The Super-Sbox Cryptanalysis

Thomas Peyrin

CCRG seminar - Nanyang Technological University

Singapore - October 26, 2010
Outline

Introduction

The Super-Sbox attack

A case study: Grøstl (Gauravaram et al.)

Results and future works
Introduction

The Super-Sbox attack

A case study: Grøstl (Gauravaram et al.)

Results and future works
What is a Hash Function?

- $H$ maps an **arbitrary length input** (the message $M$) to a **fixed length output** (typically $n = 128$, $n = 160$ or $n = 256$).
- no secret parameter.
- $H$ must be easy to compute.
The security goals

- **pre-image resistance**: given an output challenge $y$, the attacker cannot find a message $x$ such that $H(x) = y$, in less than $\theta(2^n)$ operations.

- **2nd pre-image resistance**: given a challenge $(x, y)$ so that $H(x) = y$, the attacker cannot find a message $x' \neq x$ such that $H(x') = y$, in less than $\theta(2^n)$ operations.

- **collision resistance**: the attacker cannot find two messages $(x, x')$ such that $H(x) = H(x')$, in less than $\theta(2^{n/2})$ operations (a generic attack with the birthday paradox exists [Yuval-79]).
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
- 4/5/6 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?

- **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.

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

Introduction

The Super-Sbox attack

A case study: Grøstl (Gauravaram et al.)

Results and future works
Truncated differences

- Originally introduced by Knudsen for block ciphers [Knudsen FSE 1994]
- Later applied to hash functions (collision attack on Grindahl) [Peyrin ASIACRYPT 2007]
- **Idea:** consider byte-differences, without considering their actual value (active or inactive).
- Only the truncated differences propagation through MixColumns behave probabilistically. Per column:
  \[
  \text{nb active input cells} + \text{nb active output cells} \geq r + 1.
  \]
  \[
  P \approx 2^{-xc} \quad \text{for } x \neq r \text{ inactive output cells}.
  \]
Controlled and uncontrolled rounds

- **Idea:** use the freedom degrees in the middle of the differential path.

- The path is divided into two different kind of steps:

  - **The controlled rounds:** the part where the freedom degrees are used (usually in the middle of the path). On average, finding a solution for the controlled rounds should cost only a few operations.

  - **The uncontrolled rounds:** the part where all the events are verified probabilistically (left and right part of the path) because no more freedom degree is available. Determine the complexity of the overall attack.
**Rebound Attack and Start-from-the-middle**

- **Rebound attack:** allows to get 2 controlled rounds [Mendel et al. FSE 2009]. Requires $2^{rc}$ memory. It broke compression functions of many SHA-3 candidates.

- **Start-from-the-middle:** use more complicated techniques to get up to 3 controlled rounds in the case of low weight differential paths [Mendel et al. SAC 2009]. Requires $2^{rc}$ memory.
The Super-Sbox view

- Introduced by Daemen and Rijmen (e.g. [Daemen Rijmen SCN 2006]) to simplify the analysis of AES differential properties and not for cryptanalysis purposes.

- **Idea:** one can view two rounds of an AES-like permutation as a layer of big $2^{rc}$-bit Sboxes preceded and followed by simple affine transformations. We call those **Super-Sboxes**
The controlled rounds in the Super-Sbox view

- **One can get 3 controlled rounds**, even for high weight differential paths.

- **Forward**: start with a random (not truncated) difference $\delta_\text{start}'$ at the beginning of round 2 (such that we obtain a compatible truncated difference $\Delta_\text{start}$ when inverting SB and AC). Then, pass ShR, MC, AC and ShR to obtain the aimed input difference $\Delta_\text{in}$ on the $r$ Super-Sboxes.

- **Backward**: start with a random (not truncated) difference $\Delta_\text{end}$ at the end of round 4, and invert MC and ShR in order to obtain the aimed output difference $\Delta_\text{out}$ on the $r$ Super-Sboxes.

- **Problem**: need the ability to find for each of the $r$ columns, a value that maps $\Delta_\text{in}$ to $\Delta_\text{out}$. ... seems hard.
The controlled rounds

- **Idea:** pay a big price ($2^{rc}$ operations and memory), but get many solutions ($2^{rc}$) once you paid.

