Chosen-Prefix Collision Attacks on SHA-1

Thomas Peyrin

(joint work with Gaëtan Leurent)

NTU - Singapore

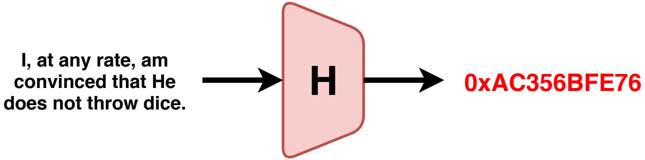
Kangacrypt 2018

Adelaide - December 7, 2018

NANYANG TECHNOLOGICAL UNIVERSITY

Outline

What is a Hash Function ?



- *H* maps an **arbitrary length input** (the message *M*) to a **fixed length output**. Typically $n = 128$ (MD5), $n = 160$ (SHA−1) or $n = 256$ bits (SHA−256).
- no secret parameter.
- *H* must be easy to compute.

The security goals

pre-image resistance :

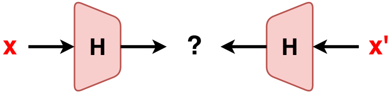
given an output challenge y, the attacker can not find a message x such that $H(x) = y$, in less than $\theta(2^n)$ operations.

2nd pre-image resistance :

given a challenge (x, y) so that $H(x) = y$, the attacker can not find a message $x' \neq x$ such that $H(x') = y$, in less than $\theta(2^n)$ operations.

collision resistance :

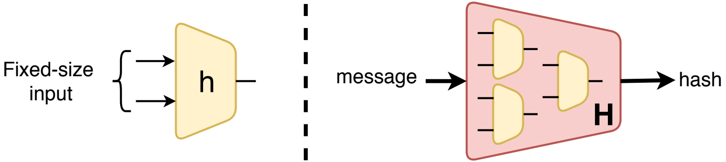
the attacker can not find two messages (x, x') such that $H(x) = H(x')$, in less than $\theta(2^{n/2})$ operations (a generic attack with the birthday paradox exists [Yuval-79]).



General hash construction

For historical reasons, most hash functions are composed of two elements :

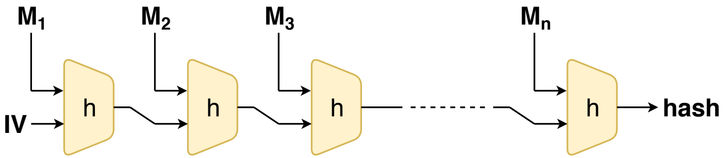
- **a compression function** *h* **:** a function for which the input and output size is fixed.
- **a domain extension algorithm :** an iterative process that uses the compression function *h* so that the hash function *H* can handle inputs of arbitrary length.



The Merkle-Damgård domain extension algorithm

The most famous domain extension algorithm used is called the **Merkle-Damgård** [Merkle Damgård-89] iterative algorithm.

$$\mathbf{pad(M) = M_1 \,||\, M_2 \,||\, M_3 \,||\, ... \,||\, M_n}$$



The compression function *h* now takes two fixed-size inputs, the incoming chaining variable *cv* and the message block *m*, and outputs a new chaining variable.

Compression function security notions

Idea : a collision on an iterated hash function H always comes from a collision on the compression function h :

$$H(M) = H(M^*) \Longrightarrow h(cv, m) = h(cv^*, m^*)$$

The conditions on (cv, m) give different kind of attacks

Collision $cv = cv^*$ fixed and $m \neq m^*$ free.

Semi-free-start Collision $cv = cv^*$ and $m \neq m^*$ are free.

Free-start Collision $(cv, m) \neq (cv^*, m^*)$ are free.

If the compression function is broken, go away from this hash function as soon as possible !

The sad story of MD5

The cryptanalysis history of MD5 is a good example of why **(semi)-free-start collisions are a serious warning.**

- 1992 MD5 RFC published
- 1993 pseudo-collision on the compression function [BB93]
- 1994 semi-free-start collision on the compression function [Dob96]
- 2004 practical collisions on the hash function [WFL04]
- 2007 chosen-prefix collisions and colliding X.509 certificates [SLW07]
- 2009 rogue CA certificates for RapidSSL [S+09] (used chosen-prefix collision)
- 2010-2012 Flame malware (used chosen-prefix collision)

Current state of MD-SHA-family

MD-SHA-family encompasses many hash functions :

- 1990' MDx (MD4, MD5, SHA−0, SHA−1, HAVAL, RIPEMD)
- 2002 SHA−2 (SHA−224, . . . , SHA−512)

Some old hash functions are still unbroken :

- Broken MD4, MD5
- Broken SHA−0, SHA−1
- Broken HAVAL
- Broken RIPEMD−0
- Unbroken RIPEMD−128, RIPEMD−160
- Unbroken SHA−2

What about SHA-1 ?

The current situation of SHA-1 :

- **1995** SHA-1 NIST FIPS 180-1 published : basically SHA-0 (1993) with a very small twist
- **2005** theoretical collision attack on the full hash function [WYY05] - 2^{69}
- **2006-2011** lots of works computing actual collisions for reduced-round versions of SHA-1
- **2015** free-start collision on the full compression computed [SKP15] - 2^{57}
- **2017** full collision on the full hash function computed [SBKAM17] - $2^{64.7}$
- **? ? ?** chosen-prefix collision attack ?
- **? ? ?** SHA-1 collisions in the wild ?

Motivations to study SHA−1

Why still studying SHA−1 ?

Design from NSA, **Standard** from NIST, **Still used worldwide** despite deprecation efforts (major browsers now refusing to connect to servers still using SHA−1- based certificates) :

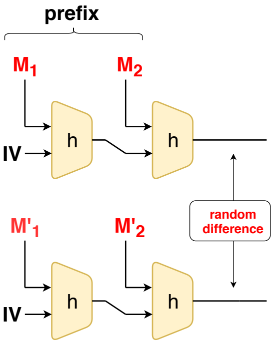
- more than 5% of Alexa's top 1 million **prefer** to use SHA−1 to authenticate TLS handshake messages (including www.skype.com)
- about 30000 servers with SHA−1 certificates (out of 720000 servers with HTTPS support)
- other protocols : about 1 million out of 4.55 millions mail servers (with IMAPS) use a SHA−1 certificate
- it is still possible to buy a SHA−1 certificate from a trusted root, and some can be found in the wild
- the "Mail" application included in Windows 10 still accepts SHA−1 certificates without warnings when opening an IMAPS connection

Yet another push is perhaps needed to accelerate the retirement of SHA−1

What are chosen-prefix collisions ?

