Collision Spectrum, Entropy Loss, T-Sponges and Cryptanalysis of GLUON-64

Léo Perrin   Dmitry Khovratovitch
firstname.lastname@uni.lu

University of Luxembourg

UNIVERSITÉ DU LUXEMBOURG

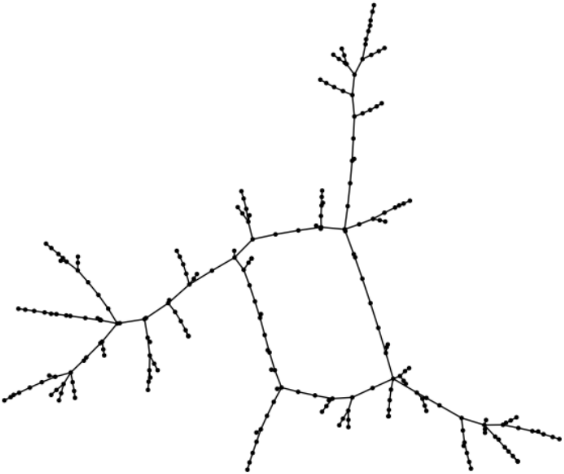
March 3, 2014

Random functions

What happens when a random function is used to update the internal state of a cryptographic primitive?

Random functions

What happens when a random function is used to update the internal state of a cryptographic primitive?



Known Results

CPS and Iterated (Pre)-Images

Applications to Cryptography

Application to GLUON-64

Conclusion

Plan

Known Results

CPS and Iterated (Pre)-Images

Applications to Cryptography

Application to GLUON-64

Conclusion

Random Functions Statistics

Flajolet and Odlyzko (89), on a random functions $g : \mathcal{S} \rightarrow \mathcal{S}$:

Random Functions Statistics

Flajolet and Odlyzko (89), on a random functions $g : \mathcal{S} \to \mathcal{S}$:

- Distribution of the preimages sizes for $a \xleftarrow{\$} \mathcal{S}$:

$$\mathbb{P}[g(x) = a \text{ has } k \text{ solutions for } a \xleftarrow{\$} \mathcal{S}] = e^{-1}/k!$$

Random Functions Statistics

Flajolet and Odlyzko (89), on a random functions $g : \mathcal{S} \to \mathcal{S}$:

- Distribution of the preimages sizes for $a \xleftarrow{\$} \mathcal{S}$:

$$\mathbb{P}[g(x) = a \text{ has } k \text{ solutions for } a \xleftarrow{\$} \mathcal{S}] = e^{-1}/k!$$

- (Expected) size of iterated image: $|g^i(\mathcal{S})| \approx \frac{|\mathcal{S}|}{i/2}$

Random Functions Statistics

Flajolet and Odlyzko (89), on a random functions $g : \mathcal{S} \to \mathcal{S}$:

- Distribution of the preimages sizes for $a \xleftarrow{\$} \mathcal{S}$:

$$\mathbb{P}[g(x) = a \text{ has } k \text{ solutions for } a \xleftarrow{\$} \mathcal{S}] = e^{-1}/k!$$

- (Expected) size of iterated image: $|g^i(\mathcal{S})| \approx \frac{|\mathcal{S}|}{i/2}$
- (Expected) cycle and tail length: $\sqrt{\frac{\pi|\mathcal{S}|}{8}}$

Random Functions Statistics

Flajolet and Odlyzko (89), on a random functions $g : \mathcal{S} \to \mathcal{S}$:

- Distribution of the preimages sizes for $a \stackrel{\$}{\leftarrow} \mathcal{S}$:

$$\mathbb{P}[g(x) = a \text{ has } k \text{ solutions for } a \stackrel{\$}{\leftarrow} \mathcal{S}] = e^{-1}/k!$$

- (Expected) size of iterated image: $|g^i(\mathcal{S})| \approx \frac{|\mathcal{S}|}{i/2}$
- (Expected) cycle and tail length: $\sqrt{\frac{\pi|\mathcal{S}|}{8}}$
- ...

Random Functions Statistics

Flajolet and Odlyzko (89), on a random functions $g : \mathcal{S} \to \mathcal{S}$:

- Distribution of the preimages sizes for $a \overset{\$}{\leftarrow} \mathcal{S}$:

$$\mathbb{P}[g(x) = a \text{ has } k \text{ solutions for } a \overset{\$}{\leftarrow} \mathcal{S}] = e^{-1}/k!$$

- (Expected) size of iterated image: $|g^i(\mathcal{S})| \approx \frac{|\mathcal{S}|}{i/2}$
- (Expected) cycle and tail length: $\sqrt{\frac{\pi|\mathcal{S}|}{8}}$
- ...

For functions chosen uniformly at random among all the functions from $\mathcal{S}$ to itself (random mappings).

Using state shrinking/presence of trees

- Trees and output shrinking used to attack A5/1 (Golic 97, Biryukov et. al. 01).

Using state shrinking/presence of trees

- Trees and output shrinking used to attack A5/1 (Golic 97, Biryukov et. al. 01).
- Shrinking of the state space of MICKEY observed by Hong and Kim (05), studied by Röck (08).

Plan

Known Results

CPS and Iterated (Pre)-Images

Applications to Cryptography

Application to GLUON-64

Conclusion

Collision Probability Spectrum (CPS)

Definition (Collision Probability Spectrum)

We call *Collision Probability Spectrum* (CPS) of $g : \mathcal{S} \rightarrow \mathcal{S}$ the set $\{\mathfrak{c}_k\}_{k \geq 1}$

$$\mathfrak{c}_k = \mathbb{P}[g(a+x) = g(a) \text{ has } k \text{ solutions}].$$

Collision Probability Spectrum (CPS)

Definition (Collision Probability Spectrum)

We call *Collision Probability Spectrum* (CPS) of $g : \mathcal{S} \to \mathcal{S}$ the set $\{\mathfrak{c}_k\}_{k \geq 1}$

$$\mathfrak{c}_k = \mathbb{P}[g(a + x) = g(a) \text{ has } k \text{ solutions}].$$

Definition

The average number of non-zero roots is denoted κ and called *collision average*:

$$\kappa = \sum_{k \geq 1} \mathfrak{c}_k \cdot k - 1$$

Output Shrinking

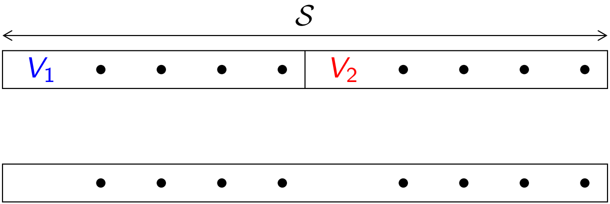
Let $V_k = \{x_0 \in \mathcal{S}, g(x_0 + y) = g(x_0) \text{ has } k \text{ solutions}\}$. $\Rightarrow |V_k| = \mathfrak{c}_k \cdot |\mathcal{S}|$

Output Shrinking

Let $V_k = \{x_0 \in \mathcal{S}, g(x_0 + y) = g(x_0) \text{ has } k \text{ solutions}\}$. $\Rightarrow |V_k| = \mathfrak{c}_k \cdot |\mathcal{S}|$
Let g have CPS $\{\mathfrak{c}_1 = \mathfrak{c}_2 = 1/2\}$.

