compression Function
- Statistical Tests
- Efficiency

PHAL-256

Message Padding



PHAL-256

Iteration Schema



Hash function is designed using the compression function

$$\varphi: \{0,1\}^n \times \{0,1\}^m \times \{0,1\}^c \times \{0,1\}^s \times \{0,1\}^{rs} \rightarrow \{0,1\}^n$$

where:

- n : length of the chaining variable (256 bits),
- m : length of the message block (512 bits),
- c : the size of counter (number of bits hashed so far) modulo 2^{64} (64 bits)
- s : length of the *salt* (128 bits),
- rs : the size of number of *rounds* (4 bits),
- *rounds* and *salt* are values defined by the user.

The process of hashing looks as follows::

$$CV_0 = IV,$$

$$CV_{i+1} = \varphi(CV_i, m_i, counter, salt, rounds), \quad i = 0, 1, \dots, k-1,$$

$$h(m) = CV_k.$$

PHAL-256

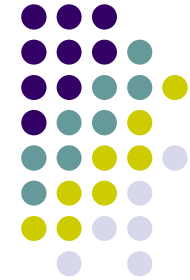
Message Modification



```
Wix[0] := Wix[0] ⊕ Wix[15] ⊕ 0x5FA2C0D3
Wix[1] := Wix[1] ⊕ Wix[0]
Wix[2] := Wix[2] ⊕ Wix[1]
Wix[3] := Wix[3] ⊕ Wix[2] ⊕ (¬Wix[1] ≪≪ 1)
Wix[4] := Wix[4] ⊕ Wix[3]
Wix[5] := Wix[5] ⊕ Wix[4]
Wix[6] := Wix[6] ⊕ Wix[5]
Wix[7] := Wix[7] ⊕ Wix[6] ⊕ (¬Wix[5] ≫≫ 5)
Wix[8] := Wix[8] ⊕ Wix[7] ⊕ 0xB1487E96
Wix[9] := Wix[9] ⊕ Wix[8]
Wix[10] := Wix[10] ⊕ Wix[9]
Wix[11] := Wix[11] ⊕ Wix[10] ⊕ (¬Wix[9] ≪≪ 13)
Wix[12] := Wix[12] ⊕ Wix[11]
Wix[13] := Wix[13] ⊕ Wix[12]
Wix[14] := Wix[14] ⊕ Wix[13]
Wix[15] := Wix[15] ⊕ Wix[14] ⊕ (¬Wix[13] ≫≫ 3)
```

PHAL-256

Constants and Initial Value



$\Omega[0] = \text{salt}[0], \Omega[4] = \text{salt}[1] \oplus \text{counter},$
 $\Omega[8] = \text{salt}[2], \Omega[12] = \text{salt}[3] \boxplus \text{counter},$
 $\Omega[1] = 0x698D3AD4, \Omega[2] = 0xA62E8B66, \Omega[3] = 0x557255A9,$
 $\Omega[5] = 0x9AD1E41B, \Omega[6] = 0x2D3CF0C3, \Omega[7] = 0x3AC6359C,$
 $\Omega[9] = 0xC5638B36, \Omega[10] = 0xD2994E69, \Omega[11] = 0x5393E178,$
 $\Omega[13] = 0xB82E1D6C, \Omega[14] = 0x0F556A93, \Omega[15] = 0xE4E89687.$

Table. Constants word ordering

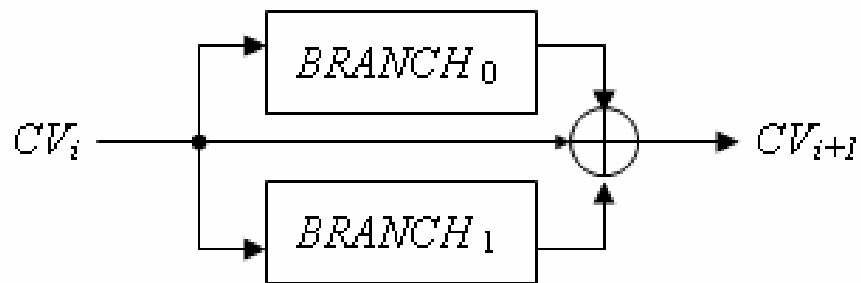
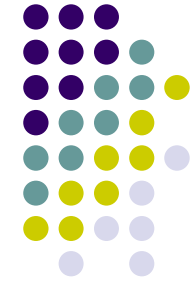
t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\rho_0(t)$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\rho_1(t)$	13	8	12	9	15	14	10	11	6	0	2	4	1	5	7	3