Chosen-prefix collision attack

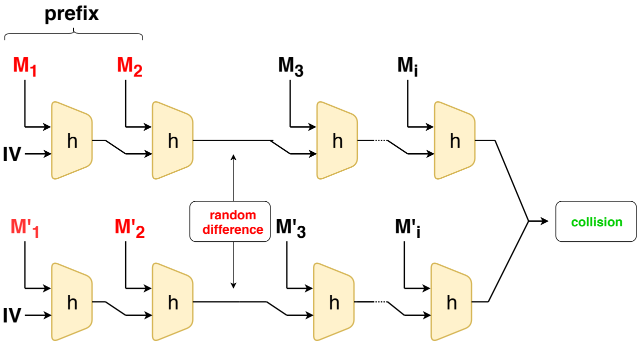
The attacker is first challenged with **two message prefixes** P and P', and its goal is to compute two messages M and M' to create the **collision** $H(P||M) = H(P'||M')$, where $||$ denotes concatenation



What are chosen-prefix collisions ?

Chosen-prefix collision attack

The attacker is first challenged with **two message prefixes** P and P', and its goal is to compute two messages M and M' to create the **collision** $H(P||M) = H(P'||M')$, where $||$ denotes concatenation



What are chosen-prefix collisions ?

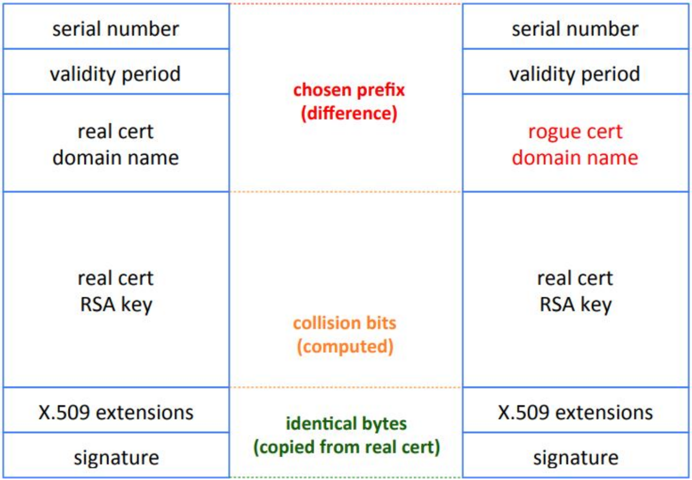
Chosen-prefix collision attack

The attacker is first challenged with **two message prefixes** P and P', and its goal is to compute two messages M and M' to create the **collision** $H(P||M) = H(P'||M')$, where $||$ denotes concatenation

- **much more powerful** than a simple collision attack (i.e. rogue CA certificate)
- supposedly much harder than a simple collision attack (currently for MD5, from 2^{16} to 2^{39})
- birthday attack can apply, thus generic cost remains $2^{n/2}$ (i.e. 2^{80} in the case of SHA−1)

Why chosen-prefix collisions ?

Colliding SSL certificates [S+09] :



Alex Sotirov `https://trailofbits.files.wordpress.com/2012/06/flame-md5.pdf`

Chosen-prefix collisions for SHA−1

Current status of chosen-prefix collisions on SHA−1 :

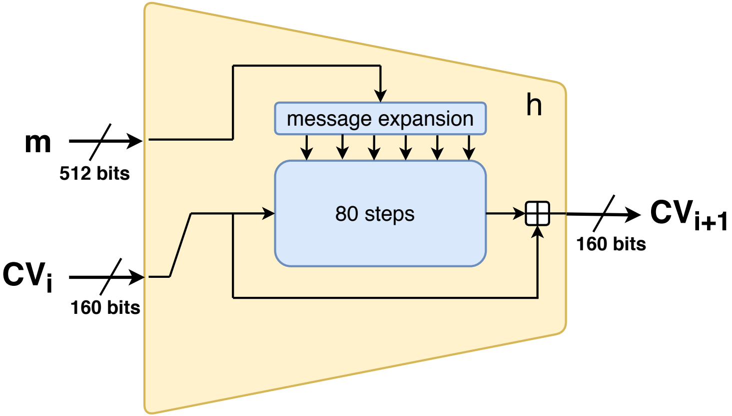
- **current best known chosen-prefix attack against SHA−1 requires $2^{77.1}$ computations** [S13] (a factor 8 better than generic attack).
- ... while classical collision could be found with $2^{64.7}$ computations.
- one can't apply directly the SHA−1 collision attack, because of the random state difference due to the challenge prefix.

Can we reduce the gap and make chosen-prefix collisions practical for SHA−1 ?

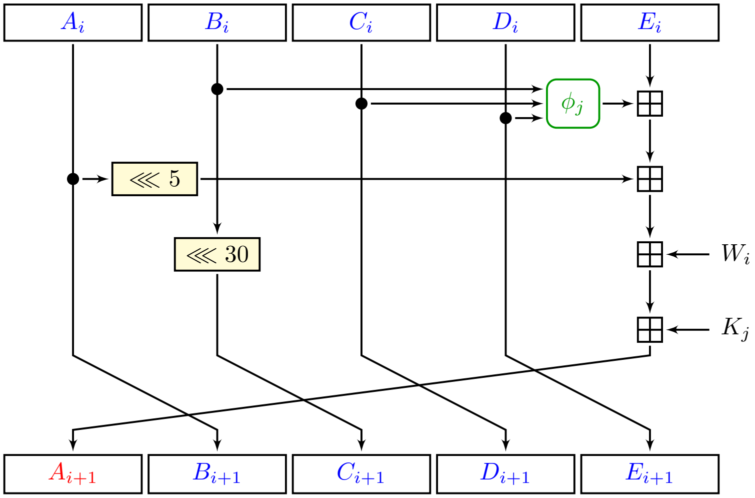
Outline

The SHA-1 compression function

$$m = M_0 || M_1 || \cdots || M_{15}$$



The SHA-1 step function



The SHA−1 step function : alternative representation

Message expansion :

$W_i = M_i$ for $i = [1, \ldots, 15]$, then

$W_i = (W_{i-16} \oplus W_{i-14} \oplus W_{i-8} \oplus W_{i-3})^{\lll 1}$

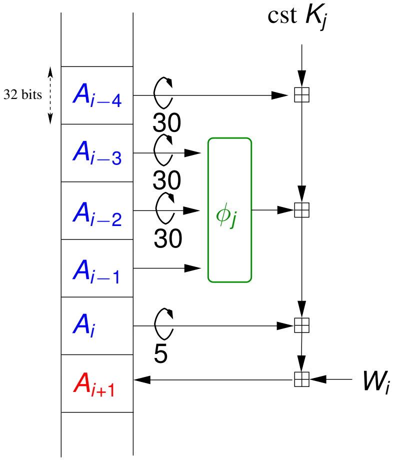
Boolean functions ϕ_j :

$\mathrm{IF}(x, y, z) := (x \wedge y) \oplus (\bar{x} \wedge z)$
for steps 0 to 19

