# Hash Functions that Avoid Computational Shortcuts

Ahto Buldas

University of Tartu / Tallinn University of Technology / Cybernetica AS

# Computations and Trees

$h\colon \{0,1\}^k \times \{0,1\}^k \to \{0,1\}^k$ – a binary operation.
$T^h(x_1, \ldots, x_N)$ – a tree with leaves $x_1, \ldots, x_N$. Each non-leaf vertex represents an $h$-operation. Each variable $x_i$ represents an element of $\{0,1\}^k$.

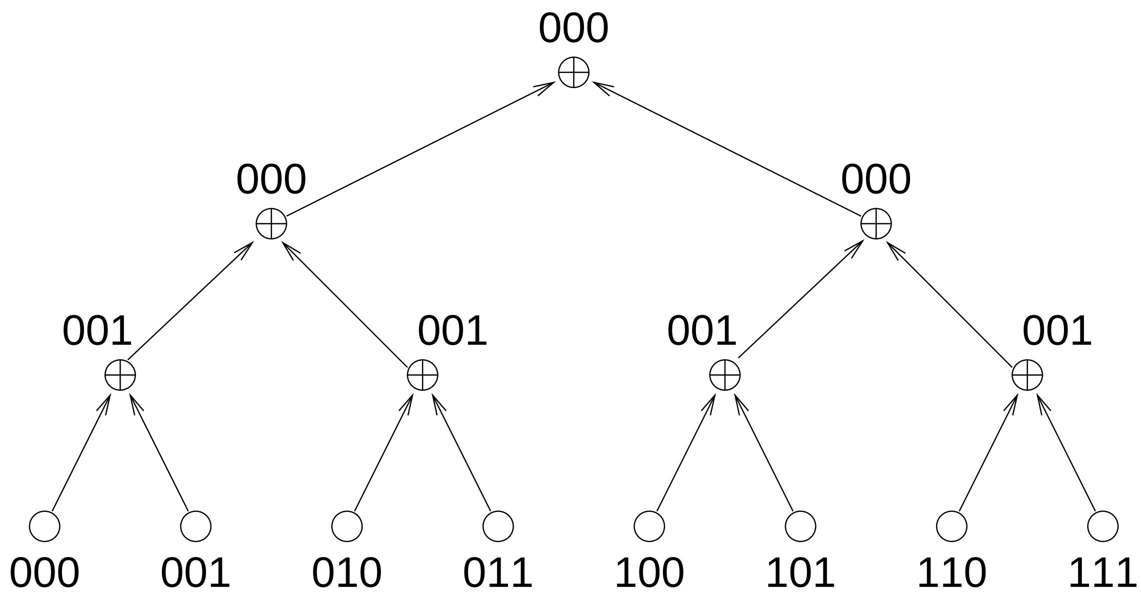
*Def.* A family of trees $T_k^h(v_1, \ldots, v_{N(k)})$ (where $v_i \in \{0,1\}^k$ are fixed) is said to be *hard to compute* if for every poly-time adversary A the following success probability is negligible:

$$\Pr[h \leftarrow \mathfrak{F}, r \leftarrow \mathsf{A}(1^k, h)\colon\ r = T_k^h(v_1, \ldots, v_{N(k)})]\ .$$

*Def. (Shortcut-Freeness)*: A function family $h\colon \{0,1\}^{2k} \to \{0,1\}^k$ is *shortcut-free* if every tree family $T_k^h(v_1, \ldots, v_N)$ with $\sharp\{v_1, \ldots, v_N\} = 2^k/k^{O(1)}$ is hard to compute.

# Not every function is shortcut-free ...

For example, if $h(x, y) = x \oplus y$ and $T_k^{\oplus}$ is the complete binary tree with $2^k$ leaves that represent all possible $k$-bit strings. This tree is called a *complete Merkle tree*.