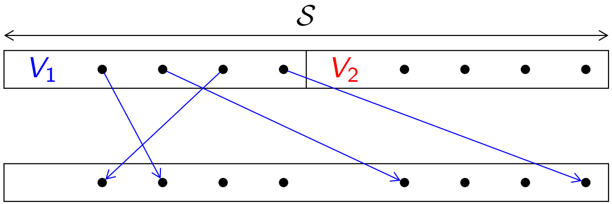
Output Shrinking

Let $V_k = \{x_0 \in \mathcal{S}, g(x_0 + y) = g(x_0) \text{ has } k \text{ solutions}\}$. $\Rightarrow |V_k| = \mathfrak{c}_k \cdot |\mathcal{S}|$
Let g have CPS $\{\mathfrak{c}_1 = \mathfrak{c}_2 = 1/2\}$.



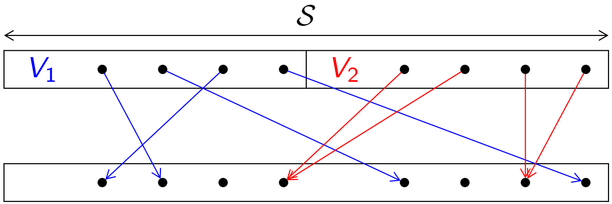
Output Shrinking

Let $V_k = \{x_0 \in \mathcal{S}, g(x_0 + y) = g(x_0) \text{ has } k \text{ solutions}\}$. $\Rightarrow |V_k| = \mathfrak{c}_k \cdot |\mathcal{S}|$
Let g have CPS $\{\mathfrak{c}_1 = \mathfrak{c}_2 = 1/2\}$.



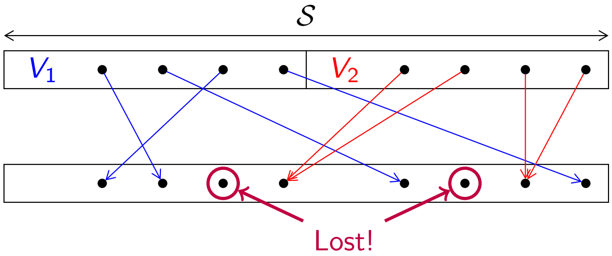
Output Shrinking

Let $V_k = \{x_0 \in \mathcal{S}, g(x_0 + y) = g(x_0) \text{ has } k \text{ solutions}\}$. $\Rightarrow |V_k| = \mathfrak{c}_k \cdot |\mathcal{S}|$
Let g have CPS $\{\mathfrak{c}_1 = \mathfrak{c}_2 = 1/2\}$.



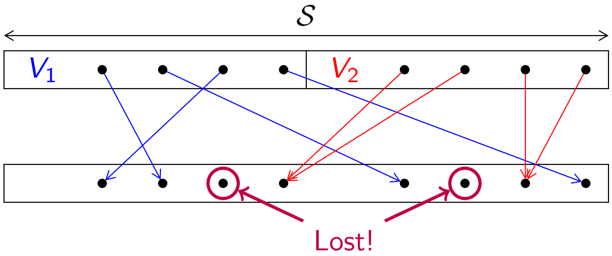
Output Shrinking

Let $V_k = \{x_0 \in \mathcal{S}, g(x_0 + y) = g(x_0) \text{ has } k \text{ solutions}\}$. $\Rightarrow |V_k| = \mathfrak{c}_k \cdot |\mathcal{S}|$
Let g have CPS $\{\mathfrak{c}_1 = \mathfrak{c}_2 = 1/2\}$.



Output Shrinking

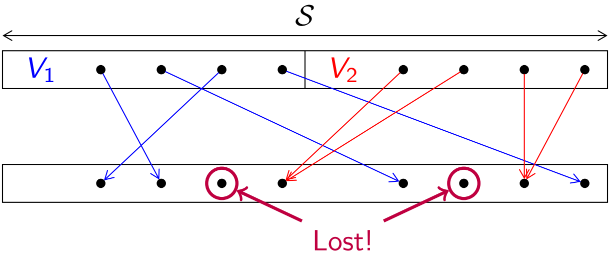
Let $V_k = \{x_0 \in \mathcal{S}, g(x_0 + y) = g(x_0) \text{ has } k \text{ solutions}\}$. $\Rightarrow |V_k| = \mathfrak{c}_k \cdot |\mathcal{S}|$
Let g have CPS $\{\mathfrak{c}_1 = \mathfrak{c}_2 = 1/2\}$.



$$|g(V_k)| = \frac{\mathfrak{c}_k}{k} \cdot |\mathcal{S}|$$

Output Shrinking

Let $V_k = \{x_0 \in \mathcal{S}, g(x_0 + y) = g(x_0) \text{ has } k \text{ solutions}\}$. $\Rightarrow |V_k| = \mathfrak{c}_k \cdot |\mathcal{S}|$
Let g have CPS $\{\mathfrak{c}_1 = \mathfrak{c}_2 = 1/2\}$.

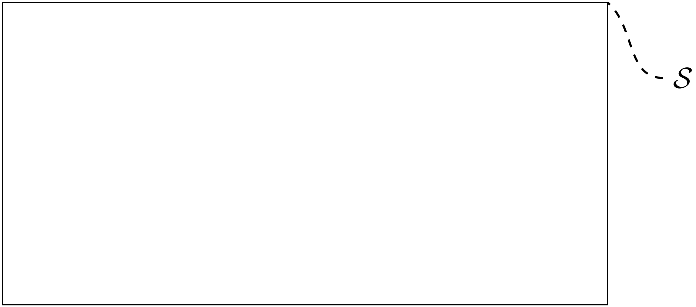


$$|g(V_k)| = \frac{\mathfrak{c}_k}{k} \cdot |\mathcal{S}|$$

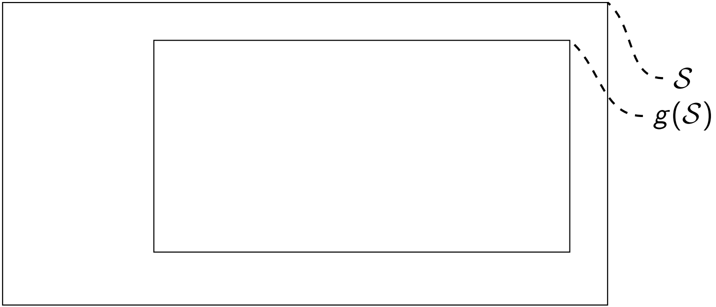
Independence Assumption: In what follows, we assume that $x \in g(V_k)$ and $x \in V_j$ are independent for any k, j.

