Improved Differential Attacks for ECHO and Grøstl

(extended version available on eprint)

Thomas Peyrin

CRYPTO 2010

Santa Barbara - November 19, 2010

ingenico
beyond payment

Outline

Introduction

ECHO (Benadjila et al.)

Grøstl (Gauravaram et al.)

Results and future works

Outline

Introduction

ECHO (Benadjila et al.)

Grøstl (Gauravaram et al.)

Results and future works

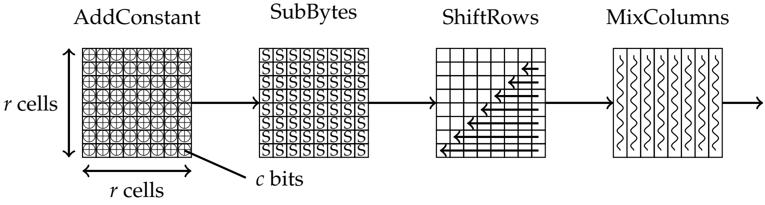
SHA-3 competition

The SHA-3 hash function competition:

- started in October 2008, 64 submissions
- 51 candidates accepted for the first round
- 14 semi-finalists selected in 2009
- finalists to be selected end 2010
- winner to be announced in 2012

Among the 14 semi-finalists, one can identify 4 AES-based candidates. For example ECHO and Grøstl.

What is an AES-like permutation ?



$$MixColumns \circ ShiftRows \circ SubBytes \circ AddConstant(C)$$

- **AddConstant:** in known-key model, just add a round-dependent constant (breaks natural symmetry of the three other functions)
- **SubBytes:** application of a c-bit Sbox (only non-linear part)
- **ShiftRows:** rotate column position of all cells in a row, according to its row position
- **MixColumns:** linear diffusion layer.

Hash function collision attacks

In general, there are **two basic tools** in order to find a collision: the differential path building technique and the freedom degree utilization method.

The differential path building techniques (for `SHA-1`):

- local collisions
- linear perturbation mask
- non-linear parts

The freedom degree utilization methods (for `SHA-1`):

- neutral bits
- message modifications
- boomerang trails

Hash function collision attacks

In general, there are **two basic tools** in order to find a collision: the differential path building technique and the freedom degree utilization method.

The differential path building techniques (for AES-based):

- truncated differential paths

The freedom degree utilization methods (for AES-based):

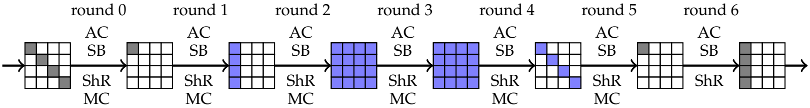
- rebound attacks
- multiple-inbound attacks
- start-from-the-middle attacks
- super-Sbox attacks

In this talk, we will mostly focus on how to find good differential paths for ECHO and Grøstl

The Super-Sbox method

In general, the **Super-Sbox method** seem to be more powerful than classical rebound or start-from-the-middle attacks.

It allows to control 3 rounds in the middle (controlled rounds): a valid pair can be found with one operation on average and a minimal cost of $2^{r \cdot c}$.



The rest is fulfilled probabilistically (uncontrolled rounds). In our example, we have twice a probability $2^{-8\times 3} = 2^{-24}$ to pay.

Outline

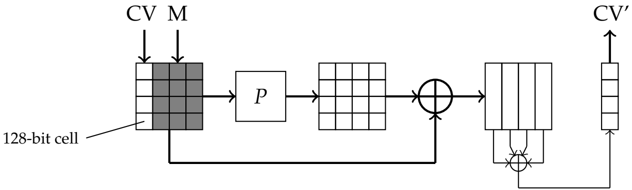
Introduction

ECHO (Benadjila et al.)

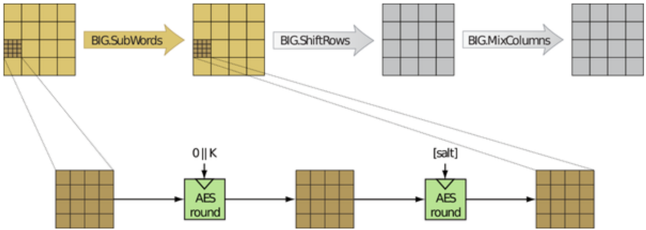
Grøstl (Gauravaram et al.)

