Higher order differential MiTM preimages attacks

CRYPTO 15, Santa Barbara

Thomas S π τ [1,2] *Pierre-Alain Fouque*[3,4] *Pierre Karpman*[2,5]

1 École normale supérieure de Cachan, France
2 Inria, France
3 Université de Rennes 1, France
4 Institut Universitaire de France, France
5 Nanyang Technological University, Singapore

Outline

Hash Functions

$$F: \{0,1\}^* \longrightarrow \{0,1\}^k$$

- Compress a message to a fixed-size hash.
- Example of applications: *Hash & Sign*.

MD4("Thomas Espitau") = 41567fe4aeaf92f9affa00a7f015d0e7

MD4("Thomas Espitou") =17280cc68a26f22e2d2ba5da6a23aa

Avalanche effect

Hash Functions

3 notions of security

Collisions

Find M, N such that:

$$\boxed{F(M) = F(N)}$$

$$\boxed{2^{k/2}}$$

(Birthday paradox)

Preimage

For C, find M such that:

$$\boxed{F(M) = C}$$

$$\boxed{2^{k}}$$

2nd Preimage

For N, if C = F(N), find M such that:

$$\boxed{F(M) = C}$$

$$\boxed{2^{k}}$$

Construction

Markle-Damgård scheme

One way compression funtion

conservation of security

propreties

Hash function

Construction

Markle-Damgård scheme

One way compression funtion $\xrightarrow[\text{propreties}]{\text{conservation of security}}$ Hash function

