Cryptanalysis of the ESSENCE Hash Function

M. Naya-Plasencia[1] A. Röck[2] J-P. Aumasson[3]
Y. Laigle-Chapuy[1] G. Leurent[4] W. Meier[3] T. Peyrin[5]

[1]INRIA project-team SECRET, France
[2]Helsinki University of Technology (TKK), Finland
[3]FHNW, Windisch, Switzerland
[4]École Normale Supérieure, Paris, France
[5]Ingenico, France

Darmstadt - November 26, 2009

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science

Outline

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science

Hash Functions

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science

Hash Functions

- **Symmetric cryptography**:
  Stream ciphers, Block ciphers, Hash functions
- Hash functions:
  - Given a message $\mathcal{M}$ of arbitrary length, a value $\mathcal{H}(\mathcal{M})$ of fixed length ℓ_h is returned
  - Many applications: MAC's (authentification), digital signatures...

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science

Security Requirements of Hash Functions

- Collision resistance:
  - Finding two messages $\mathcal{M}$ and $\mathcal{M}'$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}(\mathcal{M}')$ must be "hard"
- Second preimage resistance:
  - Given a message $\mathcal{M}$ and $\mathcal{H}(\mathcal{M})$, finding another message $\mathcal{M}'$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}(\mathcal{M}')$ must be "hard"
- Preimage resistance:
  - Given a hash $\mathcal{H}$, finding a message $\mathcal{M}$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}$ must be "hard"
- Remark: We never say impossible

Security Requirements of Hash Functions

- Collision resistance:
  - Finding two messages $\mathcal{M}$ and $\mathcal{M}'$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}(\mathcal{M}')$ must be "hard"
- Second preimage resistance:
  - Given a message $\mathcal{M}$ and $\mathcal{H}(\mathcal{M})$, finding another message $\mathcal{M}'$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}(\mathcal{M}')$ must be "hard"
- Preimage resistance:
  - Given a hash $\mathcal{H}$, finding a message $\mathcal{M}$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}$ must be "hard"
- Remark: We never say impossible

Security Requirements of Hash Functions

- Collision resistance:
  - Finding two messages $\mathcal{M}$ and $\mathcal{M}'$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}(\mathcal{M}')$ must be "hard"
- Second preimage resistance:
  - Given a message $\mathcal{M}$ and $\mathcal{H}(\mathcal{M})$, finding another message $\mathcal{M}'$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}(\mathcal{M}')$ must be "hard"
- Preimage resistance:
  - Given a hash $\mathcal{H}$, finding a message $\mathcal{M}$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}$ must be "hard"
- Remark: We never say impossible

Security Requirements of Hash Functions

- Collision resistance:
  - Finding two messages $\mathcal{M}$ and $\mathcal{M}'$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}(\mathcal{M}')$ must be "hard"
- Second preimage resistance:
  - Given a message $\mathcal{M}$ and $\mathcal{H}(\mathcal{M})$, finding another message $\mathcal{M}'$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}(\mathcal{M}')$ must be "hard"
- Preimage resistance:
  - Given a hash $\mathcal{H}$, finding a message $\mathcal{M}$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}$ must be "hard"
- Remark: We never say impossible

Security Requirements of Hash Functions

A strict definition of "hard":

- Collision resistance
  - Generic attack needs $2^{\ell_h/2}$ hash function calls
    $\Rightarrow$ any attack requires at least as many hash function calls as the generic attack.
- Second preimage resistance and preimage resistance
  - Generic attack needs 2^{ℓ_h} hash function calls
    $\Rightarrow$ any attack requires at least as many hash function calls as the generic attack.