Results and future works

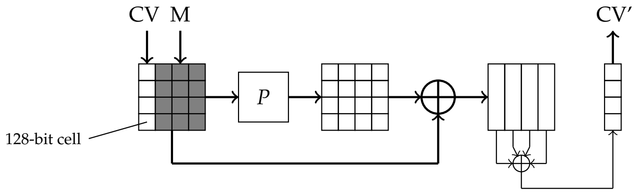
ECHO compression function



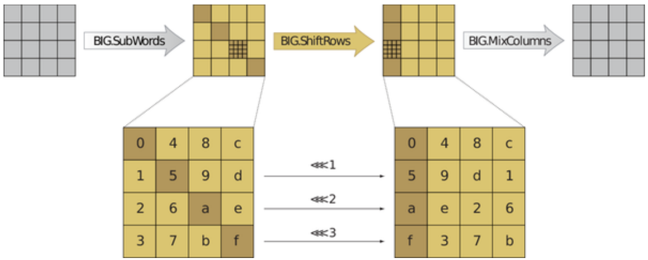
One round of the internal permutation P



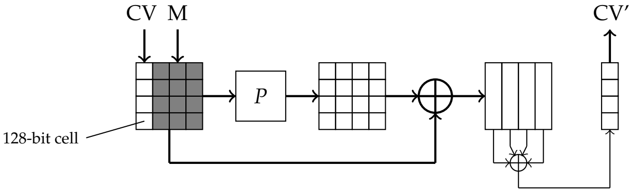
ECHO compression function



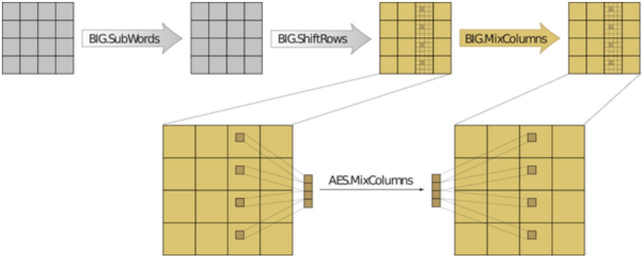
One round of the internal permutation P



ECHO compression function

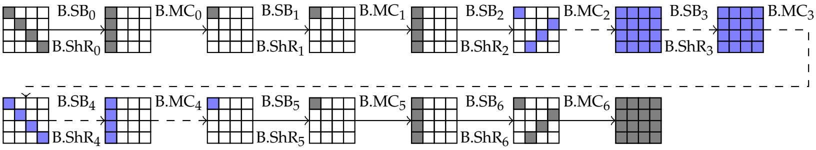


One round of the internal permutation P



Previous attacks

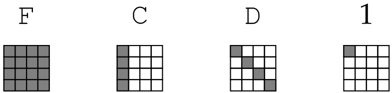
Previous attacks focused on the internal permutation only, because the complexities were already very high.



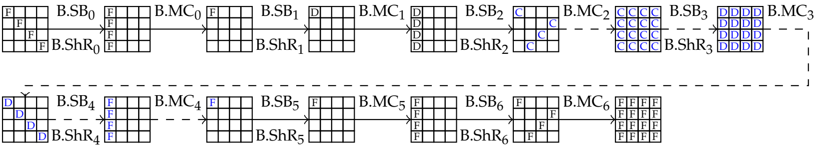
For this 7-round trail, one can find a valid pair with $2^{128\times 3} = 2^{384}$ computations on average ... but with a minimal cost of 2^{512} because of the super-Sbox method.

Improved differential paths for ECHO

Increase the granularity of the path:



Force all intra-word differences to be of the same type



Problem: this path has an average complexity of 2^{96} comp. per solution, but we still have to pay the huge 2^{512} minimal cost of the Super-Sbox method anyway.

Idea: improve the Super-Sbox technique for this particular differential path: 2^{32} comp. and memory for one solution in the controlled round.

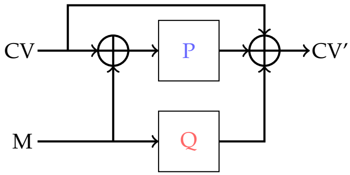
Results for ECHO

target	rounds	computational complexity	memory requirements	type
ECHO-256 comp. function	3/8	2^{64}	2^{32}	free-start collision
	3/8	2^{96}	2^{32}	semi-free-start collision*
	4.5/8	2^{96}	2^{32}	distinguisher
ECHO-512 comp. function	3/10	2^{96}	2^{32}	(semi)-free-start collision*
	6.5/10	2^{96}	2^{32}	distinguisher
ECHO-SP-256 comp. function	3/8	2^{64}	2^{32}	(semi)-free-start collision
	3/8	2^{64}	2^{32}	distinguisher
ECHO-SP-512 comp. function	3/10	2^{64}	2^{32}	free-start collision
	3/10	2^{96}	2^{32}	semi-free-start collision*
	4.5/10	2^{96}	2^{32}	distinguisher