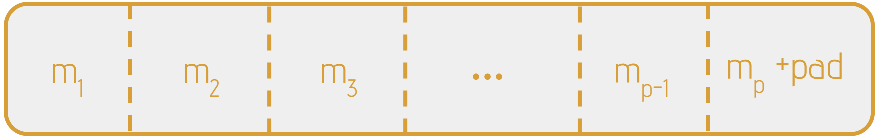
$\boxed{m}$

Construction

Markle-Damgård scheme

One way compression funtion → Hash function

conservation of security propreties



Construction

Markle-Damgård scheme

One way compression funtion

conservation of security

propreties

Hash function

Construction

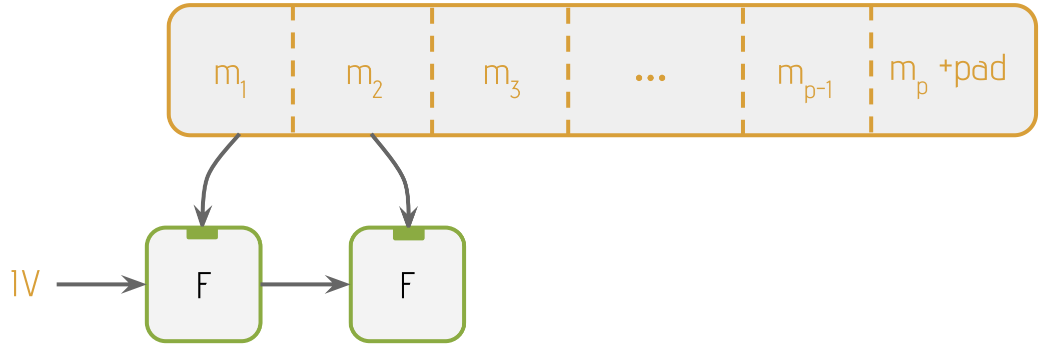
Markle-Damgård scheme

One way compression funtion

conservation of security

propreties

Hash function



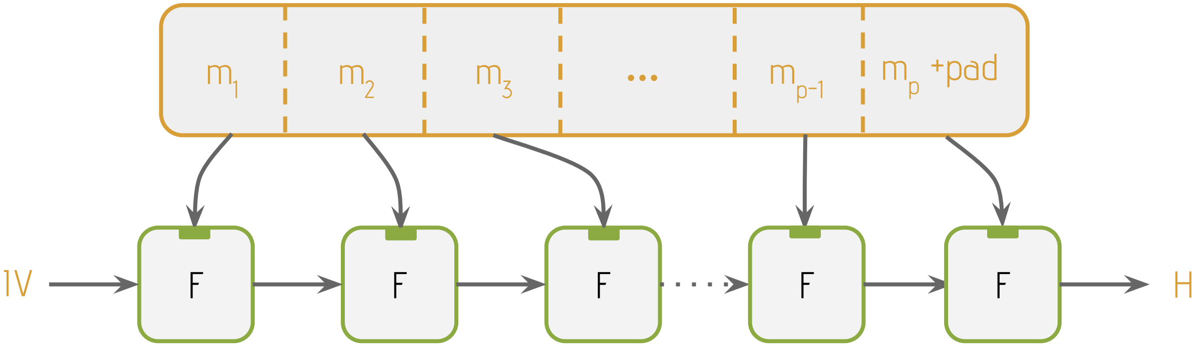
Construction

Markle-Damgård scheme

One way compression funtion

conservation of security propreties

Hash function

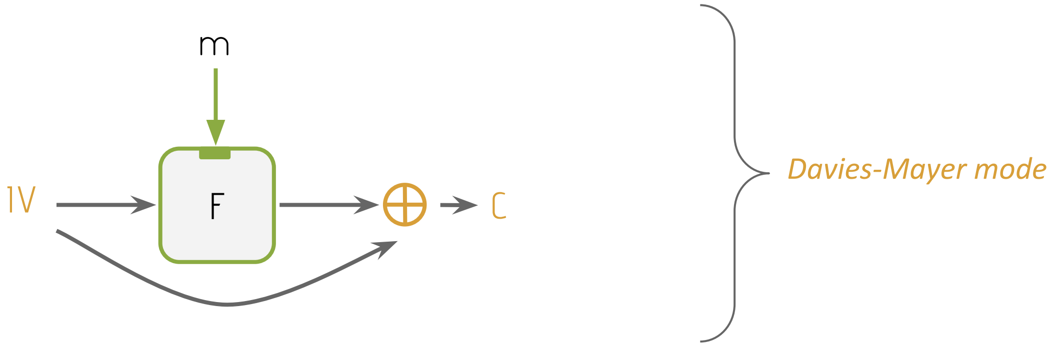


Differential Meet-in-the-middle

Framework of Knellwolf & Khovratovich, 2012

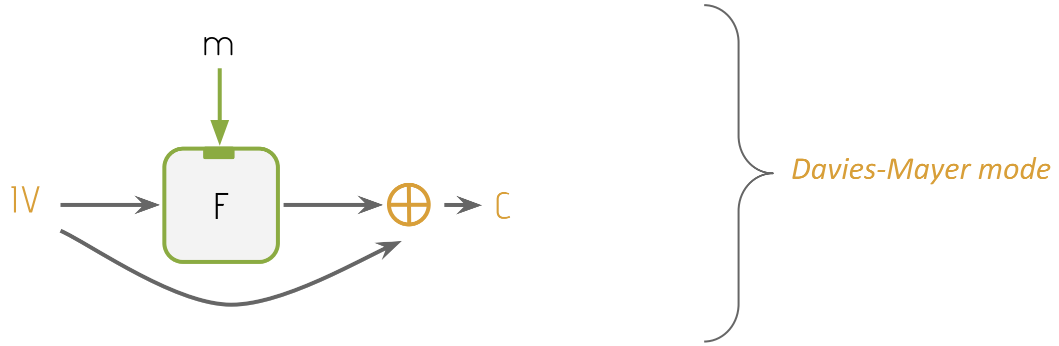
Differential Meet-in-the-middle

Framework of Knellwolf & Khovratovich, 2012



Differential Meet-in-the-middle

Framework of Knellwolf & Khovratovich, 2012



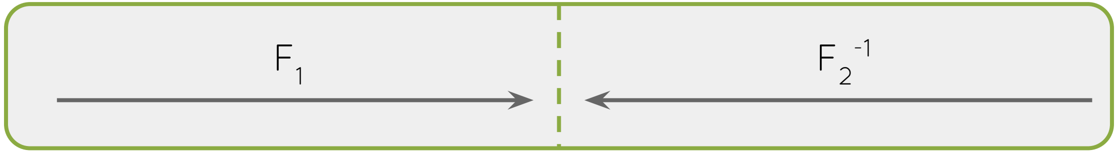
Find preimage for the compression function : find a preimage of $H = C + IV$ by the function $F(\_ , IV)$

Differential Meet-in-the-middle

Framework of Knellwolf & Khovratovich, 2012

Compression function cut in two chunks:

$F = F1 \circ F2$



Differential Meet-in-the-middle

Framework of Knellwolf & Khovratovich, 2012

$\underbrace{D_1 \,,\; D_2}_{\textit{dim } n}$ two sub spaces in direct sum in the space of messages.

Differential Meet-in-the-middle

Framework of Knellwolf & Khovratovich, 2012

D_1 , D_2 two sub spaces in direct sum in the space of messages.

$$\boxed{F_1(M + \partial_1 , IV) = F_1(M, IV) + \Delta_1}$$

Differential Meet-in-the-middle

Framework of Knellwolf & Khovratovich, 2012

D_1 , D_2 two sub spaces in direct sum in the space of messages.

$$\boxed{F_1(M + \partial_1 , IV) = F_1(M, IV) + \Delta_1}$$

$\partial_1 \longrightarrow \Delta_1$ *is a message differential of probability 1*

Differential Meet-in-the-middle

Framework of Knellwolf & Khovratovich, 2012

D_1 , D_2 two sub spaces in direct sum in the space of messages.

$$F_1(M + \partial_1 , IV) = F_1(M, IV) + \Delta_1$$

$$F_2^{-1}(M + \partial_2 , H) = F_2^{-1}(M, H) + \Delta_2$$

Differential Meet-in-the-middle

Framework of Knellwolf & Khovratovich, 2012

D_1 , D_2 two sub spaces in direct sum in the space of messages.

$$F_1(M + \partial_1 , IV) = F_1(M, IV) + \Delta_1$$

$$F_2^{-1}(M + \partial_2 , H) = F_2^{-1}(M, H) + \Delta_2$$

$\partial_2 \longrightarrow \Delta_2$ *is a message differential of probability 1*

Differential Meet-in-the-middle

What happens if $M + \partial_1 + \partial_2$ is a preimage?