$\mathrm{XOR}(x, y, z) := x \oplus y \oplus z$
for steps 20 to 39

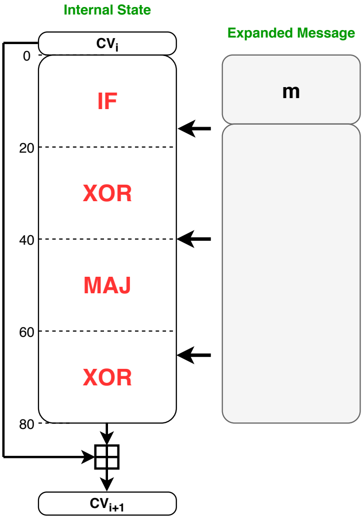
$\mathrm{MAJ}(x, y, z) := (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$
for steps 40 to 59

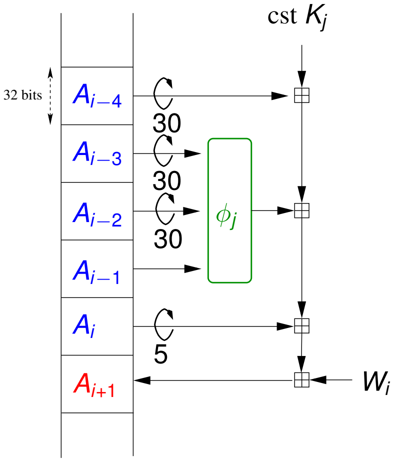
$\mathrm{XOR}(x, y, z) := x \oplus y \oplus z$
for steps 60 to 79



step i, round j

The SHA-1 step function : alternative representation



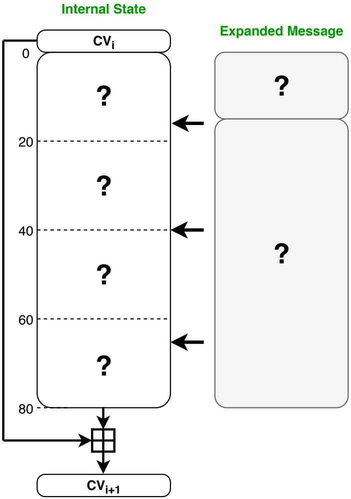


step i, round j

Outline

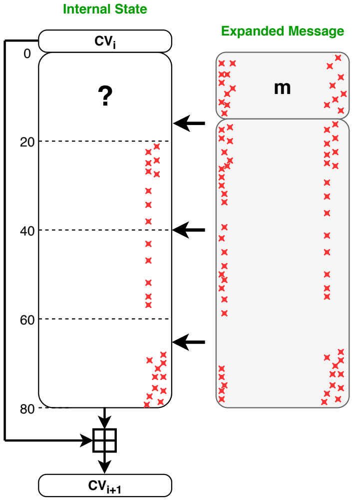
Goal : Find a collision for SHA−1

- **Find a linear path for rounds 16-80, using local collisions**
  this will fix the entire message difference, and the internal state difference for steps 16-80
- **Find a non-linear path for steps 1-15, using heuristic algorithm**
  this will fix the internal state difference for rounds 1-15
- **Prepare the collision search speed-up techniques** by using the freedom degrees available
- **Launch the collision search !**



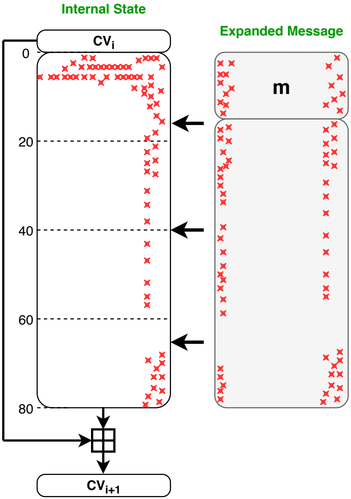
Goal : Find a collision for SHA−1

- **Find a linear path for rounds 16-80, using local collisions** this will fix the entire message difference, and the internal state difference for steps 16-80
- **Find a non-linear path for steps 1-15, using heuristic algorithm** this will fix the internal state difference for rounds 1-15
- **Prepare the collision search speed-up techniques** by using the freedom degrees available
- **Launch the collision search !**



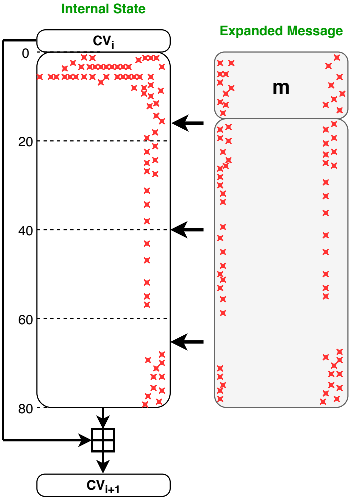
Goal : Find a collision for SHA-1

- **Find a linear path for rounds 16-80, using local collisions**
  this will fix the entire message difference, and the internal state difference for steps 16-80
- **Find a non-linear path for steps 1-15, using heuristic algorithm**
  this will fix the internal state difference for rounds 1-15
- **Prepare the collision search speed-up techniques** by using the freedom degrees available
- **Launch the collision search !**



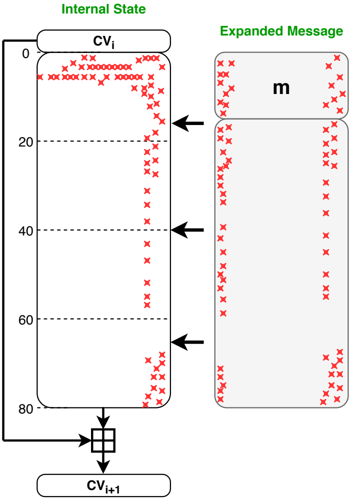
Goal : Find a collision for SHA−1

- **Find a linear path for rounds 16-80, using local collisions**
  this will fix the entire message difference, and the internal state difference for steps 16-80
- **Find a non-linear path for steps 1-15, using heuristic algorithm**
  this will fix the internal state difference for rounds 1-15
- **Prepare the collision search speed-up techniques** by using the freedom degrees available
- **Launch the collision search !**



Goal : Find a collision for SHA−1

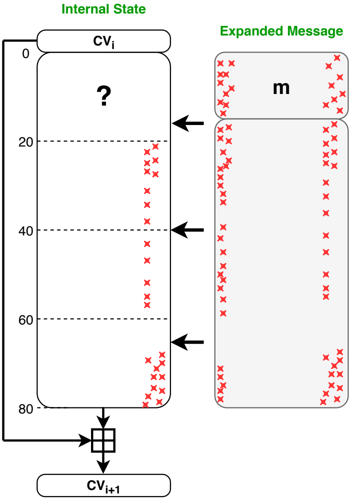
- **Find a linear path for rounds 16-80, using local collisions**
  this will fix the entire message difference, and the internal state difference for steps 16-80
- **Find a non-linear path for steps 1-15, using heuristic algorithm**
  this will fix the internal state difference for rounds 1-15
- **Prepare the collision search speed-up techniques** by using the freedom degrees available
- **Launch the collision search !**