- **1st step:** Fix a random $\Delta_{\text{start}}'$ difference value, which gives a fixed random $\Delta_{\text{in}}$. For each of the $r$ Super-Sboxes, exhaust all $2^{rc}$ possible actual values, then sort the results in $r$ tables according to the output difference obtained.

- **2nd step:** try $2^{rc}$ distinct $\Delta_{\text{end}}$ differences. Then, for each $\Delta_{\text{out}}$ obtained by computing backward, check if for all the $r$ columns the appropriate $2^{rc}$-bit difference is present in the corresponding table. On average, one solution is found per $\Delta_{\text{end}}$ try.

- **The average complexity for finding one internal state pair verifying the controlled rounds is 1.**
The uncontrolled rounds

8-round path:

- On the left side, one has one $4 \leftrightarrow 1$ MixColumns transition to control (round 1): $P \approx 2^{-(r-1)c}$

- On the right side, one has one $4 \leftrightarrow 1$ MixColumns transition to control (round 5): $P \approx 2^{-(r-1)c}$

- Total complexity for finding a solution for the whole path: $2^{2(r-1)c}$ operations.

One has also to check that we have enough freedom degrees, such that a valid pair can be found.
Limited-birthday distinguishers

What is the generic complexity for mapping $i$ fixed-difference bits to $j$ fixed-difference bits through a random permutation $E$?

Wlog, assume that $i \geq j$ and let $n := r^2c$. Due to the birthday paradox, each structure of $2^{n-i}$ input values obtained by fixing the value of the $i$ fixed-difference bits allows to get fixed-difference on $2(n-i)$ output bits:

- if $j \leq 2(n-i)$, then one can select $2^{j/2}$ input values from one single structure and this suffices to achieve a collision on the $j$ target positions. The attack complexity is about $2^{j/2}$.
- if $j > 2(n-i)$, then about $2^{j-2(n-i)}$ structures have to be used to obtain a collision on the $j$ prescribed positions. Overall, the complexity of the attack is about $2^{n-i} \times 2^{j-2(n-i)} = 2^{i+j-n}$.

Same reasoning for the $n-j$ free difference bits on the output and attacking $E^{-1}$:

- if $i \leq 2(n-j)$, then the attack complexity is about $2^{i/2}$.
- if $i > 2(n-j)$, then the attack complexity is about $2^{i+j-n}$.

**Final complexity:** $\max\{2^{j/2}, 2^{i+j-n}\}$. 
Results on AES and Grøstl

Table: Results on the underlying permutation

<table>
<thead>
<tr>
<th>target</th>
<th>rounds</th>
<th>computational complexity</th>
<th>memory requirements</th>
<th>type</th>
<th>source</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES</td>
<td>7</td>
<td>$2^{24}$</td>
<td>$2^{16}$</td>
<td>known-key-dist.</td>
<td>[Mendel et al. SAC 2009]</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>$2^{48}$</td>
<td>$2^{32}$</td>
<td>known-key-dist.</td>
<td>[Gilbert Peyrin FSE 2010]</td>
</tr>
<tr>
<td>Grøstl-256 permutation</td>
<td>7</td>
<td>$2^{56}$</td>
<td>$2^{64}$</td>
<td>distinguisher</td>
<td>[Mendel et al. SAC 2009]</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>$2^{112}$</td>
<td></td>
<td>distinguisher</td>
<td>[Gilbert Peyrin FSE 2010]</td>
</tr>
</tbody>
</table>

Table: Results on the compression function

<table>
<thead>
<tr>
<th>target</th>
<th>rounds</th>
<th>computational complexity</th>
<th>memory requirements</th>
<th>type</th>
<th>source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grøstl-256 compression function</td>
<td>6</td>
<td>$2^{120}$</td>
<td>$2^{64}$</td>
<td>semi-free-start coll.</td>
<td>[Mendel et al. FSE 2009]</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>$2^{64}$</td>
<td></td>
<td>semi-free-start coll.</td>
<td>[Mendel et al. SAC 2009]</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>$2^{120}$</td>
<td></td>
<td>semi-free-start coll.</td>
<td>[Gilbert Peyrin FSE 2010]</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>$2^{56}$</td>
<td>$2^{64}$</td>
<td>distinguisher</td>
<td>[Mendel et al. SAC 2009]</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>$2^{112}$</td>
<td></td>
<td>distinguisher</td>
<td>[Gilbert Peyrin FSE 2010]</td>
</tr>
</tbody>
</table>

* Results also independently obtained by Lamberger et al.
Outline

Introduction

The Super-Sbox attack

A case study: Grøstl (Gauravaram et al.)