Differential Meet-in-the-middle

What happens if $M + \partial_1 + \partial_2$ is a preimage?

$$F_1(M + \partial_1 + \partial_2, IV) = F_2^{-1}(M + \partial_1 + \partial_2, H)$$

Differential Meet-in-the-middle

What happens if $M + \partial_1 + \partial_2$ is a preimage?

$$F_1(M + \partial_2, IV) + \Delta_1 = F_2^{-1}(M + \partial_1, H) + \Delta_2$$

Differential Meet-in-the-middle

What happens if $M + \partial_1 + \partial_2$ is a preimage?

$$F_1(M + \partial_2, IV) + \Delta_2 = F_2^{-1}(M + \partial_1, H) + \Delta_1$$

Differential Meet-in-the-middle

What happens if $M + \partial_1 + \partial_2$ is a preimage?

$$\underbrace{F_1(M + \partial_2, IV) + \Delta_2}_{\text{only depends on } \partial_2} = \underbrace{F_2^{-1}(M + \partial_1, H) + \Delta_1}_{\text{only depends on } \partial_1}$$

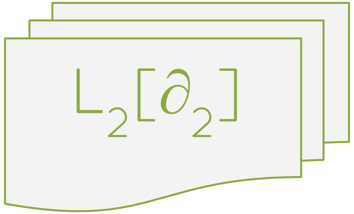
Differential Meet-in-the-middle

What happens if $M + \partial_1 + \partial_2$ is a preimage?

$$F_1(M + \partial_2, IV) + \Delta_2 = F_2^{-1}(M + \partial_1, H) + \Delta_1$$

only depends on ∂_2

only depends on ∂_1



Differential Meet-in-the-middle

What happens if $M + \partial_1 + \partial_2$ is a preimage?

$$F_1(M + \partial_2, IV) + \Delta_2 = F_2^{-1}(M + \partial_1, H) + \Delta_1$$

only depends on ∂_2

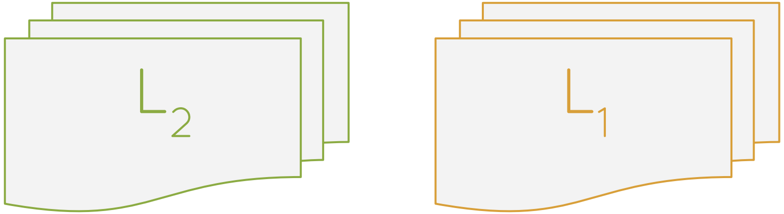
only depends on ∂_1

$L_2[\partial_2]$

$L_1[\partial_1]$

Differential Meet-in-the-middle

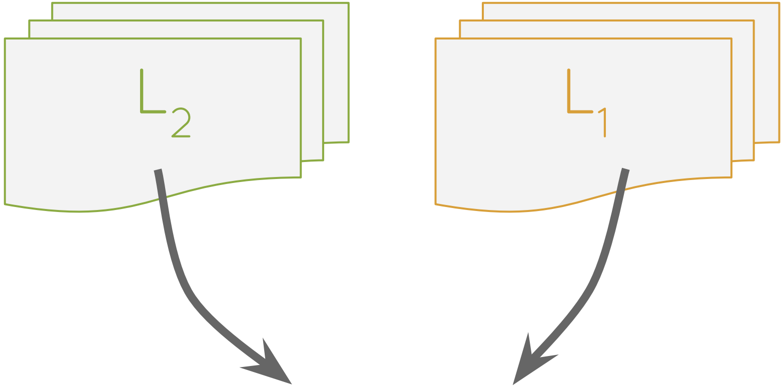
Algorithm



Computation of the two lists (independantly)