Outline

Goal : Find a collision for SHA−1

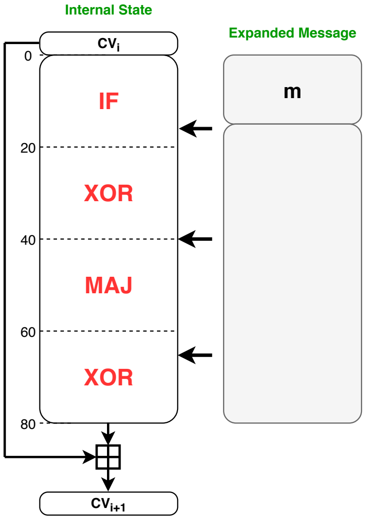
- **Find a linear path for rounds 16-80, using local collisions**
  this will fix the entire message difference, and the internal state difference for steps 16-80
- **Find a non-linear path for steps 1-15, using heuristic algorithm**
  this will fix the internal state difference for rounds 1-15
- **Prepare the collision search speed-up techniques** by using the freedom degrees available
- **Launch the collision search !**



What shape should have the differential path ?

Properties of the boolean functions :

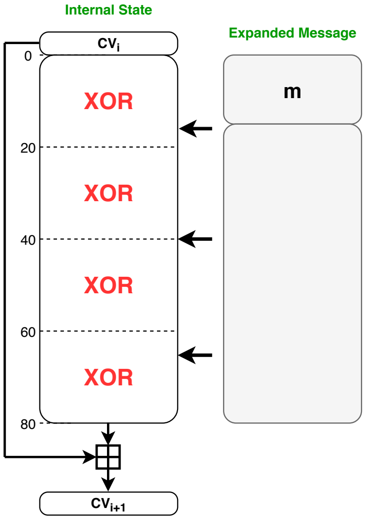
- XOR : no possible absorption of a difference
- MAJ : possible absorption of a difference (can always behave as a XOR)
- IF : possible absorption of a difference (can always behave as a XOR, except if two differences in the inputs)



What shape should have the differential path ?

Idea : linearise SHA−1

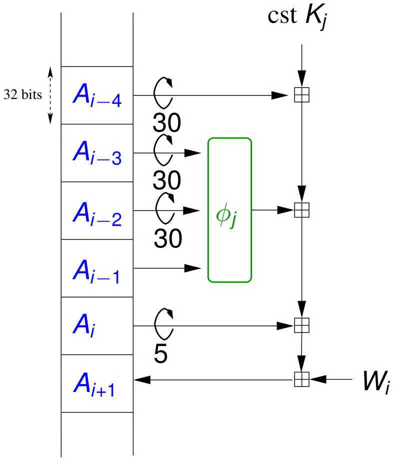
- **modular addition is replaced by XOR** (probability 1/2 per difference bit)
- **boolean functions IF and MAJ are replaced by XOR** (probability depends on situation)
- all the rest is already linear !



Local collisions [CJ98]

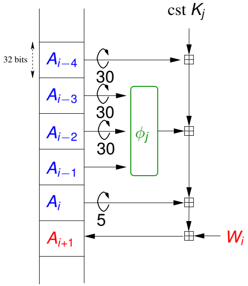
Local collision [CJ98]

Insert a perturbation difference and **correct it** so that it doesn't spread out !



step i, round j

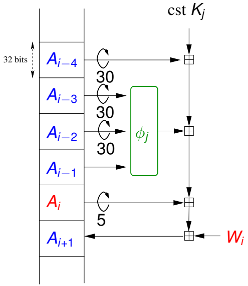
Local collisions



step	type	constraints
$i+1$	no carry	$W_i^j = a$, $A_{i+1}^j = a$

i	A_i	W_i
-1 :	--------------------------------	
00 :	--------------------------------	-----------------------------a--
01 :	-----------------------------a--	--------------------------------
02 :	--------------------------------	--------------------------------
03 :	--------------------------------	--------------------------------
04 :	--------------------------------	--------------------------------
05 :	--------------------------------	--------------------------------
06 :	--------------------------------	--------------------------------

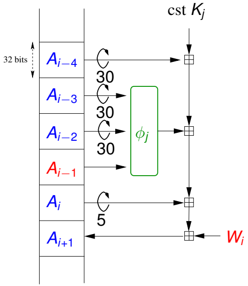
Local collisions



step	type	constraints
$i+1$	no carry	$W_i^j = a$, $A_{i+1}^j = a$
$i+2$	correction	$W_{i+1}^{j+5} = \overline{a}$

i	A_i	W_i
-1 :	--------------------------------	
00 :	--------------------------------	-----------------------------a--
01 :	-----------------------------a--	------------------------$\overline{a}$-------
02 :	--------------------------------	--------------------------------
03 :	--------------------------------	--------------------------------
04 :	--------------------------------	--------------------------------
05 :	--------------------------------	--------------------------------
06 :	--------------------------------	--------------------------------

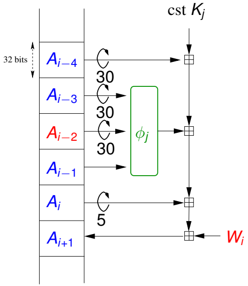
Local collisions



step	type	constraints
$i+1$	no carry	$W_i^j = a$, $A_{i+1}^j = a$
$i+2$	correction	$W_{i+1}^{j+5} = \bar{a}$
$i+3$	no correction	$A_{i-1}^{j+2} = A_i^{j+2}$
	correction	$A_{i-1}^{j+2} \neq A_i^{j+2}$, $W_{i+2}^j = \bar{a}$

i	A_i	W_i
-1 :	`---------------------------d----`	
00 :	`---------------------------d----`	`-----------------------------a--`
01 :	`-----------------------------a--`	`------------------------ā-------`
02 :	`--------------------------------`	`-----------------------------ā--`
03 :	`--------------------------------`	`--------------------------------`
04 :	`--------------------------------`	`--------------------------------`
05 :	`--------------------------------`	`--------------------------------`
06 :	`--------------------------------`	`--------------------------------`