We know that $T_k^{\oplus}(0, \ldots, 2^k - 1) = 0^k$, without doing any computations!

# Hash Functions and Hash Trees

Let $h = \{h_k\colon \{0,1\}^{2k} \to \{0,1\}^k\}_{k\in\mathbb{N}}$ be a poly-time computable family of functions that is chosen according to a distribution $\mathfrak{F}$.

*Def. (Collision-Resistance of $h$)*. For every poly-time adversary A:

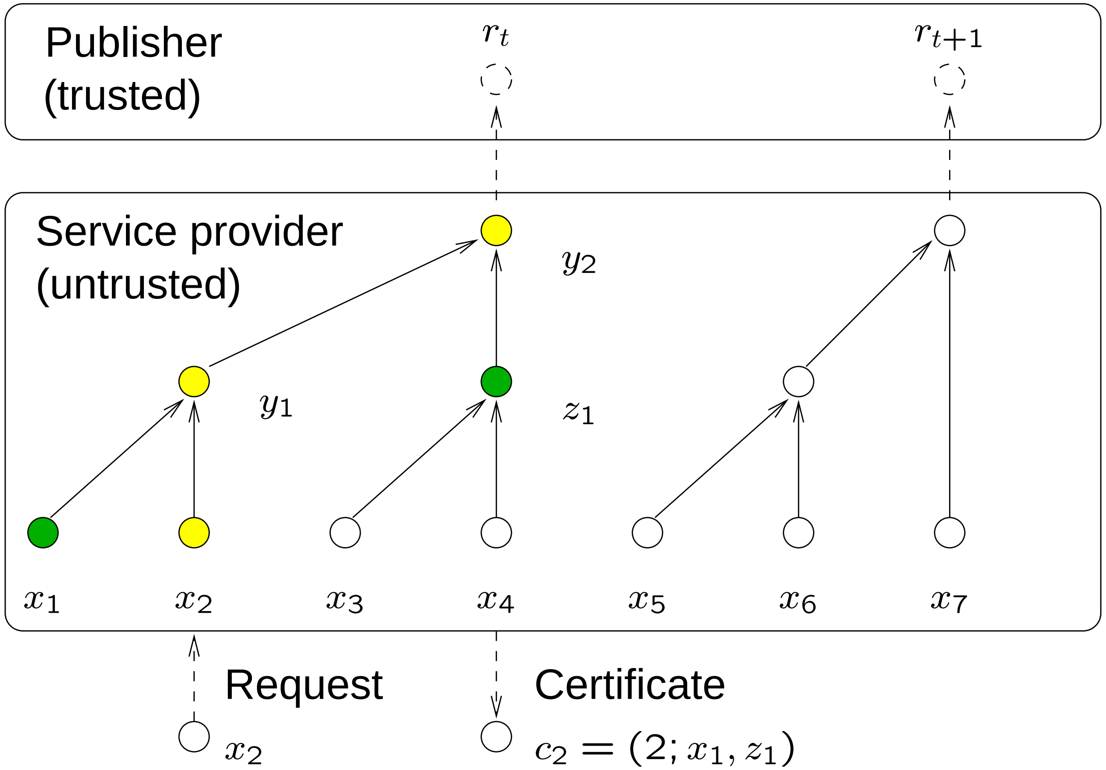
$$\Pr[h \leftarrow \mathfrak{F}, (x_1, x_2) \leftarrow \mathsf{A}(1^k, h)\colon\ x_1 \neq x_2,\ h(x_1) = h(x_2)] = k^{-\omega(1)}\ .$$

Nice overview on security properties of hash functions: see the recent paper by Rogaway and Shrimpton.

A conventional way to think is that cryptographic hash functions are short-cut free, mainly because they are often modelled as *random oracles*.