Differential Meet-in-the-middle

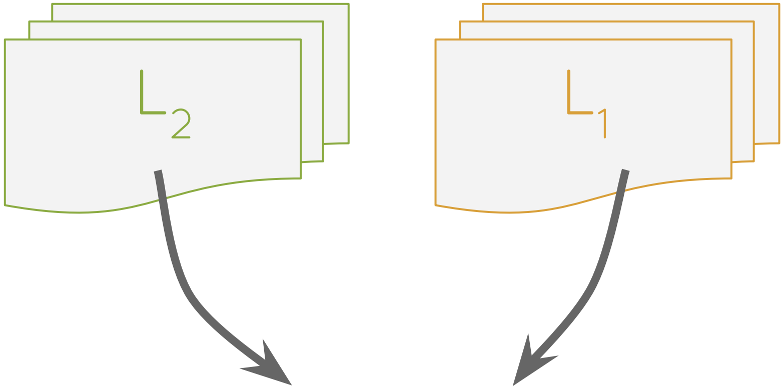
Algorithm



Lookup of a common element

Differential Meet-in-the-middle

Algorithm

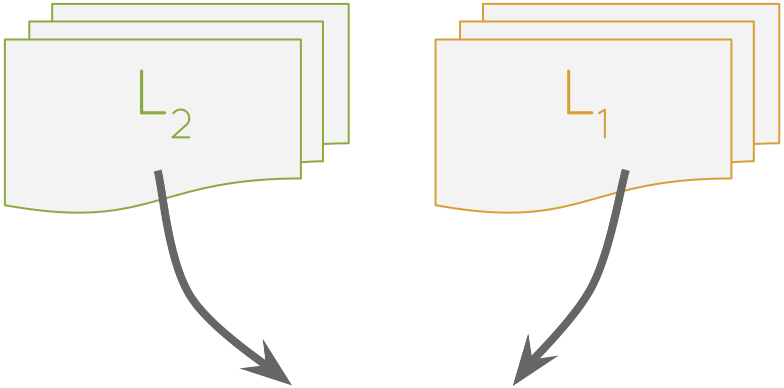


$$L_2[\partial_2] = L_1[\partial_1]$$

Lookup of a common element

Differential Meet-in-the-middle

Algorithm



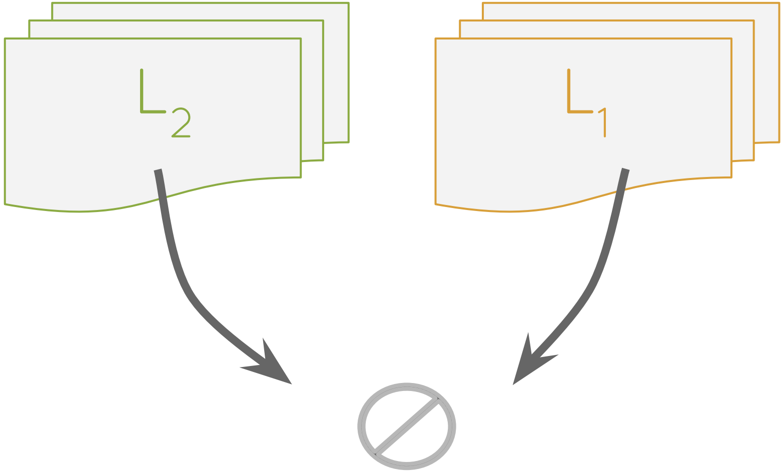
Lookup of a common element

$$L_2[\partial_2] = L_1[\partial_1]$$

$M + \partial_1 + \partial_2$ *is a preimage*

Differential Meet-in-the-middle

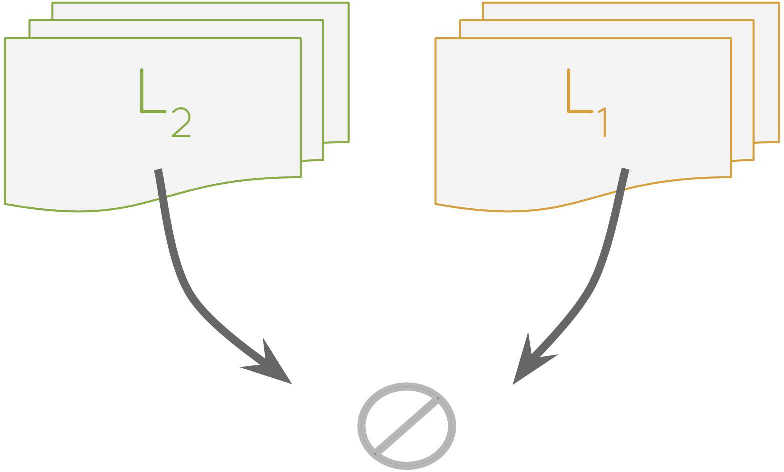
Algorithm



Lookup of a common element

Differential Meet-in-the-middle

Algorithm

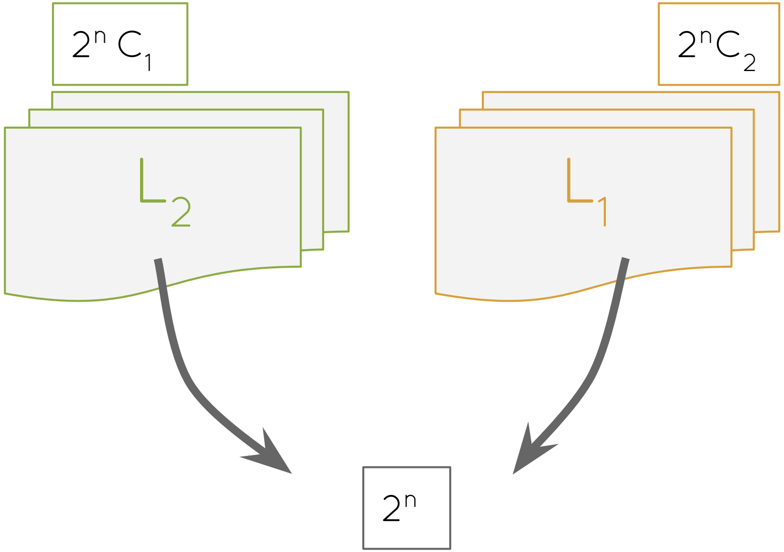


Lookup of a common element

$M + D_1 + D_2$ *doesn't contain a preimage*

Differential Meet-in-the-middle

Algorithm (complexity)



Computation of the two lists (independantly)

Lookup of a common element

$\boxed{2^n C_1 + 2^n C_2 = 2^n C \text{ for testing } 2^{2n} \text{ messages}}$

Differential Meet-in-the-middle

Algorithm (complexity)

$2^n C_1 + 2^n C_2 = 2^n C$ for testing 2^{2n} messages

Differential Meet-in-the-middle

Algorithm (complexity)

$$\boxed{2^n C_1 + 2^n C_2 = 2^n\, C \text{ for testing } 2^{2n} \text{ messages}}$$

To obtain a preimage we need to test 2^k messages.

Algorithm must be launched 2^{k-2n} times to test these messages.

Total complexity: $2^{k-2n} . 2^n\, C = 2^{k-n}\, C$

Higher order differentials

Lai (94) , Knudsen (94)

Higher order differentials

Lai (94) , Knudsen (94)

$$F(M + \partial) = F(M) + \Delta$$

Higher order differentials

Lai (94) , Knudsen (94)

$$F(M + \partial) + F(M) = \Delta$$

Higher order differentials

Lai (94) , Knudsen (94)

$$D_{\partial}(F)(M) = \Delta$$

Higher order differentials

Lai (94) , Knudsen (94)

$$\underbrace{D_{\partial}}(F)(M) = \Delta$$

D_{∂} *is a finite difference operator*

Higher order differentials

Lai (94) , Knudsen (94)