Iterated Output Shrinking and Collision Tree

$g : \mathcal{S} \to \mathcal{S}$ has CPS $\{c_3 = 1\}$; # iterations $< \sqrt{|\mathcal{S}|}$

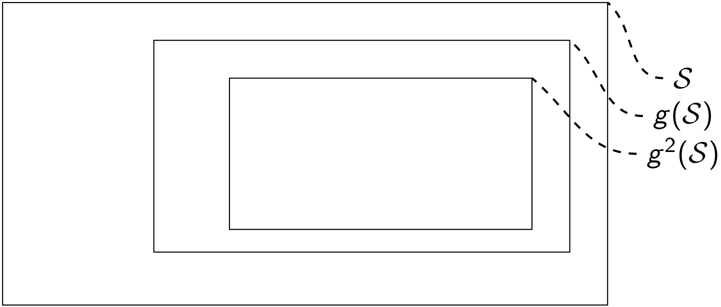


$g : \mathcal{S} \rightarrow \mathcal{S}$ has CPS $\{c_3 = 1\}$; # iterations $< \sqrt{|\mathcal{S}|}$



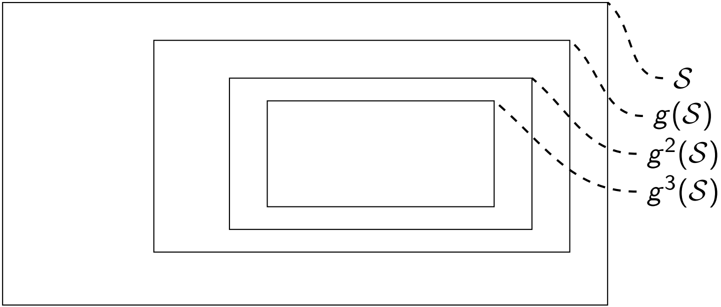
Iterated Output Shrinking and Collision Tree

$g : \mathcal{S} \to \mathcal{S}$ has CPS $\{c_3 = 1\}$; # iterations $< \sqrt{|\mathcal{S}|}$



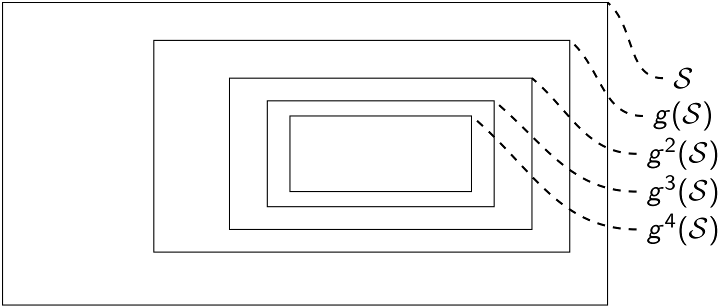
Iterated Output Shrinking and Collision Tree

$g : \mathcal{S} \rightarrow \mathcal{S}$ has CPS $\{c_3 = 1\}$; # iterations $< \sqrt{|\mathcal{S}|}$

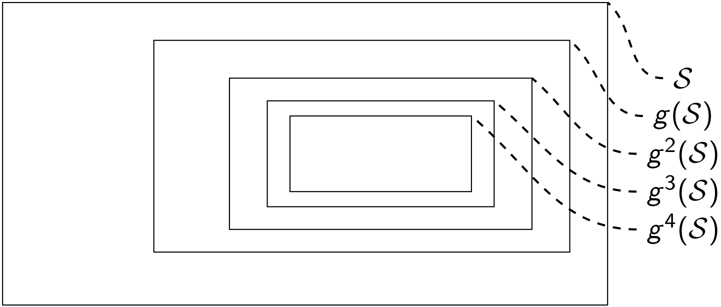


Iterated Output Shrinking and Collision Tree

$$g : \mathcal{S} \to \mathcal{S} \text{ has CPS } \{c_3 = 1\}; \# \text{ iterations} < \sqrt{|\mathcal{S}|}$$

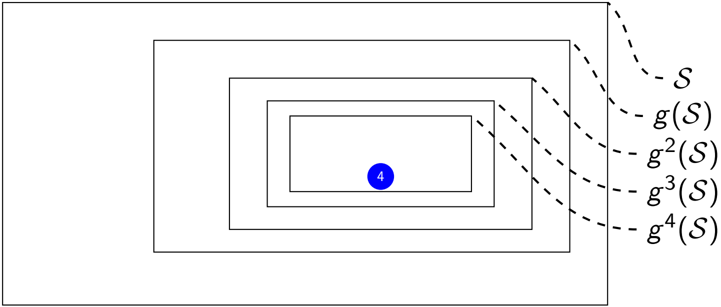


$$g : \mathcal{S} \to \mathcal{S} \text{ has CPS } \{c_3 = 1\};\ \#\text{ iterations} < \sqrt{|\mathcal{S}|}$$



$$|g^i(\mathcal{S})| \sim \frac{|\mathcal{S}|}{i \cdot \kappa/2}$$

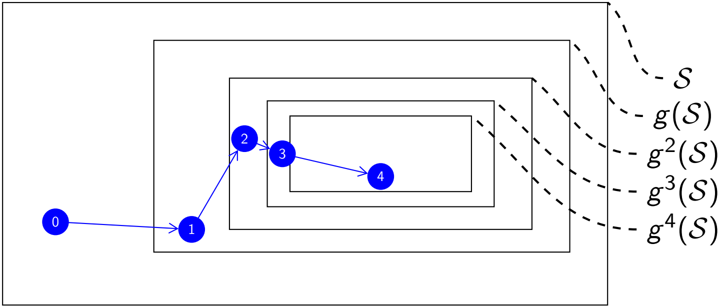
$g : \mathcal{S} \to \mathcal{S}$ has CPS $\{c_3 = 1\}$; # iterations $< \sqrt{|\mathcal{S}|}$



$$|g^i(\mathcal{S})| \sim \frac{|\mathcal{S}|}{i \cdot \kappa/2}$$

Iterated Output Shrinking and Collision Tree

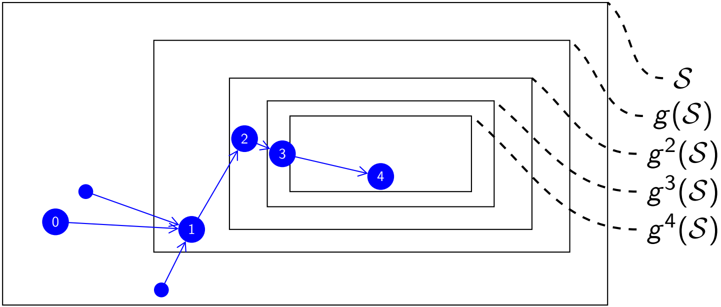
$$g : \mathcal{S} \to \mathcal{S} \text{ has CPS } \{c_3 = 1\};\ \#\text{ iterations} < \sqrt{|\mathcal{S}|}$$



$$|g^i(\mathcal{S})| \sim \frac{|\mathcal{S}|}{i \cdot \kappa/2}$$

Iterated Output Shrinking and Collision Tree

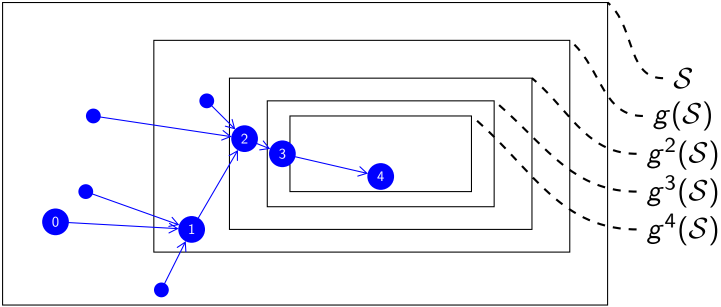
$$g : \mathcal{S} \rightarrow \mathcal{S} \text{ has CPS } \{c_3 = 1\};\ \#\text{ iterations} < \sqrt{|\mathcal{S}|}$$



$$|g^i(\mathcal{S})| \sim \frac{|\mathcal{S}|}{i \cdot \kappa/2}$$

Iterated Output Shrinking and Collision Tree

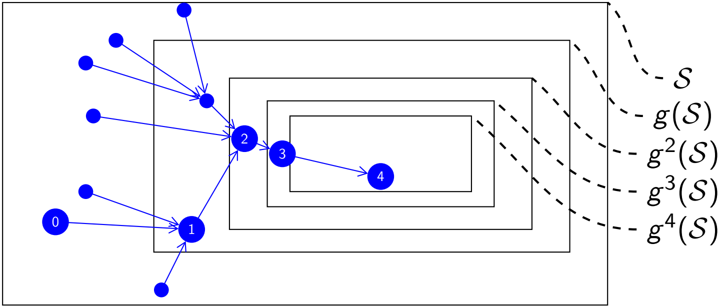
$$g : \mathcal{S} \to \mathcal{S} \text{ has CPS } \{c_3 = 1\};\ \#\text{ iterations} < \sqrt{|\mathcal{S}|}$$