SHA-3 Competition [NIST]

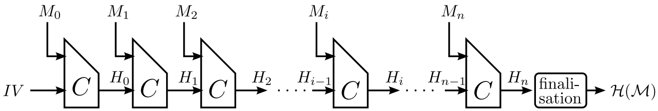
- Attacks against MD5, SHA-1,...
- Confidence in SHA-2 (standard) undermined
- Need of SHA-3: NIST has launched a public competition

SHA-3 Competition - Candidates

- 64 submissions (October 2008)
- 51 first round candidates
  - ESSENCE
- 14 second round candidates (July 2009)

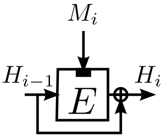
Merkle-Damgård

- Merkle-Damgård is an often used construction
  - Split message $\mathcal{M}$ into message blocks $M_0, M_1, \ldots, M_n$ of fixed size m
  - If M_n is not bit enough extend it to m bits: padding
  - H_i are the intermediate chaining values
  - If the one-way compression function C is collision resistant, then so is the hash function



Davies-Meyer

- Davies-Meyer is a method to construct a secure one-way function from a block cipher E
  - Secure under the "black-box" model (the block cipher has the required randomness properties and the attacker cannot use any special properties or internal details of E)



The ESSENCE Hash Functions

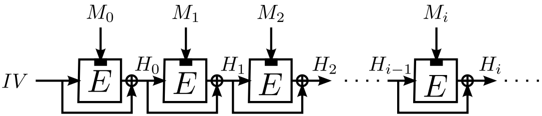
HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science

ESSENCE [Jason W. Martin]

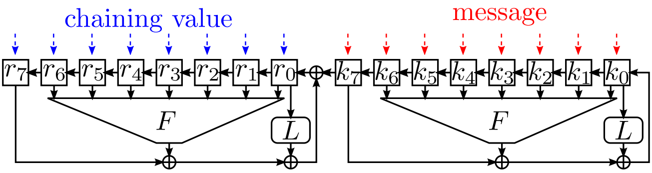
- First round candidate of the SHA-3 competition
- Bases on feedback shift registers
  - over 32-bit words for ESSENCE-256/224
  - over 64-bit words for ESSENCE-512/384
- Message block: 8 words
- Chaining value: 8 words

Structure

- Merkle-Damgård tree
  - A leaf hashes a fixed number of message blocks using MD
  - The inner nodes are combined again by MD
  - The height of the trees depends on a changeable parameter
  - The roots are combined with a final block containing the message length
- Davies-Meyer construction for the compression function



Block Cipher of the Compression Function



$32\times$ clocked

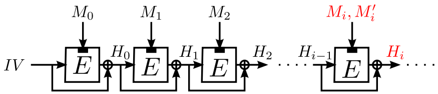
- *F*: bitwise non-linear function
- *L*: linear function on the whole word
- 32 reversible steps

Attack on ESSENCE

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science

Principle

- Collision in compression function
- Using a differential path

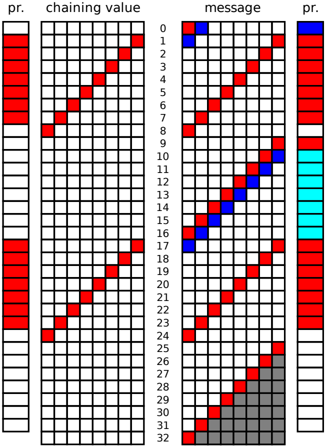


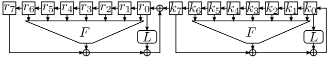
HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science

Differential Path - General

- For iterative structures
- Let Z_i, Z_i' denote the states after i (out of N) iterations starting from Z_0, Z_0'
  - Consider differences $\Delta_i = Z_i \oplus Z_i'$ for $0 \leq i \leq N$
  - Transition from Δ_i to Δ_{i+1} with certain probability
- Finally we want no difference in the chaining value

Differential Path





Differences:

- no difference
- α
- $\beta = L(\alpha)$
- unknown

Probabilities:

- $2^{-|\alpha|}$
- $2^{-|\beta|}$
- $2^{-|\alpha \vee \beta|}$
- 1

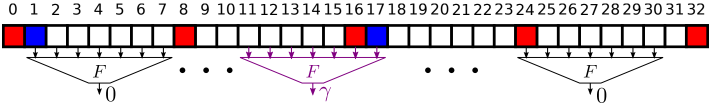
Condition:

$$\alpha \vee \beta \vee L(\beta) = \alpha \vee \beta$$

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science

Exact Complexities

- Probabilities based on Hamming weight (HW) underestimates the real complexity of the attack:
  - e.g. a 1 bit difference has probability $2^{-8.4}$ to pass the 7 steps of F, and not 2^{-7} as we would guess from the HW
- For accurate estimates consider the whole path bitwise
  - Possible differences: $(\alpha_i, \beta_i, \gamma_i)$ with $0 \leq i \leq 32/64$ and $\beta = L(\alpha)$ and $\gamma = L(\beta)$
  - Have to test 2^{30} values for each each $(\alpha_i, \beta_i, \gamma_i)$



Probability of Complete Path - Bitwise

- Bitwise probability, independent of α

$(\alpha_i, \beta_i, \gamma_i)$	(0,0,0)	(0,0,1)	(0,1,0)	(0,1,1)
probability	1	0	$2^{-9.5}$	$2^{-9.1}$
$(\alpha_i, \beta_i, \gamma_i)$	(1,0,0)	(1,0,1)	(1,1,0)	(1,1,1)
probability	$2^{-24.4}$	0	2^{-23}	2^{-26}

- Gives two conditions for α:
  - $\neg\alpha \wedge \neg\beta \wedge \gamma = 0$
  - $\alpha \wedge \neg\beta \wedge \gamma = 0$

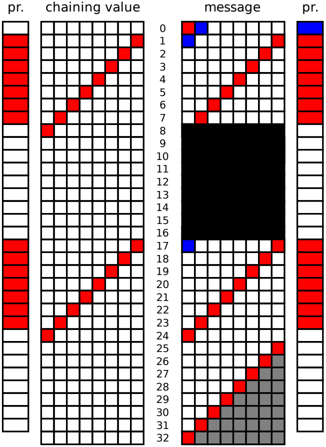
Complexity of Complete Path

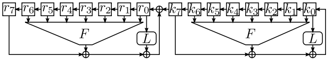
- Complexity for the α's used in our attack:

	differential path left	differential path right	generic method
ESSENCE-256	$2^{67.4}$	$2^{240.6}$	2^{128}
ESSENCE-512	$2^{134.7}$	$2^{478.9}$	2^{256}

- About $2^{15.4}$ pairs pass the whole path for ESSENCE-256 ($2^{37.1}$ for ESSENCE-512)

Idea: Computing the Middle Part





Differences:

- □ no difference
- α
- $\beta = L(\alpha)$
- unknown

Probabilities:

- $2^{-|\alpha|}$
- $2^{-|\beta|}$
- $2^{-|\alpha \vee \beta|}$
- 1

Conditions:

$$\neg\alpha \wedge \neg\beta \wedge \gamma = 0$$
$$\alpha \wedge \neg\beta \wedge \gamma = 0$$

Strategy of the Attack

- Compute many pairs that fulfill the middle part (step 8-17)
- Search among those one message pair that passes the rest of the path (step 0-8 and step 17-32)
- Try different chaining values (random starting messages) with our message pair to find a collision

Computing the Middle Part

8	$x_0 \oplus \alpha$	x_1	x_2	x_3	x_4	x_5	x_6	x_7
9	x_1	x_2	x_3	x_4	x_5	x_6	x_7	$x_8 \oplus \alpha$
10	x_2	x_3	x_4	x_5	x_6	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$
11	x_3	x_4	x_5	x_6	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}
12	x_4	x_5	x_6	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}	x_{11}
13	x_5	x_6	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}	x_{11}	x_{12}
14	x_6	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}	x_{11}	x_{12}	x_{13}
15	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}
16	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
17	$x_9 \oplus \beta$	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	$x_{16} \oplus \alpha$