$$\underbrace{D_{\partial}}(F)(M) = \Delta$$

D_{∂} *is a finite difference operator*

$$D_{a,b,\ldots,n}(F) = D_{a}(D_{b,\ldots,n}(F))$$

Higher order differentials

Lai (94) , Knudsen (94)

$$D_{\partial}(F)(M) = \Delta$$

D_{∂} *is a finite difference operator*

$$D_{a,b}(F)(M) = F(M+a)+F(M+b)+F(M+a+b)+F(M)$$

Higher order differentials

Lai (94) , Knudsen (94)

$$\underbrace{D_{\partial}}(F)(M) = \Delta$$

D_{∂} is a *finite difference operator*

$$\Pr[\ D_{a,b}(F)(M) = \Delta\ ] = p$$

$a,b \longrightarrow \Delta$ *is an order 2 message differential of probability p*

Higher order differentials MiTM

D_1 , D_2 , D_3 , D_4 four sub spaces in direct sum in the space of messages.

$$F_1(M + \partial_1 + \partial_3 , IV) + F_1(M + \partial_3 , IV) + F_1(M + \partial_1 , IV) = F_1(M, IV)$$

$\partial_1\partial_3 \longrightarrow 0$ *is an order 2 message differential of probability 1*

Higher order differentials MiTM

D_1 , D_2 , D_3 , D_4 four sub spaces in direct sum in the space of messages.

$$\boxed{F_2^{-1}(M + \partial_2 + \partial_4 , H) + F_2^{-1}(M + \partial_2 , H) + F_2^{-1}(M + \partial_4 , H) = F_2^{-1}(M, H)}$$

$\partial_2\partial_4 \longrightarrow 0$ *is an order 2 message differential of probability 1*

Higher order differentials MiTM

Previously:

$$F_1(M + \partial_1 + \partial_2 \quad , IV) = F_2^{-1}(M + \partial_1 + \partial_2 \quad , H)$$

Higher order differentials MiTM

With HOD:

$$F_1\,(M + \partial_1 + \partial_2 + \partial_3 + \partial_4\,,\, IV) = F_2^{-1}\,(M + \partial_1 + \partial_2 + \partial_3 + \partial_4\,,\, H)$$

Higher order differentials MiTM

With HOD:

$$F_1(M + \partial_3 + \partial_2 + \partial_4, IV) + F_1(M + \partial_1 + \partial_2 + \partial_4, IV) + F_1(M + \partial_2 + \partial_4, IV)$$

$$= F_2^{-1}(M + \partial_2 + \partial_1 + \partial_3, H) + F_2^{-1}(M + \partial_4 + \partial_1 + \partial_3, H) + F_2^{-1}(M + \partial_1 + \partial_3, H)$$