$$|g^i(\mathcal{S})| \sim \frac{|\mathcal{S}|}{i \cdot \kappa/2}$$

Iterated Output Shrinking and Collision Tree

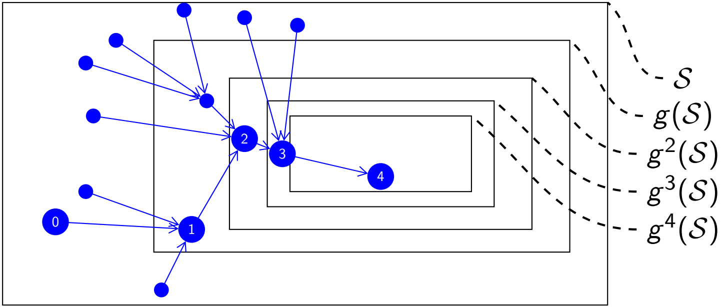
$$g : \mathcal{S} \to \mathcal{S} \text{ has CPS } \{c_3 = 1\};\ \#\text{ iterations} < \sqrt{|\mathcal{S}|}$$



$$|g^i(\mathcal{S})| \sim \frac{|\mathcal{S}|}{i \cdot \kappa/2}$$

Iterated Output Shrinking and Collision Tree

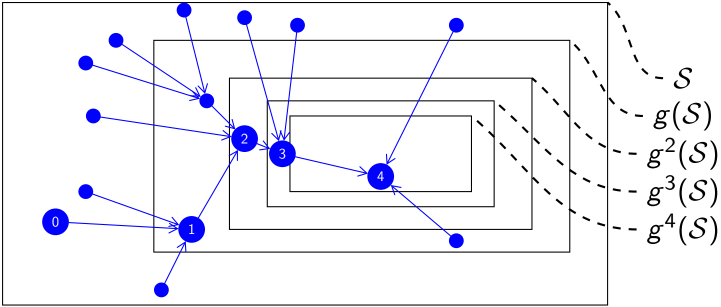
$$g : \mathcal{S} \to \mathcal{S} \text{ has CPS } \{c_3 = 1\}; \# \text{ iterations} < \sqrt{|\mathcal{S}|}$$



$$|g^i(\mathcal{S})| \sim \frac{|\mathcal{S}|}{i \cdot \kappa/2}$$

Iterated Output Shrinking and Collision Tree

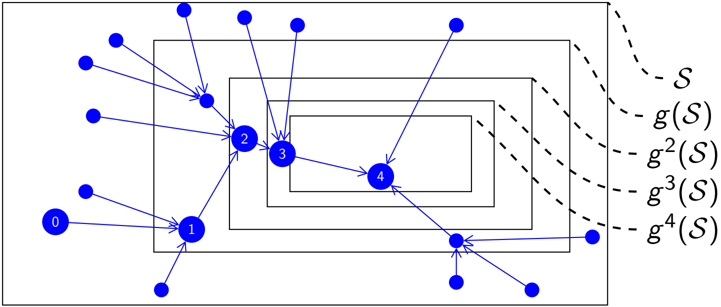
$$g : \mathcal{S} \to \mathcal{S} \text{ has CPS } \{c_3 = 1\};\ \#\text{ iterations} < \sqrt{|\mathcal{S}|}$$



$$|g^i(\mathcal{S})| \sim \frac{|\mathcal{S}|}{i \cdot \kappa/2}$$

Iterated Output Shrinking and Collision Tree

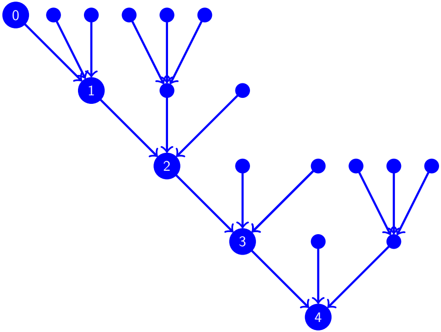
$$g : \mathcal{S} \rightarrow \mathcal{S} \text{ has CPS } \{c_3 = 1\}; \# \text{ iterations} < \sqrt{|\mathcal{S}|}$$



$$|g^i(\mathcal{S})| \sim \frac{|\mathcal{S}|}{i \cdot \kappa/2}$$

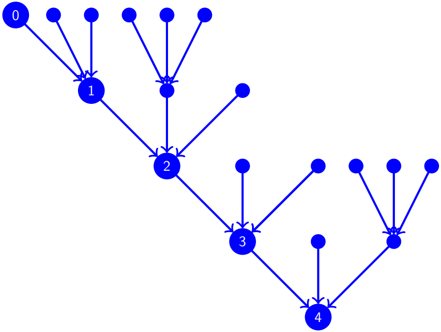
Iterated Output Shrinking and Collision Tree

$$g : \mathcal{S} \to \mathcal{S} \text{ has CPS } \{c_3 = 1\};\ \#\text{ iterations} < \sqrt{|\mathcal{S}|}$$



$$|g^i(\mathcal{S})| \sim \frac{|\mathcal{S}|}{i \cdot \kappa/2}$$

$$g : \mathcal{S} \to \mathcal{S} \text{ has CPS } \{c_3 = 1\}; \ \# \text{ iterations} < \sqrt{|\mathcal{S}|}$$



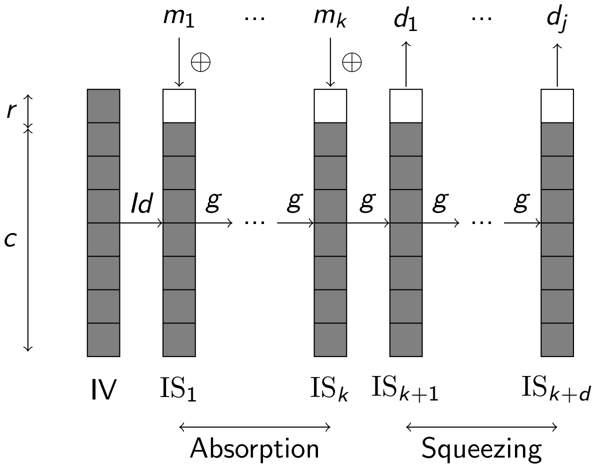
$$|g^i(\mathcal{S})| \sim \frac{|\mathcal{S}|}{i \cdot \kappa/2} \qquad \#\{ \text{ nodes in tree rooted in } g^i(\mathcal{S})\} \sim \frac{\kappa}{4} \cdot i^2$$

Known CPS's

Function	κ	$\lvert\mathcal{S}\rvert / \lvert g^i(\mathcal{S})\rvert$	tree size
MICKEY's update function	0.625	$2^{-1.7} \cdot i$	$2^{-2.7} \cdot i^2$
Random mapping	1	$2^{-1} \cdot i$	$2^{-2} \cdot i^2$
GLUON-64's update function	6.982	$2^{1.8} \cdot i$	$2^{0.8} \cdot i^2$

Plan

The T-sponge Construction



- c: capacity
- r: bitrate
- $m_1, ..., m_k$: Message
- $d_1, ..., d_j$: Digest
- g: random function

Flat Sponge Claim Revisited

If g is a function with collision average κ, then finding collisions with Q queries succeeds with probability

$$\frac{Q^2}{2^{c+1}} \cdot \left(1 + \frac{\kappa - 1}{2^r}\right).$$

Flat Sponge Claim Revisited

If g is a function with collision average κ, then finding collisions with Q queries succeeds with probability

$$\frac{Q^2}{2^{c+1}} \cdot \left(1 + \frac{\kappa - 1}{2^r}\right).$$

Intuition: $\mathcal{S}$ has size 2^{c+r}. Collisions occur because of the "trimming" of the bitrate ($2^r/2^{c+r} = 2^c$) and because of inner-collisions ($\kappa/2^{c+r}$).