- Let ℓ be the word size (32 or 64), $\beta = L(\alpha)$, $\gamma = L(\beta)$, $s = |\alpha \vee \beta|$ and $\mathcal{S} = \{i : \alpha_i \vee \beta_i = 1\}$

Computing the Middle Part - Bit Level

- For all bit-difference $(\alpha_i, \beta_i, \gamma_i)$, $0 \leq i < 32/64$:
  - Store bit-tuples $(x_1, \ldots, x_{15})_i$ passing F in the middle part:

    $e.g.: F(x_2, x_3, x_4, x_5, x_6, x_7, x_8\ )_i = F(x_2, x_3, x_4, x_5, x_6, x_7, x_8 \oplus \alpha)_i$
  - Better: Store only those tuples which have a possibility to pass the rest of the path
- Number of tuples depending having the bit-differences:

$(\alpha_i, \beta_i, \gamma_i)$	$(0,0,1)$	$(0,1,0)$	$(0,1,1)$	$(1,0,0)$	$(1,0,1)$	$(1,1,0)$	$(1,1,1)$
	0	96	128	96	120	96	176
better	0	96	128	2	0	4	2

- Number of possibilities to choose $(x_1, \ldots, x_{15})_i$, $i \in \mathcal{S}$:

$$N_\alpha = 2^{|\alpha \wedge \neg\beta \wedge \neg\gamma|} \times 4^{|\alpha \wedge \beta \wedge \neg\gamma|} \times 96^{|\neg\alpha \wedge \beta \wedge \neg\gamma|} \times 2^{|\alpha \wedge \beta \wedge \gamma|} \times 128^{|\neg\alpha \wedge \beta \wedge \gamma|}$$

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science

Computing the Middle Part - Bit Level

- For all bit-difference $(\alpha_i, \beta_i, \gamma_i)$, $0 \leq i < 32/64$:
  - Store bit-tuples $(x_1, \ldots, x_{15})_i$ passing F in the middle part:

    $$e.g.: F(x_2, x_3, x_4, x_5, x_6, x_7, x_8\ )_i = F(x_2, x_3, x_4, x_5, x_6, x_7, x_8 \oplus \alpha)_i$$

  - Better: Store only those tuples which have a possibility to pass the rest of the path
- Number of tuples depending having the bit-differences:

$(\alpha_i, \beta_i, \gamma_i)$	(0,0,1)	(0,1,0)	(0,1,1)	(1,0,0)	(1,0,1)	(1,1,0)	(1,1,1)
	0	96	128	96	120	96	176
better	0	96	128	2	0	4	2

- Number of possibilities to choose $(x_1, \ldots, x_{15})_i$, $i \in \mathcal{S}$:

$$N_\alpha = 2^{|\alpha \wedge \neg\beta \wedge \neg\gamma|} \times 4^{|\alpha \wedge \beta \wedge \neg\gamma|} \times 96^{|\neg\alpha \wedge \beta \wedge \neg\gamma|} \times 2^{|\alpha \wedge \beta \wedge \gamma|} \times 128^{|\neg\alpha \wedge \beta \wedge \gamma|}$$

Computing the Middle Part - Fix s Bits

$$L(\overbrace{x_7}^{s \text{ bits fixed}}) = x_0 \oplus \overbrace{x_8 \oplus F(x_1, x_2, x_3, x_4, x_5, x_6, x_7)}^{s \text{ bits fixed}}$$

$$L(\overbrace{x_8}^{s \text{ bits fixed}}) = \overbrace{x_1 \oplus x_9 \oplus F(x_2, x_3, x_4, x_5, x_6, x_7, x_8 \oplus \alpha)}^{s \text{ bits fixed}}$$