Higher order differentials MiTM

With HOD:

$$\overbrace{F_1(M + \partial_3 + \partial_2 + \partial_4, IV)}^{\text{3 indices}} + \overbrace{F_1(M + \partial_1 + \partial_2 + \partial_4, IV)}^{\text{3 indices}} + \overbrace{F_1(M + \partial_2 + \partial_4, IV)}^{\text{2 indices}}$$

$$= \underbrace{F_2^{-1}(M + \partial_2 + \partial_1 + \partial_3, H)}_{\text{3 indices}} + \underbrace{F_2^{-1}(M + \partial_4 + \partial_1 + \partial_3, H)}_{\text{3 indices}} + \underbrace{F_2^{-1}(M + \partial_1 + \partial_3, H)}_{\text{2 indices}}$$

Higher order differentials MiTM

$A[\partial_3, \partial_2, \partial_4]$ $B[\partial_1, \partial_2, \partial_4]$ $C[\partial_2, \partial_4]$

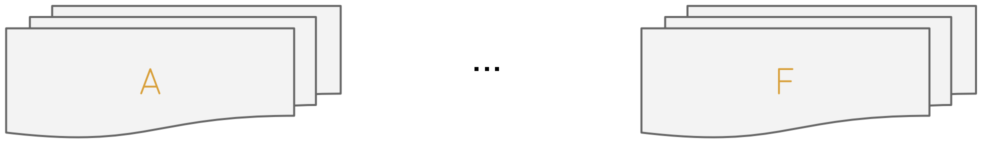
$$F_1(M + \partial_3 + \partial_2 + \partial_4, IV) + F_1(M + \partial_1 + \partial_2 + \partial_4, IV) + F_1(M + \partial_2 + \partial_4, IV)$$

$$= F_2^{-1}(M + \partial_2 + \partial_1 + \partial_3, H) + F_2^{-1}(M + \partial_4 + \partial_1 + \partial_3, H) + F_2^{-1}(M + \partial_1 + \partial_3, H)$$

$D[\partial_1, \partial_3, \partial_2]$ $E[\partial_1, \partial_3, \partial_4]$ $F[\partial_1, \partial_3]$

Differential Meet-in-the-middle

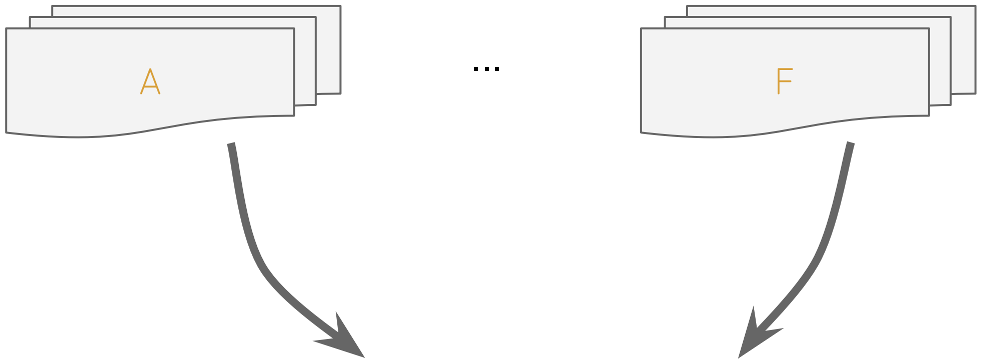
Algorithm



Computation of the 6 lists (independantly)

Differential Meet-in-the-middle

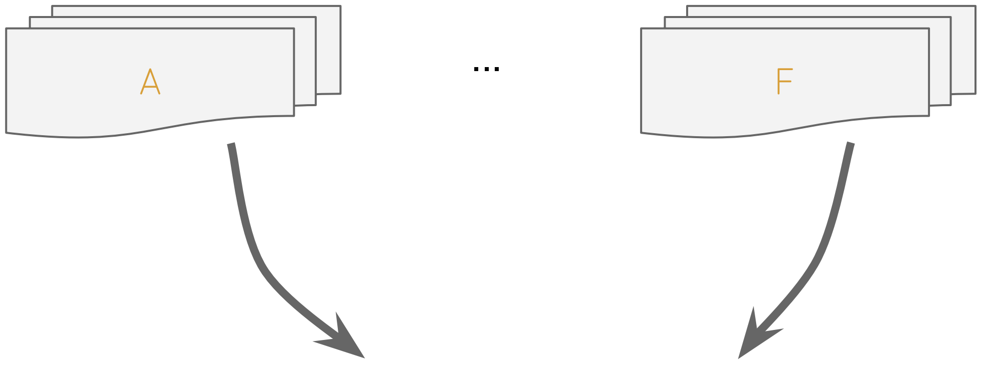
Algorithm



Lookup for $\partial_{1,}\partial_3$ $\partial_2\partial_4$ such that the equality is fullfilled