Keyed Walk

Definition

Given a space $\mathcal{S}$, functions $g_k : \mathcal{S} \rightarrow \mathcal{S}$ and a sequence of keys $\{k_1, ..., k_m\}$, a *keyed walk* starting in x_0 is such that

$$x_{i+1} = g_{k_i}(x_i).$$

Keyed Walk

Definition

Given a space $\mathcal{S}$, functions $g_k : \mathcal{S} \to \mathcal{S}$ and a sequence of keys $\{k_1, ..., k_m\}$, a *keyed walk* starting in x_0 is such that

$$x_{i+1} = g_{k_i}(x_i).$$

Example:

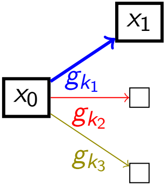
- Keys: $\{k_1, k_2, k_3\}$
- Sequence: $\{k_1, k_3, k_3, k_1, k_2\}$
- Functions: $g_{k_i}, i = 1, 2, 3.$

Keyed Walk

Definition

Given a space $\mathcal{S}$, functions $g_k : \mathcal{S} \rightarrow \mathcal{S}$ and a sequence of keys $\{k_1, ..., k_m\}$, a *keyed walk* starting in x_0 is such that

$$x_{i+1} = g_{k_i}(x_i).$$



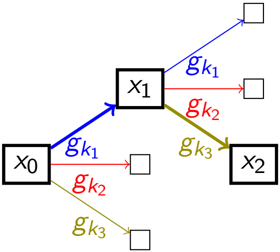
Example:

- Keys: $\{k_1, k_2, k_3\}$
- Sequence: $\{k_1, k_3, k_3, k_1, k_2\}$
- Functions: $g_{k_i}, i = 1, 2, 3.$

Definition

Given a space $\mathcal{S}$, functions $g_k : \mathcal{S} \to \mathcal{S}$ and a sequence of keys $\{k_1, ..., k_m\}$, a *keyed walk* starting in x_0 is such that

$$x_{i+1} = g_{k_i}(x_i).$$



Example:

- Keys: $\{k_1, k_2, k_3\}$
- Sequence: $\{k_1, k_3, k_3, k_1, k_2\}$
- Functions: $g_{k_i}, i = 1, 2, 3.$

Keyed Walk

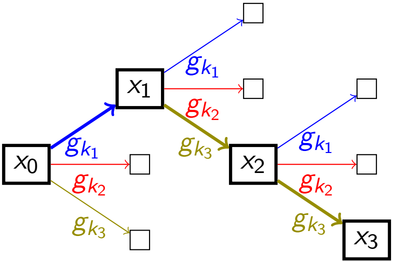
Definition

Given a space $\mathcal{S}$, functions $g_k : \mathcal{S} \to \mathcal{S}$ and a sequence of keys $\{k_1, ..., k_m\}$, a *keyed walk* starting in x_0 is such that

$$x_{i+1} = g_{k_i}(x_i).$$

Example:

- Keys: $\{k_1, k_2, k_3\}$
- Sequence: $\{k_1, k_3, k_3, k_1, k_2\}$
- Functions: $g_{k_i}, i = 1, 2, 3.$



Keyed Walk

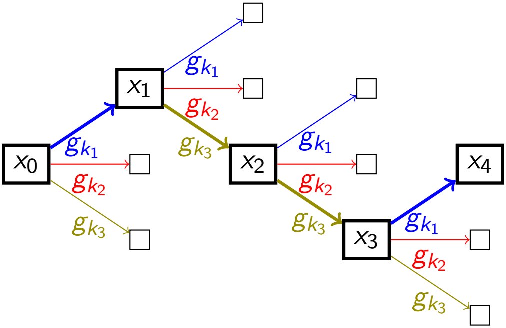
Definition

Given a space $\mathcal{S}$, functions $g_k : \mathcal{S} \to \mathcal{S}$ and a sequence of keys $\{k_1, ..., k_m\}$, a *keyed walk* starting in x_0 is such that

$$x_{i+1} = g_{k_i}(x_i).$$

Example:

- Keys: $\{k_1, k_2, k_3\}$
- Sequence: $\{k_1, k_3, k_3, k_1, k_2\}$
- Functions: $g_{k_i}, i = 1, 2, 3.$



Keyed Walk

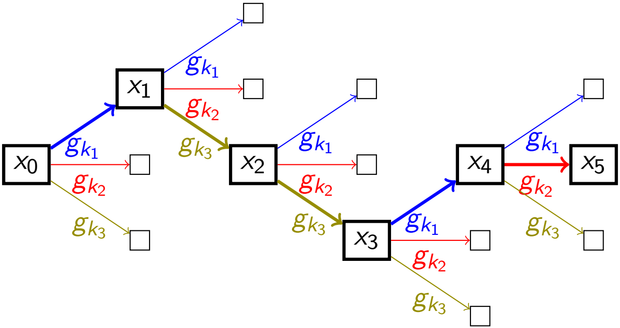
Definition

Given a space $\mathcal{S}$, functions $g_k : \mathcal{S} \rightarrow \mathcal{S}$ and a sequence of keys $\{k_1, ..., k_m\}$, a *keyed walk* starting in x_0 is such that

$$x_{i+1} = g_{k_i}(x_i).$$

Example:

- Keys: $\{k_1, k_2, k_3\}$
- Sequence: $\{k_1, k_3, k_3, k_1, k_2\}$
- Functions: $g_{k_i}, i = 1, 2, 3.$



Let t be such that
$t \in g^{i}_{k_1}(\mathcal{S})$

Let t be such that
$t \in g_{k_1}^{i}(\mathcal{S})$

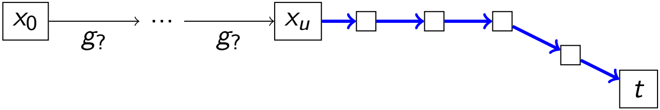
$|g_{k_1}^{-i}(t)| \approx i \cdot \kappa/2$

$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$

Let t be such that
$t \in g_{k_1}^{i}(\mathcal{S})$

$|g_{k_1}^{-i}(t)| \approx i \cdot \kappa/2$

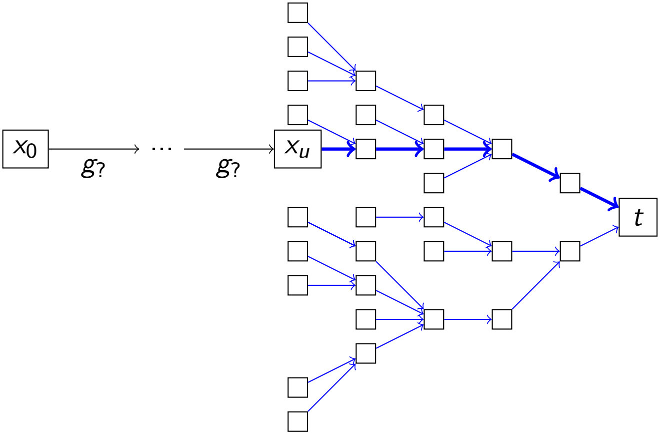
$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$