$$L(\overbrace{x_9}^{s \text{ bits fixed}}) = \overbrace{x_2 \oplus x_{10} \oplus F(x_3, x_4, x_5, x_6, x_7, x_8 \oplus \alpha, x_9 \oplus \beta) \oplus \gamma}^{s \text{ bits fixed}}$$

$$L(\overbrace{x_{10}}^{s \text{ bits fixed}}) = \overbrace{x_3 \oplus x_{11} \oplus F(x_4, x_5, x_6, x_7, x_8 \oplus \alpha, x_9 \oplus \beta, x_{10})}^{s \text{ bits fixed}}$$

$$\ldots$$

$$L(\overbrace{x_{14}}^{s \text{ bits fixed}}) = \overbrace{x_7 \oplus x_{15} \oplus F(x_8 \oplus \alpha, x_9 \oplus \beta, x_{10}, x_{11}, x_{12}, x_{13}, x_{14})}^{s \text{ bits fixed}}$$

$$L(\overbrace{x_{15}}^{s \text{ bits fixed}}) = x_{16} \oplus \overbrace{x_8 \oplus F(x_9 \oplus \beta, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15})}^{s \text{ bits fixed}}$$

Computing the Middle Part - Linear Systems

- We have 7 linear systems depending on α, $8 \leq j \leq 14$

$$L(x_j) = R_j$$

- x_j and R_j have together
  - 2ℓ bits (ℓ is the word length)
  - $2s$ bit fixed
- L gives ℓ equations
- Probability of a solution $2^{-(2s-\ell)}$ if the system has full rank

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science

Computing the Middle Part - Solving the Systems

- The position of the fixed bits is given by $\mathcal{S}$
- Using Gauss elimination we find $2s - \ell$ equation which must be satisfied to have a solution
- Order the $7(2s - \ell)$ equations depending on the variables they contain, so that changing the variables in the later equations has no influence on the results of the first ones

Computing the Middle Part - Finishing

- After solving the linear systems we have
  - In x_j,R_j all bit fixed, $8 \leq j \leq 14$
  - In $x_1, \ldots, x_7, x_{15}$ we have s bit fixed
  - In x_0, x_{16} no bit fixed
- Selecting the $\ell - s$ free bits of x_7 allows us to determine all the other free bits
  $\Rightarrow$ For each solution of the linear systems we have $2^{\ell - s}$ solution for the middle part for free
- In average, we find a solution for $x_0, \ldots, x_{16}$ in less than one call to the compression function

Final Complexity

- To find the optimal α
  - ESSENCE-256: Test all possible α
  - ESSENCE-512: Test all α's with HW ≤ 8 (limitation on the left side)

	differential path		generic method
	left	right	
ESSENCE-256	$2^{67.4}$	$2^{62.2}$	2^{128}
ESSENCE-512	$2^{134.7}$	$2^{116.1}$	2^{256}