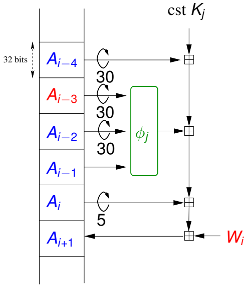
Local collisions



step	type	constraints
$i+1$	no carry	$W_i^j = a$, $A_{i+1}^j = a$
$i+2$	correction	$W_{i+1}^{j+5} = \bar{a}$
$i+3$	correction	$A_{i-1}^{j+2} \neq A_i^{j+2}$, $W_{i+2}^j = \bar{a}$
$i+4$	no correction	$A_{i+2}^{j-2} = 0$
	correction	$A_{i+2}^{j-2} = 1$, $W_{i+3}^{j-2} = \bar{a}$

i	A_i	W_i
-1 :	`---------------------------d----`	
00 :	`---------------------------d----`	`-----------------------------a--`
01 :	`-----------------------------a--`	`------------------------ā-------`
02 :	`-------------------------------1`	`-----------------------------ā--`
03 :	`--------------------------------`	`-------------------------------ā`
04 :	`--------------------------------`	`--------------------------------`
05 :	`--------------------------------`	`--------------------------------`
06 :	`--------------------------------`	`--------------------------------`

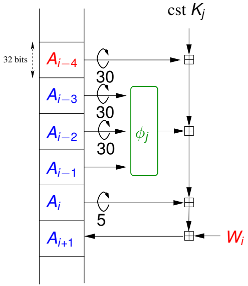
Local collisions



step	type	constraints
$i+1$	no carry	$W_i^j = a$, $A_{i+1}^j = a$
$i+2$	correction	$W_{i+1}^{j+5} = \bar{a}$
$i+3$	correction	$A_{i-1}^{j+2} \neq A_i^{j+2}$, $W_{i+2}^j = \bar{a}$
$i+4$	correction	$A_{i+2}^{j-2} = 1$, $W_{i+3}^{j-2} = \bar{a}$
$i+5$	no correction	$A_{i+3}^{j-2} = 1$
	correction	$A_{i+3}^{j-2} = 0$, $W_{i+4}^{j-2} = \bar{a}$

i	A_i	W_i
-1 :	---------------------------d----	
00 :	---------------------------d----	-----------------------------a--
01 :	-----------------------------a--	------------------------ā-------
02 :	-------------------------------1	-----------------------------ā--
03 :	-------------------------------0	-------------------------------ā
04 :	--------------------------------	-------------------------------ā
05 :	--------------------------------	--------------------------------
06 :	--------------------------------	--------------------------------

Local collisions



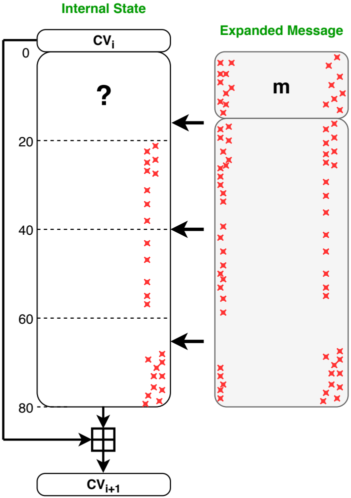
step	type	constraints
$i+1$	no carry	$W_i^j = a,\ A_{i+1}^j = a$
$i+2$	correction	$W_{i+1}^{j+5} = \bar{a}$
$i+3$	correction	$A_{i-1}^{j+2} \neq A_i^{j+2},\ W_{i+2}^j = \bar{a}$
$i+4$	correction	$A_{i+2}^{j-2} = 1,\ W_{i+3}^{j-2} = \bar{a}$
$i+5$	correction	$A_{i+3}^{j-2} = 0,\ W_{i+4}^{j-2} = \bar{a}$
$i+6$	correction	$W_{i+5}^{j-2} = \bar{a}$

i	A_i	W_i
-1 :	---------------------------d----	
00 :	---------------------------d----	-----------------------------a--
01 :	-----------------------------a--	------------------------ā-------
02 :	-------------------------------1	-----------------------------ā--
03 :	-------------------------------0	-------------------------------ā
04 :	--------------------------------	-------------------------------ā
05 :	--------------------------------	-------------------------------ā
06 :	--------------------------------	--------------------------------

Linearisation of SHA−1

How to build a linear differential path :

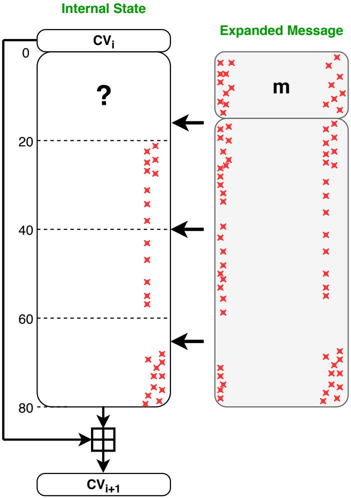
- use **only local collisions**
- the **mask of local collisions simply needs to be generated from the message expansion formula**
- this will fix all the differences in the expanded message and in the 60 last steps of the internal state



Linearisation of SHA−1

Finding the best linear path candidate :

- a local collision costs is 2^{-4} (can be improved when placed at the proper bit position to avoid carry conditions)
- go through all the possible **low hamming weight** 16-step windows (going through all the possible windows would be too costly)
- compute the cost induced by the local collisions for this candidate
- keep the best one (i.e. the cheapest) for steps 16-80 !



Structure of the differential path

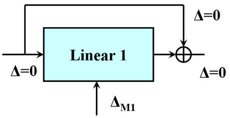
Why did we remove the 16 first steps ?

- it allows to **avoid impossibilities due to the IF function**
- impossible to get a one-block collision with good probability : necessary to use several blocks
- A non-linear part allows to **start from any incoming difference in the chaining variable** : 2-block collision

Structure of the differential path

Why did we remove the 16 first steps ?

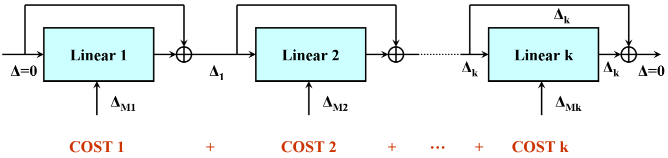
- it allows to **avoid impossibilities due to the IF function**
- impossible to get a one-block collision with good probability : necessary to use several blocks
- A non-linear part allows to **start from any incoming difference in the chaining variable** : 2-block collision



Structure of the differential path

Why did we remove the 16 first steps ?

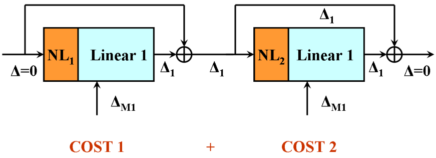
- it allows to **avoid impossibilities due to the IF function**
- impossible to get a one-block collision with good probability : necessary to use several blocks
- A non-linear part allows to **start from any incoming difference in the chaining variable** : 2-block collision



Structure of the differential path

Why did we remove the 16 first steps ?