Table. Hamming Weight of Initial Value ~~BHA-256~~

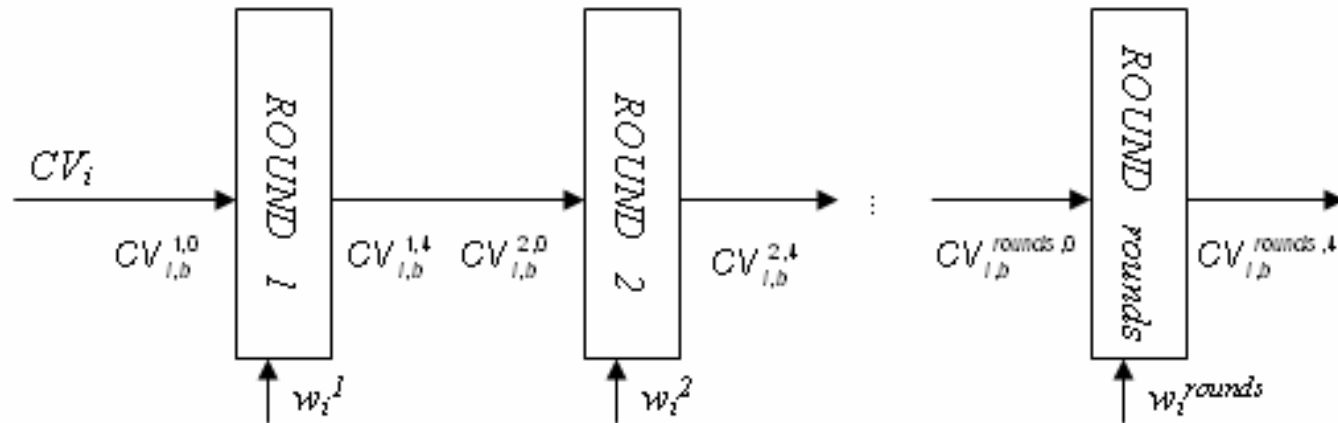
c9b4361d	11001001	10110100	00110110	00011101	4+4+4+4=16
b14ed478	10110001	01001110	11010100	01111000	4+4+4+4=16
635a65c6	01100011	01011010	01100101	11000110	4+4+4+4=16
9c271eac	10011100	00100111	00011110	10101100	4+4+4+4=16
e455a927	11100100	01010101	10101001	00100111	4+4+4+4=16
1ed84be2	00011110	11011000	01001011	11100010	4+4+4+4=16
53a3ca59	01010011	10100011	11001010	01011001	4+4+4+4=16
2ea9b193	00101110	10101001	10110001	10010011	4+4+4+4=16
	44444444	44444444	44444444	44444444	<i>hw</i>