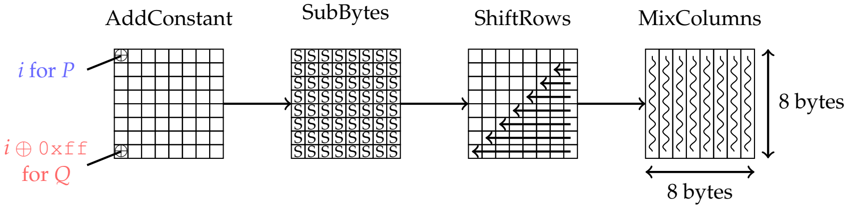
* because of a lack of freedom degrees, these attacks requires some randomization on the salt. Thus they are applicable in the chosen-salt setting only

Outline

Grøstl compression function



Round i of permutations P and Q:

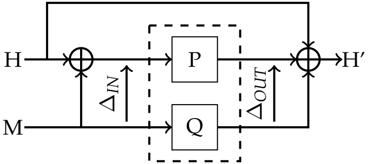


$$MixColumns \circ ShiftRows \circ SubBytes \circ AddConstant(C)$$

The internal differential attack

Problem: all previous attacks build classical differential paths for the permutation P and Q (allows to reach 8/10 rounds)

Idea: **look at the difference between the two parallel branches**
It works well on `Grøstl` because P and Q are almost identical (only the constant addition differs)



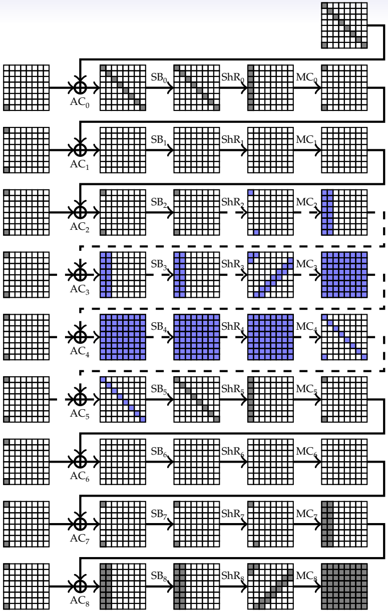
attacked primitive

Let A and B be s.t. $A \oplus B = \Delta_{IN}$ and $Q(A) \oplus P(B) = \Delta_{OUT}$

We have $h(H, M) = \Delta_{IN} \oplus \Delta_{OUT}$

What can we do with such a pair A and B ?

- **Distinguishing attack:**
  - assume Δ_{IN} is maintained in a set of x elements
  - assume Δ_{OUT} is maintained in a set of y elements
  - thus $h(H, M)$ is maintained in a set of $k = x \cdot y$ elements
  - we can distinguish the Grøstl compression function from an ideal one: such pair (H, M) can be generically obtained with $2^n/k$ computations
  - one can also distinguish the permutations P and Q from ideal permutations (see “limited birthday distinguishers” in [Gilbert Peyrin FSE 2010])
- **Collision attack:**
  - because of a lack of freedom degrees, no improvement for the compression function attacks
  - but we can attack 5/10 rounds of the hash function



An example with 9 rounds:

- we have
  - $x = 2^{56}$
  - $y = 2^{128}$
  - $k = 2^{184}$
- thus the generic complexity is $2^{512-184} = 2^{328}$ operations
- we can find a valid candidate with only 2^{80} computations and 2^{64} memory
- the amount of freedom degrees only allows us to compute one such candidate, but generalization of the internal differential attack gives additional freedom degrees

Results for Grøstl

target	rounds	computational complexity	memory requirements	type	section
Grøstl-256 internal perm.	9/10	2^{80}	2^{64}	distinguisher	**new**
	10/10	2^{192}	2^{64}	distinguisher	**new**
Grøstl-512 internal perm.	11/14	2^{640}	2^{64}	distinguisher	**new**
Grøstl-256 comp. function	8/10	2^{112}	2^{64}	distinguisher	[Gilbert Peyrin 2009]
	9/10	2^{80}	2^{64}	distinguisher*	**new**
	10/10	2^{192}	2^{64}	distinguisher*	**new**
Grøstl-512 comp. function	11/14	2^{640}	2^{64}	distinguisher*	**new**
Grøstl-256 hash function	4/10	2^{64}	2^{64}	collision	[Mendel et al. 2010]
	5/10	2^{79}	2^{64}	collision	**new**
Grøstl-512 hash function	5/14	2^{176}	2^{64}	collision	[Mendel et al. 2010]
	6/14	2^{177}	2^{64}	collision	**new**

* for these distinguishers, the amount of available freedom degrees allows us to generate only one valid candidate with good probability

Outline

Results and future works

Our results:

- first attacks on reduced versions of the ECHO compression function
- distinguishing attack against full Grøstl-256 compression function or internal permutations

Future works:

- find better differential paths for ECHO ([Schläffer - SAC 2010])
- derive collision attacks for the Grøstl hash function with internal differential paths ([Ideguchi et al. - eprint 2010])
- try to apply internal differential attack to other schemes

Be careful when designing a scheme:
also check the differential paths between the internal branches