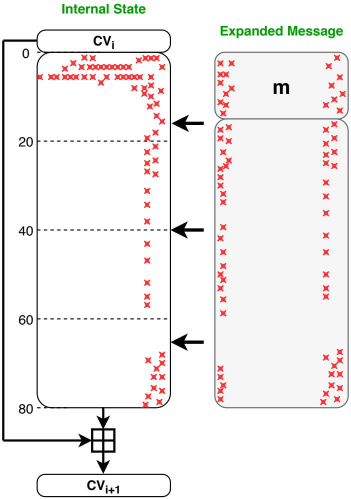
- it allows to **avoid impossibilities due to the IF function**
- impossible to get a one-block collision with good probability : necessary to use several blocks
- A non-linear part allows to **start from any incoming difference in the chaining variable** : 2-block collision



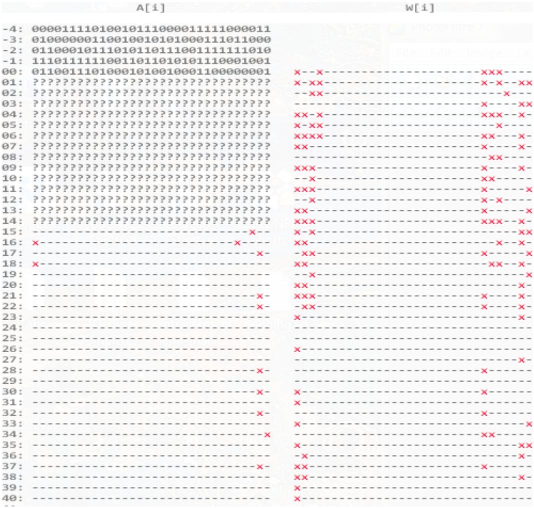
Outline

Goal : Find a collision for SHA−1

- **Find a linear path for rounds 16-80, using local collisions** this will fix the entire message difference, and the internal state difference for steps 16-80
- **Find a non-linear path for steps 1-15, using heuristic algorithm this will fix the internal state difference for rounds 1-15**
- **Prepare the collision search speed-up techniques** by using the freedom degrees available
- **Launch the collision search !**



Example of a non-linear path



Example of a non-linear path

```
                 A[i]                                  W[i]
-4: 00001111010010111000011111000011
-3: 01000000110010010101000111011000
-2: 01100010111010110111001111111010
-1: 11101111110011011010101110001001
00: 01100111010001010010001100000001   n11u00000-----------1----nuu-001
01: 0nnn111111----------0--1-11n-100   u0nu1--------------------u0u1-nu
02: n00n00-----0-------0--n1-u10--1u   --nu---------------------1-1n1--
03: -0-u11-----1------nnn-0100-01uu-   -------------------------n----xn
04: 0--011-------nnn----0010u1-0--11   xn0x---------------------unn--x1
05: 1-unu1uu--u--0-000u01001unnn-n10   u-nx-----------------------u---0
06: 1n110n00000-10-000-0100n1un10110   nnnx-----------0---0-11-1uu---u-
07: u1u01u11001-1--0-0n0111n1nn1-n-n   un--------1---1-00-0-----u----n-
08: 1nnnnnnnnnnnnnn--0-u00n00-01-1-n   --------------------------un----
09: --1--1110000000101--11-11111un11   xnn-010001-----------1--0n----n-
10: -0-10110111111111--1u01-1nu-u-1u   -1n----------------------nn----0
11: u1--1--------------00-0-01001-01   xnu--------------1-0-1-00u0----u
12: -00----------------01n1-nuu11-1u   --u----------------------u-n----
13: --0------------------nuuuu-uuun-   xn-----------------------u-----n
14: n1----------------------10n0---1   xxx----------------------xxx--n-
15: x-0--------------------011-01u11   x-n---------------------------xx
16: n--------------------------u0--0   xx-------------------------x--n-
17: -----------------------------1n1   -xn----------------------u-----u
18: x----------------------------1--   xn------------------------xx--u-
19: --------------------------------   --u----------------------------n
20: --------------------------------   xu----------------------0-----n-
21: ------------------------------u-   xxx----------------------n----x-
22: ------------------------------x-   -xx----------------------x----x-
23: --------------------------------   x-----------------------------x-
24: --------------------------------   --------------------------------
25: --------------------------------   --------------------------------
26: --------------------------------   x-------------------------------
27: --------------------------------   ------------------------------x-
28: ------------------------------x-   -------------------------x------
29: --------------------------------   --------------------------------
30: ------------------------------x-   x------------------------x------
31: --------------------------------   x-------------------------------
32: ------------------------------x-   -------------------------x------
33: --------------------------------   x------------------------------x
34: -------------------------------x   -------------------------xx-----
35: --------------------------------   x-----------------------------xx
36: --------------------------------   -x----------------------------x-
37: ------------------------------x-   xx-----------------------x------
38: --------------------------------   xx----------------------------x-
39: --------------------------------   x-------------------------------
40: --------------------------------   x-------------------------------
```

Generating a non-linear path

The non-linear path search algorithm

Input : a differential path with '?' only in the internal state in steps 1-14, with steps 14-20 being determined pseudo-linearly

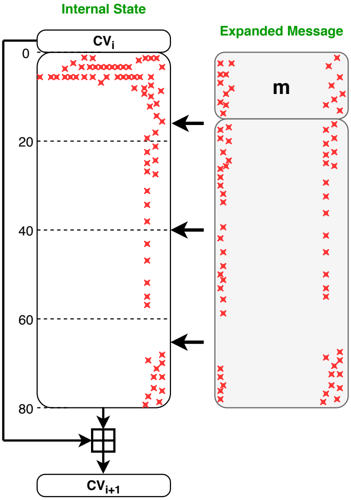
Output : a differential path where no '?' nor 'x' exist anymore in the internal state : the path is fully determined and signed

- **very sensitive to many parameters**, **quite technical** and **hard to make it right**
- heuristic strategy [CR06] or using SAT solvers [SBKAM17]

Outline

Goal : Find a collision for SHA−1

- **Find a linear path for rounds 16-80, using local collisions**
  this will fix the entire message difference, and the internal state difference for steps 16-80
- **Find a non-linear path for steps 1-15, using heuristic algorithm**
  this will fix the internal state difference for rounds 1-15
- **Prepare the collision search speed-up techniques** by using the freedom degrees available
- **Launch the collision search !**



A simple collision search algorithm

A naive collision search algorithm

Repeat until a collision is found :

- pick a random message
- test if it follows the entire differential path

The **very costly** non-linear part has to be paid ☹ ☹ ☹

A better collision search algorithm

A less-naive search algorithm :

$i = 1$

repeat until a collision is found :

- pick a random message word M_i (backtrack sometimes)
- test if it follows the differential path for step i
  - if it does, then $i = i + 1$
- when $i = 16$, test if it follows the entire differential path

The very costly non-linear part is avoided ☺ ☺ ☺, **only the linear probabilistic part remains to be paid**