Let t be such that $t \in g_{k_1}^{i}(\mathcal{S})$

$|g_{k_1}^{-i}(t)| \approx i \cdot \kappa/2$

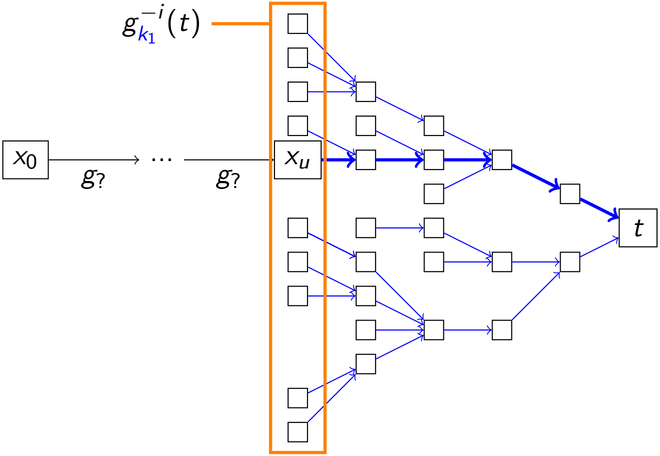
$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$



Let t be such that $t \in g^{i}_{k_1}(\mathcal{S})$

$|g^{-i}_{k_1}(t)| \approx i \cdot \kappa/2$

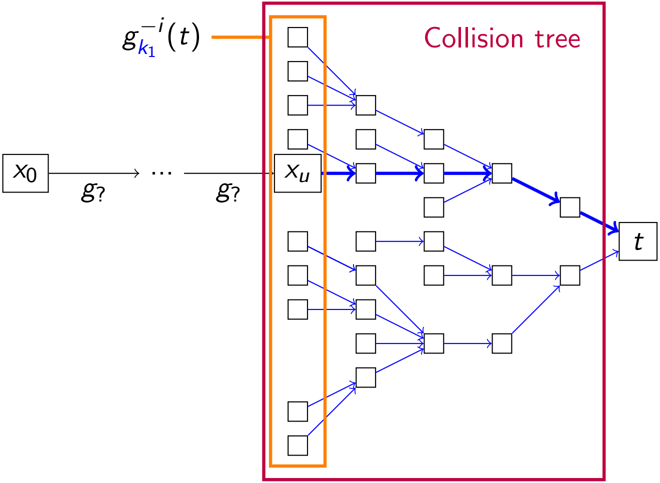
$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$



Let t be such that $t \in g^{i}_{k_1}(\mathcal{S})$

$|g^{-i}_{k_1}(t)| \approx i \cdot \kappa/2$

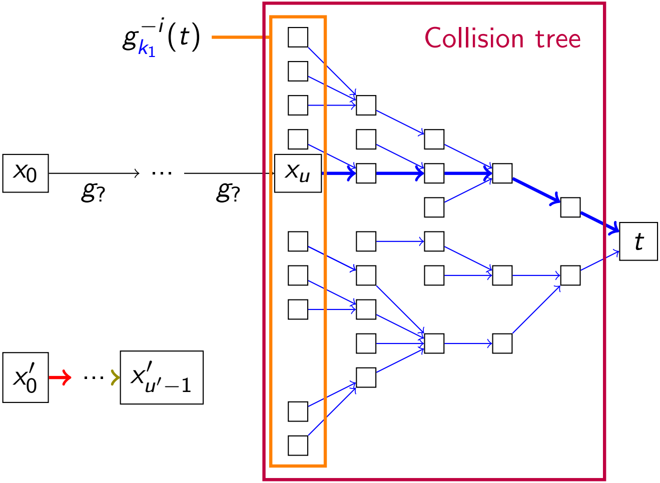
$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$



Let t be such that $t \in g^{i}_{k_1}(\mathcal{S})$

$|g^{-i}_{k_1}(t)| \approx i \cdot \kappa/2$

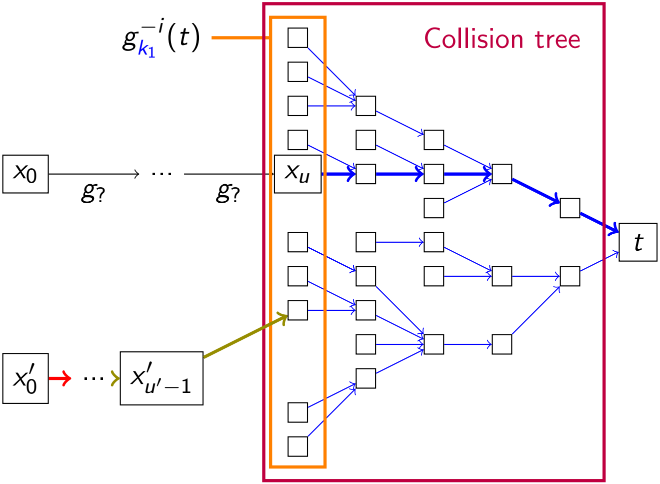
$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$



Let t be such that $t \in g_{k_1}^{i}(\mathcal{S})$

$|g_{k_1}^{-i}(t)| \approx i \cdot \kappa/2$

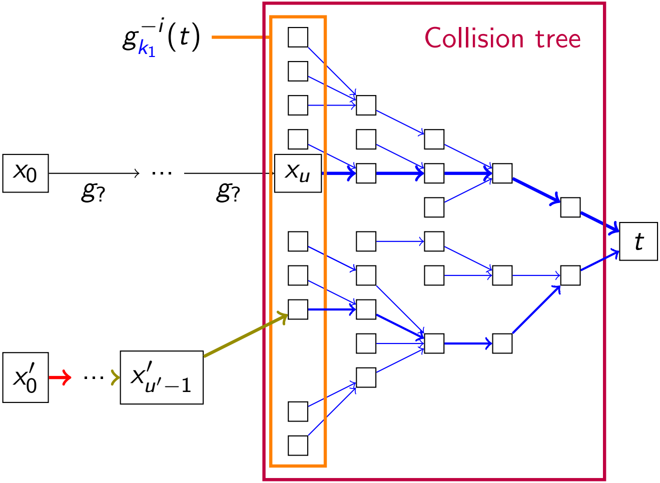
$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$



Let t be such that $t \in g^{i}_{k_1}(\mathcal{S})$

$|g^{-i}_{k_1}(t)| \approx i \cdot \kappa/2$

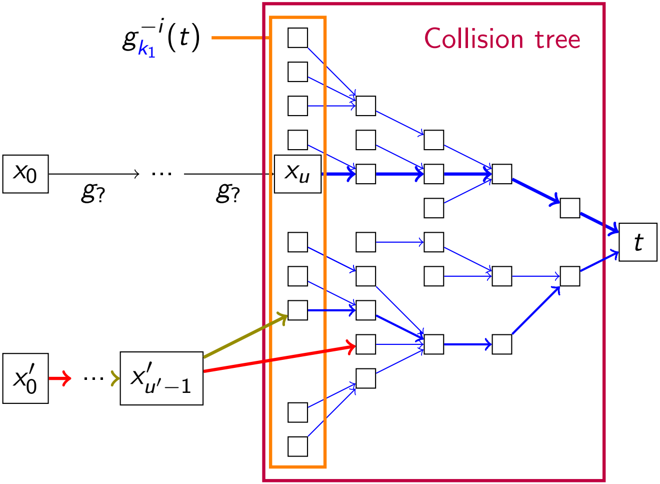
$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$