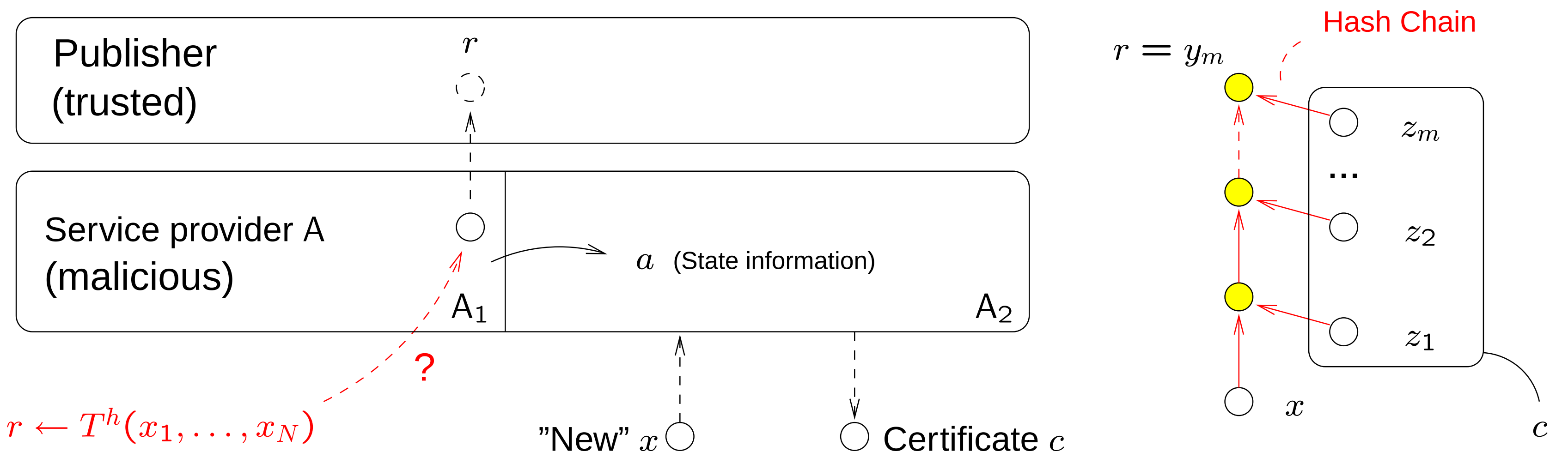
In principle, it is not excluded that shortcuts are possible in the case of cryptographic hash functions and this would affect the security of applications (like the time-stamping schemes currently in use).

# Hash-Tree Applications: Secure Registry




Verifying a certificate: Compute $y_2 = F_h(x_2; c_2) = h(h(x_1, x_2), z_1)$, obtain $r_t$, and check if $y_2 = r_t$.

# Back-Dating Attack




Successful forgery: $F_h(x; c) = r$

*Def. (Chain-Resistance of $h$)*. For every poly-time $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ and for every poly-sampleable distribution $\mathcal{D}$ with Rényi entropy $\mathsf{H}_2(\mathcal{D}) = \omega(\log k)$:

$$\Pr[(r, a) \leftarrow \mathsf{A}_1(1^k), x \leftarrow \mathcal{D}, c \leftarrow \mathsf{A}_2(x, a) \colon F_h(x, c) = r] = k^{-\omega(1)}.$$

# How to Construct Chain-Resistant Functions?

A recent negative result (Buldas et al, 2004):
"$h$ is collision-resistant $\Rightarrow$ $h$ is chain-resistant" cannot be proved in a (conventional) black-box way.

It is an open question whether chain-resistant functions can be constructed (in a black-box way) from the collision-resistant ones.

*First result of this work:* If $h$: $\{0,1\}^{2k} \rightarrow \{0,1\}^k$ is collision-resistant and shortcut-free, then $h$ is chain-resistant.

Still no idea how to construct shortcut-free functions...

*Second result of this work (a tiny step towards shortcut-freeness):* We construct a hash-function for which the complete Merkle tree is hard to compute.