Differential Meet-in-the-middle

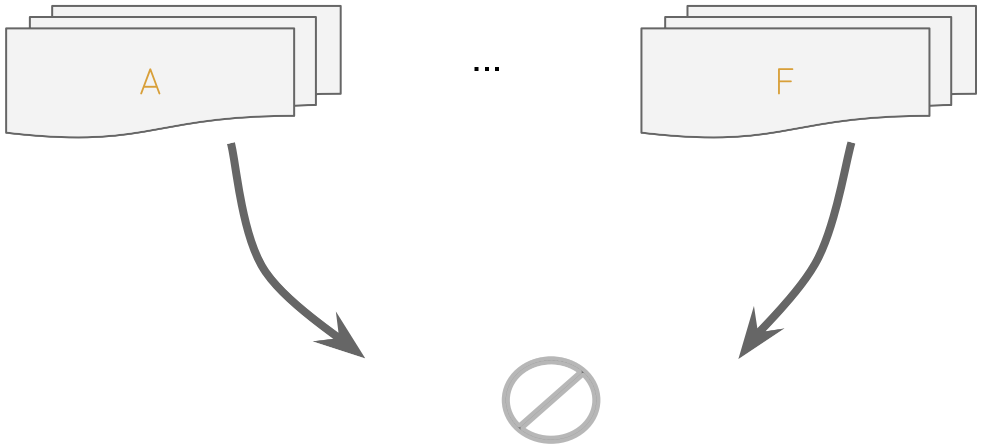
Algorithm



$M + \partial_1 + \partial_2 + \partial_3 + \partial_4$ *is a preimage*

Differential Meet-in-the-middle

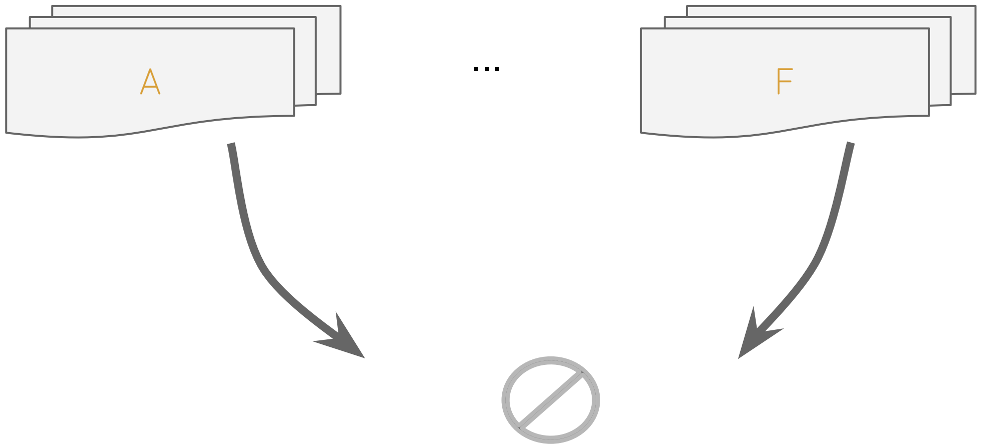
Algorithm



Computation of the two lists (independantly)

Differential Meet-in-the-middle

Algorithm



Computation of the two lists (independantly)

$M + D_1 + D_2 + D_3 + D_4$ *doesn't contain a preimage*

Differential Meet-in-the-middle

Algorithm (Complexity)

Differential Meet-in-the-middle

Algorithm (Complexity)

$A[\partial_3, \partial_2, \partial_4]$ — $2^{3n} C_1$

$B[\partial_1, \partial_2, \partial_4]$ — $2^{3n} C_1$

$C[\partial_2, \partial_4]$ — $2^{2n} C_1$

$D[\partial_1, \partial_3, \partial_2]$ — $2^{3n} C_2$

$E[\partial_1, \partial_3, \partial_4]$ — $2^{3n} C_2$

$F[\partial_1, \partial_3]$ — $2^{2n} C_2$

Differential Meet-in-the-middle

Algorithm (Complexity)

$A[\partial_3, \partial_2, \partial_4]$ — $2^{3n} C_1$

$B[\partial_1, \partial_2, \partial_4]$ — $2^{3n} C_1$

$C[\partial_2, \partial_4]$ — $2^{2n} C_1$

$2.2^{3n}C_1 + 2.2^{3n}C_2 + 2^{2n}C_1 + 2^{2n}C_2 = (2^{3n} + 2^{2n})C$ for testing 2^{4n} messages

$2^{3n} C_2$ — $D[\partial_1, \partial_3, \partial_2]$

$2^{3n} C_2$ — $E[\partial_1, \partial_3, \partial_4]$

$2^{2n} C_2$ — $F[\partial_1, \partial_3]$

Differential Meet-in-the-middle

Algorithm (Complexity)

$2^{3n}C$ for testing 2^{4n} messages

Differential Meet-in-the-middle

Algorithm (Complexity)

$2^{3n}C$ for testing 2^{4n} messages

To obtain a preimage we need to test 2^k messages.