Final collision search algorithm

Final collision search in 3 phases :

- **Step 1 :** handling the low-probability non-linear parts using the message block freedom
- **Step 2 :** apply the collision search speed-up techniques
- **Step 3 :** the remaining steps are verified probabilistically

Computation cost further reduced 😃 😃 😃 only steps ~22-80 have to be considered

Collision search speed-up techniques

Several techniques to speed-up the collision search :

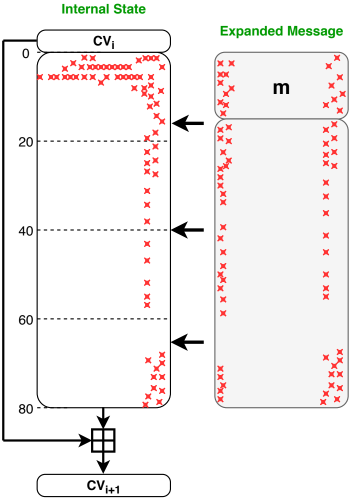
- message modifications [WYY05]
- neutral bits [BC04]
- boomerangs (or tunnels) [K06,JP07]

All these techniques trade message freedom degrees for a speed-up factor

Outline

Goal : Find a collision for SHA−1

- **Find a linear path for rounds 16-80, using local collisions** this will fix the entire message difference, and the internal state difference for steps 16-80
- **Find a non-linear path for steps 1-15, using heuristic algorithm this will fix the internal state difference for rounds 1-15**
- **Prepare the collision search speed-up techniques** by using the freedom degrees available
- **Launch the collision search !**

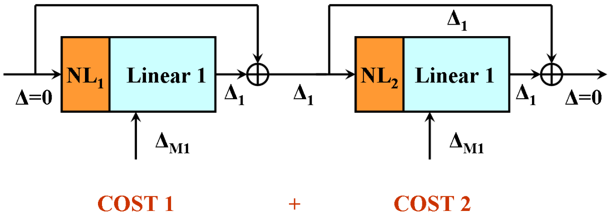


Outline

Why chosen-prefix collision is hard for SHA−1

Can the SHA−1 collision attack be directly adapted for chosen-prefix scenario ?

No : we can't remove the random difference on the chaining variable with the very small number of possible output differences of the linear path



Why chosen-prefix collision is hard for SHA−1

Can the SHA−1 collision attack be directly adapted for chosen-prefix scenario ?

No : we can't remove the random difference on the chaining variable with the very small number of possible output differences of the linear path

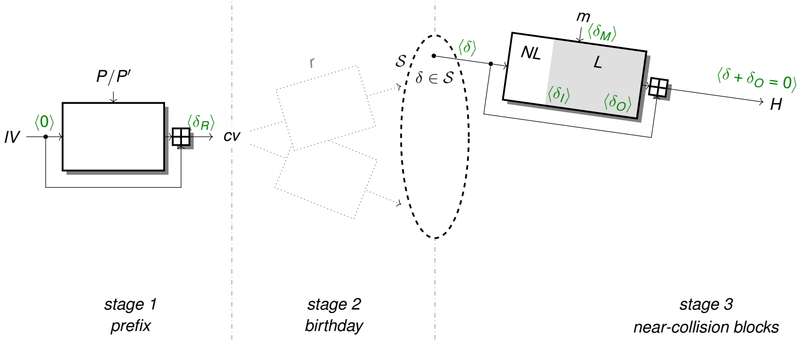
We will use the recent SHA−1 collision attack [SBKAM17] as a black box. Denote C ($\simeq 2^{64.7}$) the computational cost for the last block.

Assume that we can use the same attack, for the same cost C, whatever is the input difference (this is possible thanks to the non-linear search algorithm). We validated this assumption in practice for several randomly chosen input differences.

Birthday to the rescue !

Trick 1 : birthday search

Use **birthday search** to reduce the entropy of possible chaining variable differences



Birthday to the rescue !

Trick 1 : birthday search

Use **birthday search** to reduce the entropy of possible chaining variable differences

Assume that you have a set $\mathcal{S}$ of differences that you can reach on the output of the internal cipher, for a computation cost C.

phase 1 : compute the (random) state difference δ_R induced by the challenge prefix

phase 2 : apply birthday strategy to map difference δ_R to a difference δ that belongs to a certain set $\mathcal{S}$ (requires $\sqrt{\pi \cdot 2^n / |\mathcal{S}|}$ computations)

phase 3 : apply the collision attack as explained previously (with a cost C) to map difference δ to a pair of colliding states

Birthday to the rescue !

Trick 1 : birthday search

Use **birthday search** to reduce the entropy of possible chaining variable differences

Stevens [S13] identified 192 possible output differences that can be reached for the same minimal cost C

phase 1 : O(1)

phase 2 : a birthday phase of $2^{77.1}$ computations

phase 3 : a collision phase of $C = 2^{64.7}$ computations

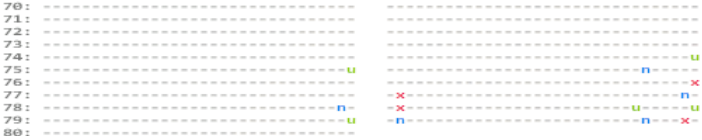
Total is $2^{77.1}$ computations (birthday phase is dominating)

Relaxing the output differences constrains

Trick 2a : generalized output differences

Using heuristics, we found **more allowable output differences** than previously known (576 instead of 192) for a cost C. This will increase $\mathcal{S}$.

For a maximal computational cost of C per block, we found a set $\mathcal{S}$ of 576 elements (instead of 192)

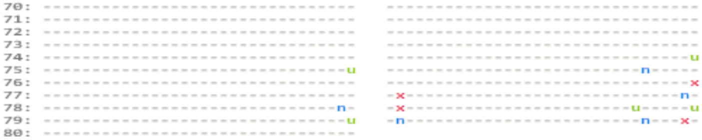


Relaxing the output differences constrains

Trick 2b : generalized output differences

Accept **more costly differential paths** to further increase $\mathcal{S}$ (so that the birthday phase and collision phase have about the same computational cost).

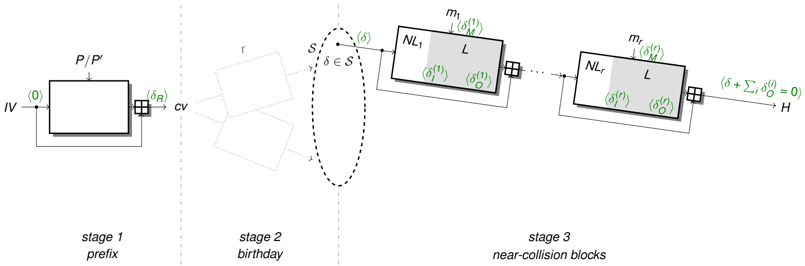
For a maximal computational cost of $8 \cdot C$ per block, we found a set $\mathcal{S}$ of 8768 elements.