Results and future works
Grøstl compression function

Round $i$ of permutations $P$ and $Q$:

AddConstant $\oplus$ SubtractBytes $\circ$ ShiftRows $\circ$ MixColumns

$\text{MixColumns} \circ \text{ShiftRows} \circ \text{SubBytes} \circ \text{AddConstant}(C)$
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)

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 $\Delta_{IN}$ is maintained in a set of $x$ elements
- assume $\Delta_{OUT}$ 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 (with “limited birthday distinguishers”)

**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

<table>
<thead>
<tr>
<th>target</th>
<th>rounds</th>
<th>computational complexity</th>
<th>memory requirements</th>
<th>type</th>
<th>section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grøstl-256</td>
<td>7/10</td>
<td>$2^{56}$</td>
<td>$2^{64}$</td>
<td>distinguisher</td>
<td>[Mendel et al. SAC 2009]</td>
</tr>
<tr>
<td></td>
<td>8/10</td>
<td>$2^{112}$</td>
<td></td>
<td>distinguisher</td>
<td>[Gilbert Peyrin FSE 2010]</td>
</tr>
<tr>
<td></td>
<td>9/10</td>
<td>$2^{80}$</td>
<td>$2^{64}$</td>
<td>distinguisher*</td>
<td>[Peyrin CRYPTO 2010]</td>
</tr>
<tr>
<td></td>
<td>10/10</td>
<td>$2^{192}$</td>
<td></td>
<td>distinguisher*</td>
<td>[Peyrin CRYPTO 2010]</td>
</tr>
<tr>
<td>Grøstl-512</td>
<td>11/14</td>
<td>$2^{640}$</td>
<td>$2^{64}$</td>
<td>distinguisher*</td>
<td>[Peyrin CRYPTO 2010]</td>
</tr>
<tr>
<td>Grøstl-256</td>
<td>4/10</td>
<td>$2^{64}$</td>
<td>$2^{64}$</td>
<td>collision</td>
<td>[Mendel et al. SAC 2010]</td>
</tr>
<tr>
<td>hash function</td>
<td>5/10</td>
<td>$2^{79}$</td>
<td>$2^{64}$</td>
<td>collision</td>
<td>[Peyrin CRYPTO 2010]</td>
</tr>
<tr>
<td>Grøstl-512</td>
<td>5/14</td>
<td>$2^{176}$</td>
<td>$2^{64}$</td>
<td>collision</td>
<td>[Mendel et al. SAC 2010]</td>
</tr>
<tr>
<td>hash function</td>
<td>6/14</td>
<td>$2^{177}$</td>
<td>$2^{64}$</td>
<td>collision</td>
<td>[Peyrin CRYPTO 2010]</td>
</tr>
</tbody>
</table>

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

**Be careful when designing a scheme:**

also check the differential paths **between** the internal branches
Introduction

The Super-Sbox attack

A case study: Grøstl (Gauravaram et al.)

Results and future works
Results and future works

The Super-Sbox method:

- a very easy-to-use yet powerful cryptanalysis tool
- provides the best attack against 128-bit AES in the known key model
- also very efficient against AES-based hash functions: ECHO, Grøstl, ... In particular, first distinguishing attack against full Grøstl-256 compression function or internal permutations

Future works:

- find better differential paths for ECHO ([Sasaki et al. - ASIACRYPT 2010] [Schläffer - SAC 2010])
- derive collision attacks for the Grøstl hash function with internal differential paths ([Ideguchi et al. - eprint 2010])
- try to apply Super-Sbox attack to other schemes (work on SHA\textit{v}ite-3 to be published soon)
- switching attack: switch completely the type of differential path considered between the left and the right controlled rounds and use the Super-Sbox setting in order to link them