Let t be such that $t \in g_{k_1}^{i}(\mathcal{S})$

$|g_{k_1}^{-i}(t)| \approx i \cdot \kappa/2$

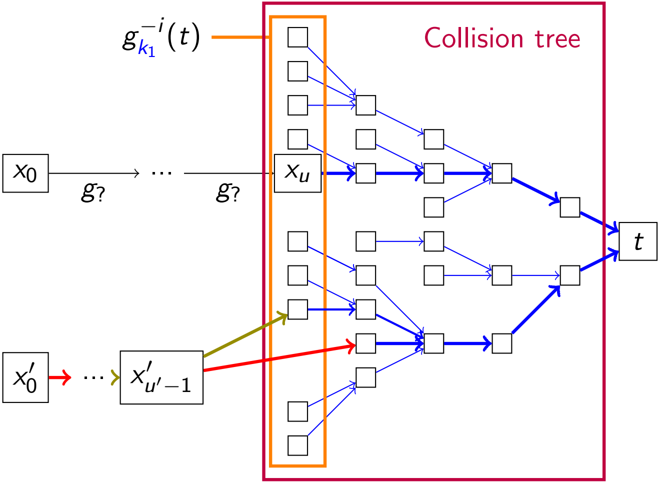
$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$



Let t be such that $t \in g_{k_1}^{i}(\mathcal{S})$

$|g_{k_1}^{-i}(t)| \approx i \cdot \kappa/2$

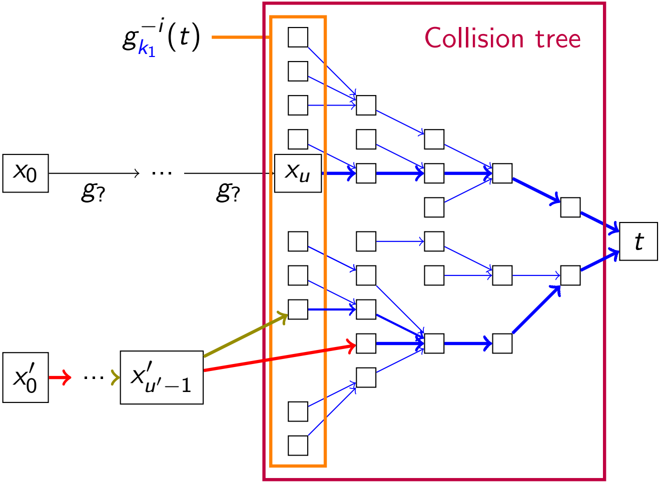
$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$



Let t be such that $t \in g^{i}_{k_1}(\mathcal{S})$

$|g^{-i}_{k_1}(t)| \approx i \cdot \kappa/2$

$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$



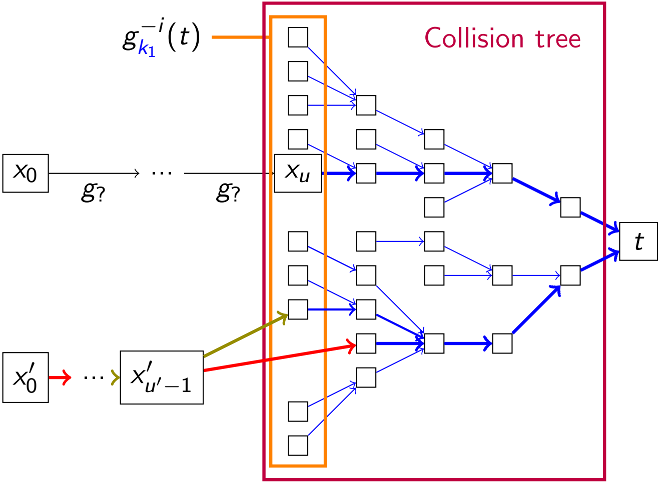
Finding element in $g^{-i}_{k_1}(t)$

$$\mathcal{C} \approx \frac{|\mathcal{S}|}{\kappa/2}$$

Let t be such that $t \in g_{k_1}^{i}(\mathcal{S})$

$|g_{k_1}^{-i}(t)| \approx i \cdot \kappa/2$

$|\mathrm{Tree}| \approx i^2 \cdot \kappa/4$



Finding element in $g_{k_1}^{-i}(t)$

$$\mathcal{C} \approx \frac{|\mathcal{S}|}{\kappa/2}$$

Finding element in collision tree:

$$\mathcal{C} \approx \frac{|\mathcal{S}|}{i \cdot \kappa/4}$$

Consequences in Cryptography

Let $m = m_0||...||m_n||0||0||...||0$ (a message ending with z 0's) and $d = H(m)$.

Consequences in Cryptography

Let $m = m_0||...||m_n||0||0||...||0$ (a message ending with z 0's) and $d = H(m)$.

- If H is a T-sponge then we can find a preimage for d in time

$$2^c \cdot \frac{2^{r+2}}{\kappa \cdot z}.$$

Consequences in Cryptography

Let $m = m_0||...||m_n||0||0||...||0$ (a message ending with z 0's) and $d = H(m)$.

- If H is a T-sponge then we can find a preimage for d in time

$$2^c \cdot \frac{2^{r+2}}{\kappa \cdot z}.$$

- If H is a Davies-Meyer based hashfunctions with internal state size n:
  - With padding: preimage found in time $2^{n+1}/\kappa$
  - No padding: preimage found in time $2^{n+2}/(z \cdot \kappa)$

Consequences in Cryptography

Let $m = m_0||\ldots||m_n||0||0||\ldots||0$ (a message ending with z 0's) and $d = H(m)$.

- If H is a T-sponge then we can find a preimage for d in time

$$2^c \cdot \frac{2^{r+2}}{\kappa \cdot z}.$$

- If H is a Davies-Meyer based hashfunctions with internal state size n:
  - With padding: preimage found in time $2^{n+1}/\kappa$
  - No padding: preimage found in time $2^{n+2}/(z \cdot \kappa)$
- More?

Plan

UNIVERSITÉ DU LUXEMBOURG

Known Results

CPS and Iterated (Pre)-Images

Applications to Cryptography

Application to GLUON-64

Conclusion

Description of GLUON-64

- Lightweight hash function.

