Preimages for Reduced SHA-0 and SHA-1

Christophe De Cannière
and
Christian Rechberger

Presented by Stav Hertz
Agenda

Background

**Iteration 1 (High level)**
- Inverting the compression function
- From $P_3$ to preimage
- Putting everything together

**Iteration 2 (Low level)**
- Fixing the columns
- Preimage method conclusions
- Outlook
Background
Recollection

- Preimage
- Second preimage
- SHA
Previous research

- All currently known generic preimage attacks require either:
  - Impractically long first preimages
  - A first preimage lying in a very small subset of the set of all possible preimages
  - A target digest constructed in a very special way
Our results

- SHA-0
  - 37 rounds – $2^{75}$
  - 49 rounds – $2^{159}$

- SHA-1
  - 34 rounds – $2^{80}$
  - 44 rounds – $2^{157}$
Iteration 1

- Inverting the compression function
- From P3 to preimage
- Putting everything together
Inverting the compression function

• Goal:
  • Find a message that transforms a given IV to a given result of the compression function

• Different(ial) approaches:
  • For second preimage - Reuse the differential characteristics used in collision attacks
  • Compute the hash value of a related message and then steer the result towards the target value
Used method

- Changing the representation and tweaking the states

**Fig. 5.** Bits affected by a single bit flip at the input (SHA-1). Black bits are guaranteed to flip; gray bits may be flipped; white bits are unaffected.
Defining Partial-Pseudo-Preimage

Pseudo → Partially controlled input

Partial → Partially matching output

and
P3 to preimage

- **Goal:**
  - Transform the attack on the compression function that gives a P3 to an attack on the hash function leading to the preimage

- **Different approaches:**
  - Meet in the middle
  - Layered Tree method
  - Alternative Backward-Forward Tree
Used method

- **P3 graph**
  - Nodes: \((h(i), m(i))\)
  - Edges: Mapping between \(h(i)\) and \(f(h(i), m(i))\)
  - First message block – forward direction
  - Last message block – backward direction

- **Finding the preimage**
  - Finding a connection (a path) between the entry node and the exit node in the graph
Putting everything together

- **Goal:**
  - Combine the two methods to receive a correctly padded message

- **Different approaches:**
  - Restrict the degrees of freedom in the compression function attack to receive correct length
  - Construct expandable messages using:
    - Multicollisions
    - Flexibility of the P3 graph method
Visual results

<table>
<thead>
<tr>
<th>SHA-O</th>
<th>SHA-1</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="SHA-O" alt="Graph" /></td>
<td><img src="SHA-1" alt="Graph" /></td>
</tr>
</tbody>
</table>

Legend:
- Plain
- MITM 2
- P^3 graph
Iteration 2

- Fixing the columns
- Preimage method conclusions
- Outlook
Fixing the columns

Old representation vs. New representation

Goal: Zeroing the E words

Bit flip observation

Stage 1 & Stage 2 – Number of free bits
Preimage method conclusions

- No structure imposed
- Precomputation
- The effort for every additional preimage attack is $2^{b+c}$
### Outlook

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Preimage</th>
<th>Collision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step-reduced variants (SHA-1)</td>
<td>45 Steps</td>
<td>58 Steps</td>
</tr>
<tr>
<td>Degrees of freedom</td>
<td>Not all degrees of freedom are used</td>
<td>Limiting factor</td>
</tr>
<tr>
<td>Sensitivity for different choices of rotation constants</td>
<td>Strong dependency to constants</td>
<td>Not such a strong dependency as used to be</td>
</tr>
</tbody>
</table>
Summary

- Inverting the compression function
- P3 graphs for hash preimages
- Dealing with padding
- Results and outlook
Questions?
Thank you