Semi-Free-Start Collision on 29 rounds

	Initial values for *r*									Initial values for *k*								
	B0741769	BA2BA1A1	349A4DC8	54204D82	292006B1	80096194	D23020E1	9098A7EA		4CD35806	4759FB6D	3ED267E5	17641536	BE1F35ED	688B0C3C	DF126549	5FAE0827	
round	differences								round	differences								round
0	0	0	0	0	0	0	0	0	0	80102040	537874EB	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	80102040	1	537874EB	0	0	0	0	0	0	80102040	1
2	0	0	0	0	0	0	80102040	0	2	0	0	0	0	0	0	80102040	0	2
3	0	0	0	0	0	80102040	0	0	3	0	0	0	0	0	80102040	0	0	3
4	0	0	0	0	80102040	0	0	0	4	0	0	0	0	80102040	0	0	0	4
5	0	0	0	80102040	0	0	0	0	5	0	0	0	80102040	0	0	0	0	5
6	0	0	80102040	0	0	0	0	0	6	0	0	80102040	0	0	0	0	0	6
7	0	80102040	0	0	0	0	0	0	7	0	80102040	0	0	0	0	0	0	7
8	80102040	0	0	0	0	0	0	0	8	80102040	0	0	0	0	0	0	0	8
9	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	80102040	9
10	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	80102040	537874EB	10
11	0	0	0	0	0	0	0	0	11	0	0	0	0	0	80102040	537874EB	0	11
12	0	0	0	0	0	0	0	0	12	0	0	0	0	80102040	537874EB	0	0	12
13	0	0	0	0	0	0	0	0	13	0	0	0	80102040	537874EB	0	0	0	13
14	0	0	0	0	0	0	0	0	14	0	0	80102040	537874EB	0	0	0	0	14
15	0	0	0	0	0	0	0	0	15	0	80102040	537874EB	0	0	0	0	0	15
16	0	0	0	0	0	0	0	0	16	80102040	537874EB	0	0	0	0	0	0	16
17	0	0	0	0	0	0	0	80102040	17	537874EB	0	0	0	0	0	0	80102040	17
18	0	0	0	0	0	0	80102040	0	18	0	0	0	0	0	0	80102040	0	18
19	0	0	0	0	0	80102040	0	0	19	0	0	0	0	0	80102040	0	0	19
20	0	0	0	0	80102040	0	0	0	20	0	0	0	0	80102040	0	0	0	20
21	0	0	0	80102040	0	0	0	0	21	0	0	0	80102040	0	0	0	0	21
22	0	0	80102040	0	0	0	0	0	22	0	0	80102040	0	0	0	0	80000040	22
23	0	80102040	0	0	0	0	0	0	23	0	80102040	0	0	0	0	80000040	38C32419	23
24	80102040	0	0	0	0	0	0	0	24	80102040	0	0	0	0	80000040	38C32419	3B50EAEF	24
25	0	0	0	0	0	0	0	0	25	0	0	0	0	80000040	38C32419	3B50EAEF	E9F738F8	25
26	0	0	0	0	0	0	0	0	26	0	0	0	80000040	38C32419	3B50EAEF	E9F738F8	D59E6BC4	26
27	0	0	0	0	0	0	0	0	27	0	0	80000040	38C32419	3B50EAEF	E9F738F8	D59E6BC4	519ECD90	27
28	0	0	0	0	0	0	0	0	28	0	80000040	38C32419	3B50EAEF	E9F738F8	D59E6BC4	519ECD90	8199374F	28
29	0	0	0	0	0	0	0	0	29	80000040	38C32419	3B50EAEF	E9F738F8	D59E6BC4	519ECD90	8199374F	1B9B997C	29
30	0	0	0	0	0	0	0	80000040	30	38C32419	3B50EAEF	E9F738F8	D59E6BC4	519ECD90	8199374F	1B9B997C	A7EF91F9	30
31	0	0	0	0	0	0	80000040	102040	31	3B50EAEF	E9F738F8	D59E6BC4	519ECD90	8199374F	1B9B997C	A7EF91F9	21E1C70	31
32	0	0	0	0	0	80000040	102040	3336DACE	32	E9F738F8	D59E6BC4	519ECD90	8199374F	1B9B997C	A7EF91F9	21E1C70	1B715D5F	32

Conclusion

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science

Conclusion

- **Complexity:**
  - ESSENCE-*256*: $2^{67.4}$
  - ESSENCE-*512*: $2^{134.7}$
- **Why does the attack work?**
  - Message precessing is independent of chaining value
  - Precompute low probability part
  - Efficient solving of linear system
  - Exact probability by considering the bit path
  - Reduced cost by considering the whole path

Conclusion

- **Complexity:**
  - ESSENCE-*256*: $2^{67.4}$
  - ESSENCE-*512*: $2^{134.7}$
- **Why does the attack work?**
  - Message precessing is independent of chaining value
  - Precompute low probability part
  - Efficient solving of linear system
  - Exact probability by considering the bit path
  - Reduced cost by considering the whole path

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Information and Computer Science