Multiblocks : the more the merrier !

Trick 3 : multiblocks

Use **several blocks** to reach the collision after the birthday phase. It will increase the size of $\mathcal{S}$ and thus reduce the birthday phase cost.



Multiblocks : the more the merrier !

Trick 3 : multiblocks

Use **several blocks** to reach the collision after the birthday phase. It will increase the size of $\mathcal{S}$ and thus reduce the birthday phase cost.

With a maximum cost of $3 \cdot C$, one obtains a set $\mathcal{S}$ containing about 2^{30} elements !

The maximum length of the chain of blocks is 54 and the average is 17.

Multiblocks : the more the merrier !

Trick 3 : multiblocks

Use **several blocks** to reach the collision after the birthday phase. It will increase the size of $\mathcal{S}$ and thus reduce the birthday phase cost.

A three-phase attack :

phase 1 : compute the (random) state difference δ_R induced by the challenge prefix

phase 2 : apply birthday strategy to map difference δ_R to a difference δ that belongs to a certain set $\mathcal{S}$

phase 3 : apply a few consecutive SHA−1 block attacks to slowly map difference δ to a pair of colliding states

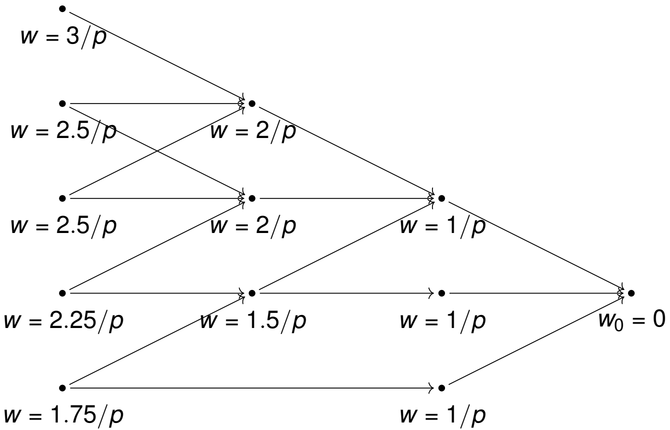
The clustering effect

Trick 4 : use the clustering effect

The attacker can sometimes **target several nodes simultaneously** to reduce the cost (because it is easier to hit one node out of many than a fixed one).

He will select **dynamically** the allowable differences at the output of each successive blocks. For that, we need a “map” of the whole situation, so we can decide which output difference I should be targeting at each new block. We will build a graph $\mathcal{G}$ for that.

Example of a graph $\mathcal{G}$



(we assume that all the edges have probability p)

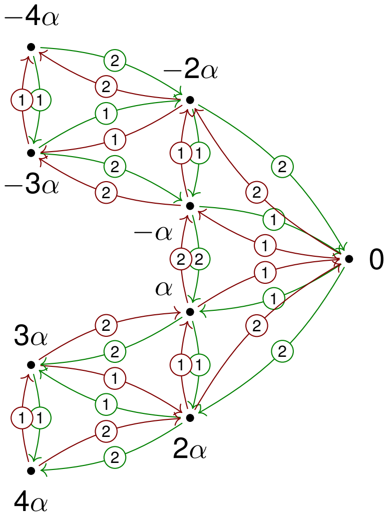
Building the graph $\mathcal{G}$ and the set $\mathcal{S}$

We call “bundle” a set of output differences that can be tried at the same time. We can build a graph $\mathcal{G}'$ with the bundles.

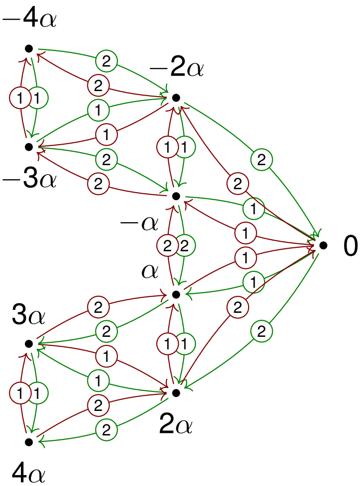
Consider that you have :

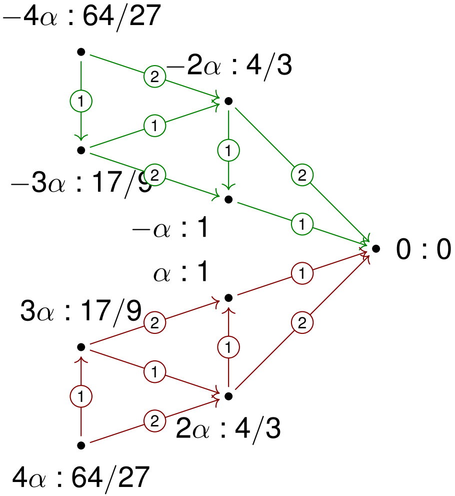
- a bundle $\{\alpha, 2\alpha\}$ with costs 1 and 2 (green lines)
- a bundle $\{-\alpha, -2\alpha\}$ with costs 1 and 2 (red lines)

The corresponding set $\mathcal{S}$ is $\{-4\alpha, -3\alpha, -2\alpha, -\alpha, 0, \alpha, 2\alpha, 3\alpha, 4\alpha\}$



Building the graph $\mathcal{G}$ and the set $\mathcal{S}$





Outline

Our results : SHA−1

We obtain a **practical chosen-prefix collision attack on** SHA−1 :

- with only $2^{66.9}$ computations ($2^{69.4}$ with very pessimistic assumptions)
- only a small factor more costly than a classical collision attack
- all subparts of the attack verified experimentally

Function	Collision type	Complexity	Reference
SHA−1	free-start collision	$2^{57.5}$	[SKP16]
	collision	2^{69} $2^{64.7}$	[WYY05] [SBKAM17]
	chosen-prefix collision	$2^{77.1}$ $2^{66.9}$	[S13] our result

Our results : MD5

We can apply our strategy to MD5 (and others ?) :

- less impressive because chosen-prefix collision attacks on MD5 are already rather cheap (thus the birthday phase becomes not competitive)
- however, when the number of blocks for the chosen-prefix collision is limited, our attacks becomes competitive

Function	Collision type	Complexity	Reference
MD5	collision	2^{40}	[WY05]
		2^{16}	[SSALMO09]
	chosen-prefix collision (9 blocks)	$2^{39.1}$	[SSALMO09]
	(3 blocks)	2^{49}	[SSALMO09]
	(1 block)	$2^{53.2}$	[SSALMO09]
	(2 blocks)	$2^{46.5}$	our result

Thank you for your attention !

NANYANG TECHNOLOGICAL UNIVERSITY