Algorithm must be launched 2^{k-4n} times to test these messages.

Total complexity: $2^{k-4n}.2^{3n}C = 2^{k-n}C$

Applications

Let's break things!

Applications

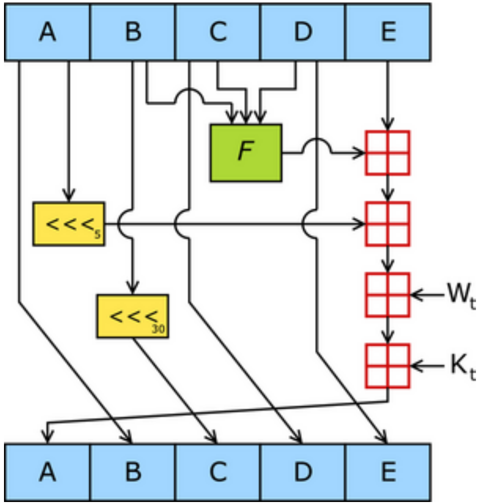
SHA-1

- Part of the MD4 family
- Hash size is 160 bits $\Rightarrow$ Preimage security should be 160 bits
- Message blocks are 512-bit long

Applications

SHA-1

- Block cipher in Davies-Meyer mode
- Structure is a 5-branch ARX Feistel with a linear message expansion



Applications

SHA-1

- 62 rounds attacked (over 80) with two blocks & correct padding *[$2^{159.3}$]*
  Prev. 57 rounds *[$2^{158.8}$]*, now 57 rounds *[$2^{157.9}$]*

- 56 rounds attacked with one blocks & correct padding *[$2^{156.7}$]*
  Prev. 52 rounds *[$2^{158.4}$]*, now 52 rounds *[$2^{156.7}$]*

- 64 rounds attacked in pseudo-preimage *[$2^{156.7}$]*
  Prev. 60 rounds *[$2^{157.4}$]*, now 61 rounds *[$2^{156.7}$]*

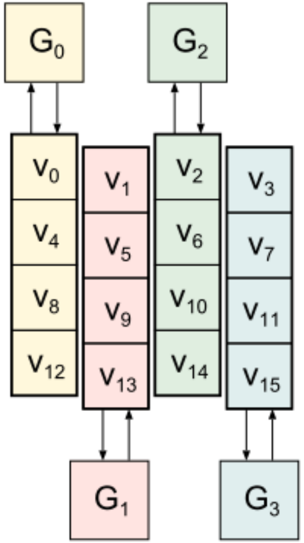
Applications

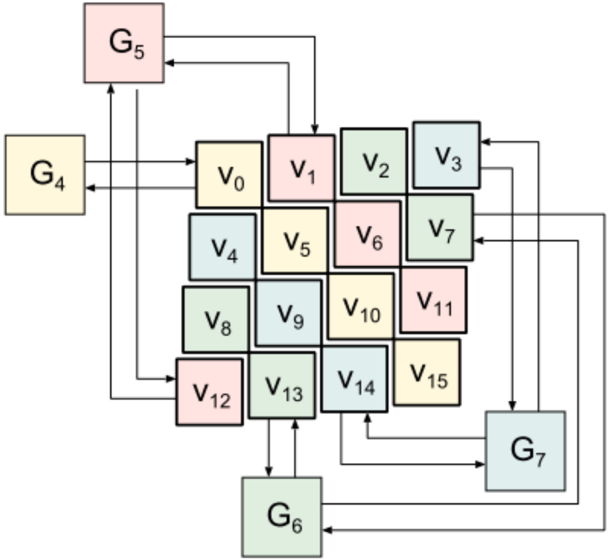
BLAKE-BLAKE2

- BLAKE is a SHA-3 finalist.   BLAKE2 is a faster version.
- Designed for high performances.
- BLAKE-256 (resp. 512) , works with 32 (resp. 64)-bit words, produce 256-(resp 512)-bits digests.

Applications

BLAKE-BLAKE2





Applications

BLAKE-BLAKE2

- 2.75 rounds attacked of BLAKE-512, BLAKE2b. (previously 2.5) *[$2^{510.3}$] [$2^{510.3}$]*
- 6.75 rounds attacked in c.f. pseudo-preimage of BLAKE-256, BLAKE2s. *[$2^{253.9}$] [$2^{253.8}$]*
- 7.5 rounds attacked in c.f. pseudo-preimage of BLAKE-512, BLAKE2b. *[$2^{510.3}$] [$2^{510.3}$]*

The end

59a4caddf715280f7a9e5da6f54e6abc19b22e49