PHAL-256

Compression Function



Picture. Compression Function

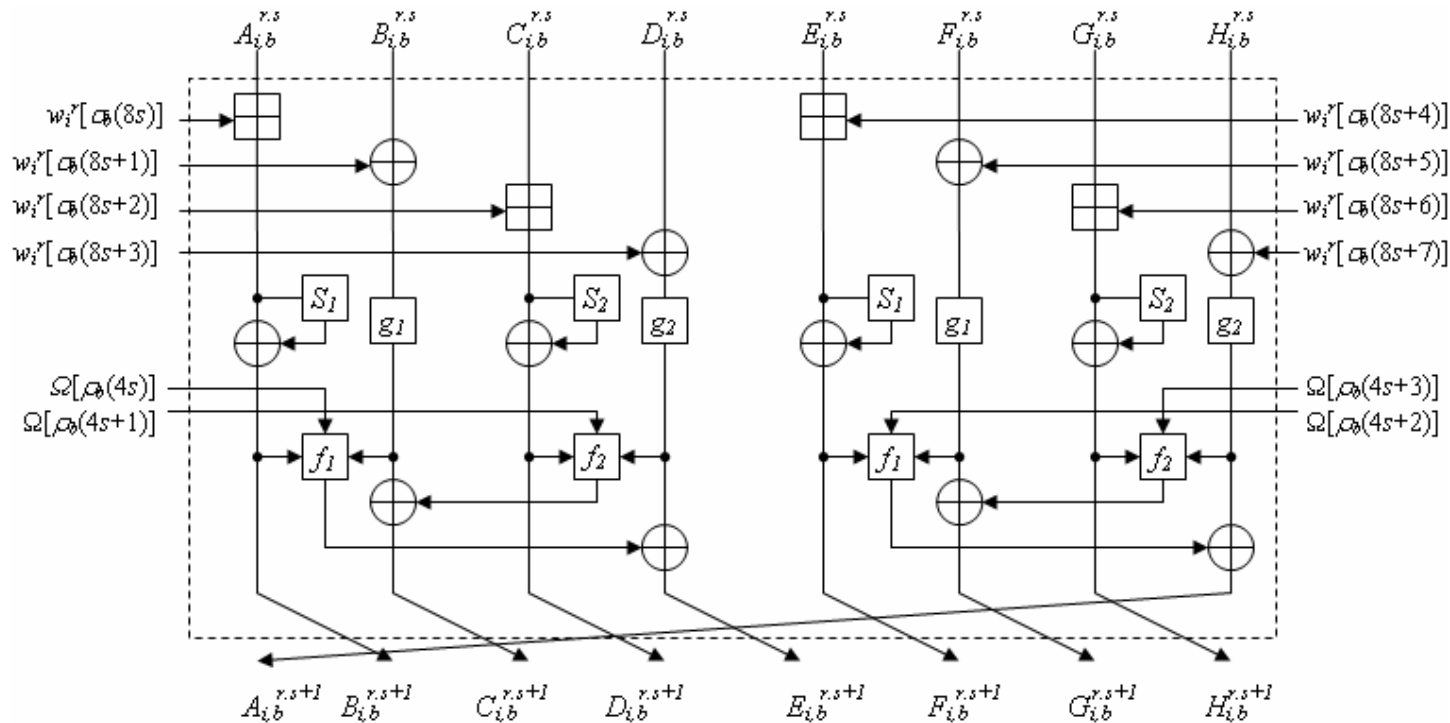


Picture. Branch Function (BRANCH)

PHAL-256



Step Function



Picture. Step Function.

- $g_1(x) = x \oplus (x \lll 5) \oplus (x \lll 18)$
- $g_2(x) = x \oplus (x \lll 13) \oplus (x \lll 27)$
- $f_1(x, y, z) = xy \oplus (\neg x)z$
- $f_2(x, y, z) = xy \oplus xz \oplus yz$
- $S_1(x) = SBox1(x \gg 16) \oplus SBox0(x \gg 24)$
- $S_2(x) = SBox1(x) \oplus SBox0(x \gg 8)$

PHAL-256

Statistical Tests



NIST statistical tests suite for random and pseudorandom number generators for cryptographic applications

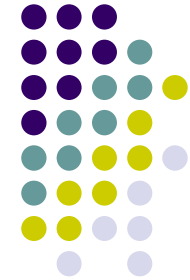
<http://csrc.nist.gov/groups/ST/toolkit/rng/>

Tested messages:

- different number of rounds ($rounds \in \{1,2,3,4\}$)
- different salt values
- random messages
- sparse messages (Hamming weight $\in \{1,2,3,509,510,511\}$)

PHAL-256

Efficiency



	PHAL-256			SHA-256
	1 round	2 rounds	3 rounds	
Message expansion or modification	\boxplus : 0 \oplus : 0 \ll : 0	\boxplus : 36 \oplus : 54 \ll : 24	\boxplus : 72 \oplus : 108 \ll : 48	\boxplus : 144 \oplus : 384 \ll : 480
Compression function	\boxplus : 32 \oplus : 304 \ll : 128 \wedge : 80	\boxplus : 64 \oplus : 608 \ll : 256 \wedge : 160	\boxplus : 96 \oplus : 912 \ll : 384 \wedge : 240	\boxplus : 448 \oplus : 832 \ll : 768 \wedge : 320
Output transformation	\oplus : 16	\oplus : 16	\oplus : 16	\boxplus : 8

Hash function	Average Speed [Mbps]
MD5	355
PHAL-256 (3 rounds)	88
SHA-256	84
PHAL-256 (4 rounds)	66

Summary



PHAL-256 :

- flexibility
- increased resistance to attacks against MD-type structure
- two parallel branches
- efficiency comparable with SHA-256

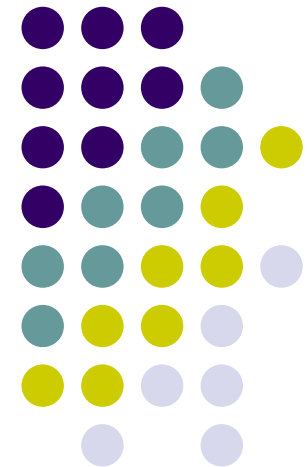
- good candidate for SHA-3 ?

Even in cryptology, silence is golden.

Laurence D.Smith

Thank you for your attention

Any questions?



Contact

janusz.stoklosa@put.poznan.pl

p.rodwald@wil.waw.pl