Description of GLUON-64

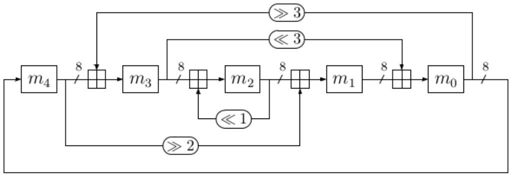
- Lightweight hash function.
- T-sponge, $r = 8$, $c = 128$

Description of GLUON-64

- Lightweight hash function.
- T-sponge, $r = 8$, $c = 128$
- $g = \Phi \circ \rho^{d+4} \circ \mathrm{pad}$

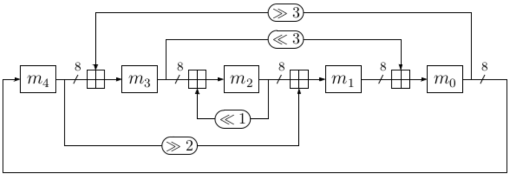
Description of GLUON-64

- Lightweight hash function.
- T-sponge, $r = 8$, $c = 128$
- $g = \Phi \circ \rho^{d+4} \circ \mathrm{pad}$



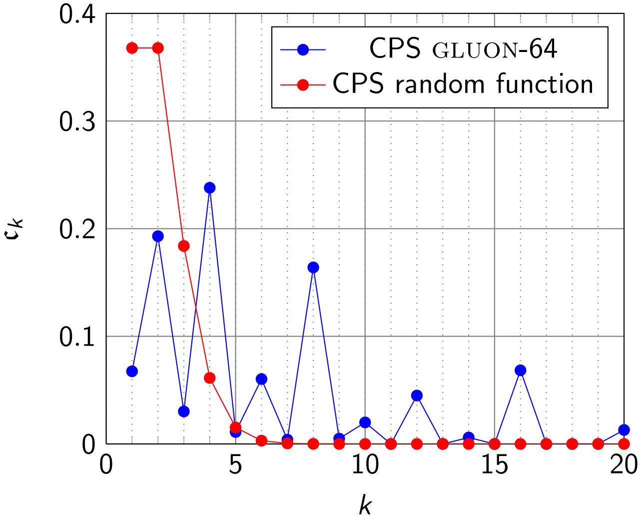
Description of GLUON-64

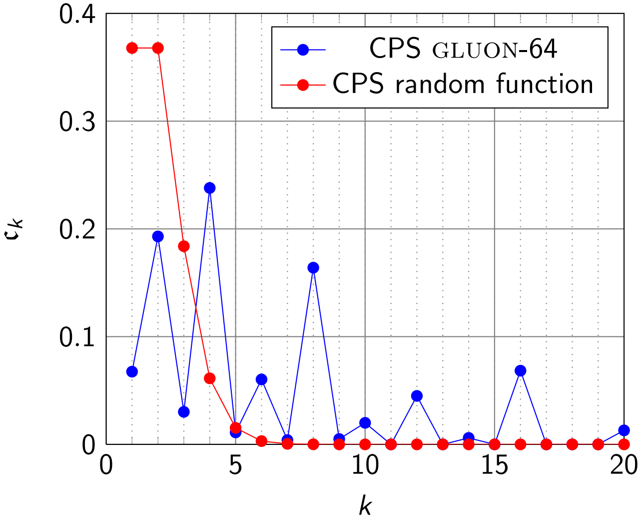
- Lightweight hash function.
- T-sponge, $r = 8$, $c = 128$
- $g = \Phi \circ \rho^{d+4} \circ \mathrm{pad}$



Possible (with a SAT-solver) to enumerate the solutions of

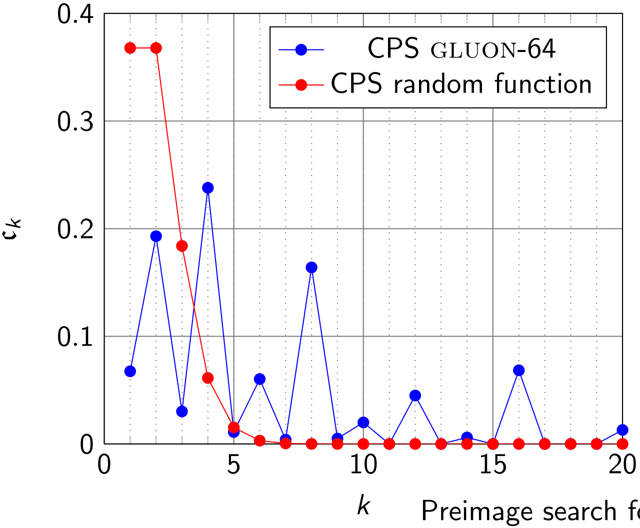
$$(\rho^{10} \circ \mathrm{pad})(x + a) = (\rho^{10} \circ \mathrm{pad})(a)$$


CPS GLUON-64
CPS random function
0.4
0.3
0.2
0.1
0
c_k
0
5
10
15
20
k



$\mathfrak{c}_1 = 0.065,\ \kappa = 6.982 = 2^{2.80}$

$\mathfrak{c}_1 = 0.368,\ \kappa = 1.000 = 2^{0}$



$\mathfrak{c}_1 = 0.065, \ \kappa = 6.982 = 2^{2.80}$

$\mathfrak{c}_1 = 0.368, \ \kappa = 1.000 = 2^0$

Preimage search for m ending with z zeroes:

$$\mathcal{C} = 2^c \cdot \frac{2^{r+2}}{\kappa \cdot z} = 2^c \cdot \frac{147}{z}$$

Plan

Known Results

CPS and Iterated (Pre)-Images

Applications to Cryptography

Application to GLUON-64

Conclusion

Conclusion

- Not permutation $\implies$ iterated output shrinking and quadratic collision trees.

Conclusion

- Not permutation $\implies$ iterated output shrinking and quadratic collision trees.
- The CPS allows more precise estimations of these.

Conclusion

- Not permutation $\implies$ iterated output shrinking and quadratic collision trees.
- The CPS allows more precise estimations of these.
- More precise flat sponge claim for T-sponge and easier preimage search $\implies$ Use large bitrate!

Conclusion

- Not permutation $\implies$ iterated output shrinking and quadratic collision trees.
- The CPS allows more precise estimations of these.
- More precise flat sponge claim for T-sponge and easier preimage search $\implies$ Use large bitrate!
- For GLUON-64, preimage for hash of unkown message ending with z zeroes needs $2^c \cdot (147/z)$.

Conclusion

- Not permutation $\implies$ iterated output shrinking and quadratic collision trees.
- The CPS allows more precise estimations of these.
- More precise flat sponge claim for T-sponge and easier preimage search $\implies$ Use large bitrate!
- For GLUON-64, preimage for hash of unkown message ending with z zeroes needs $2^c \cdot (147/z)$.
  - 500 B of zeroes: $\mathcal{C} = 2^{126.2}$

Conclusion

- Not permutation $\implies$ iterated output shrinking and quadratic collision trees.
- The CPS allows more precise estimations of these.
- More precise flat sponge claim for T-sponge and easier preimage search $\implies$ Use large bitrate!
- For GLUON-64, preimage for hash of unkown message ending with z zeroes needs $2^c \cdot (147/z)$.
  - 500 B of zeroes: $\mathcal{C} = 2^{126.2}$
  - 1 MB of zeroes: $\mathcal{C} = 2^{115.3}$

Conclusion

- Not permutation $\implies$ iterated output shrinking and quadratic collision trees.
- The CPS allows more precise estimations of these.
- More precise flat sponge claim for T-sponge and easier preimage search $\implies$ Use large bitrate!
- For GLUON-64, preimage for hash of unkown message ending with z zeroes needs $2^c \cdot (147/z)$.
    - 500 B of zeroes: $\mathcal{C} = 2^{126.2}$
    - 1 MB of zeroes: $\mathcal{C} = 2^{115.3}$
    - 1 GB of zeroes: $\mathcal{C} = 2^{105.3}$

Conclusion

- Not permutation $\implies$ iterated output shrinking and quadratic collision trees.
- The CPS allows more precise estimations of these.
- More precise flat sponge claim for T-sponge and easier preimage search $\implies$ Use large bitrate!
- For GLUON-64, preimage for hash of unkown message ending with z zeroes needs $2^c \cdot (147/z)$.
  - 500 B of zeroes: $\mathcal{C} = 2^{126.2}$
  - 1 MB of zeroes: $\mathcal{C} = 2^{115.3}$
  - 1 GB of zeroes: $\mathcal{C} = 2^{105.3}$

Thank you!