# Proof of the First Result (a Sketch)

Let $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ be a chain-finding adversary for $h$ (a collision-resistant hash function) with success probability

$$\delta(k) = \mathsf{Pr}[(r,a) \leftarrow \mathsf{A}_1(1^k), x \leftarrow \mathcal{D}, c \leftarrow \mathsf{A}_2(x,a) \colon F_h(x,c) = r] \neq k^{-\omega(1)}.$$

We show that with high probability, there is a tree $T_k^h(v_1, \ldots, v_N) = r$ with $\sharp\{v_1, \ldots, v_N\} = 2^k / k^{O(1)}$. Collision-resistance is essential in this step!

Putting all trees $T_k^h$ together, we obtain a tree-family which is computable with non-negligible probability. Hence, $h$ is not shortcut-free.

# Proof of the Second Result (a Sketch)

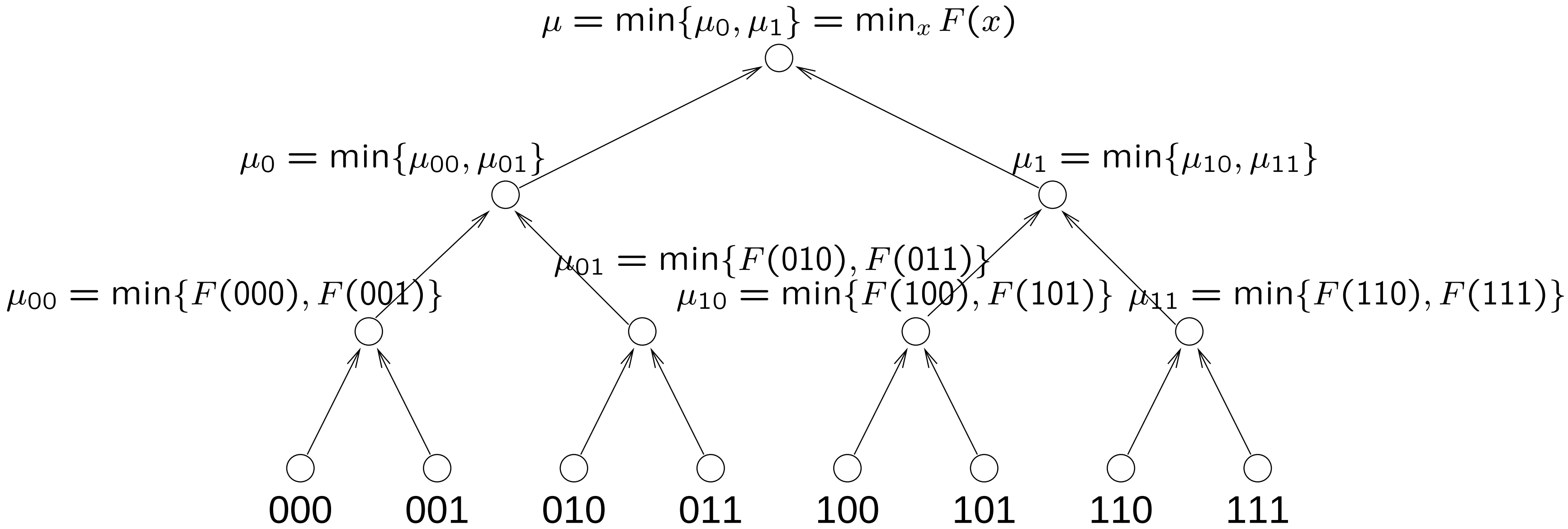
Let $h\colon \{0,1\}^* \to \{0,1\}^k$ be a collision-resistant hash function.

- We construct a new hash $H = P^h\colon \{0,1\}^{2n} \to \{0,1\}^n$, where $n = 6k$
- The root of the complete Merkle tree $\mathrm{M}^H$ contains (with high probability) a collision for $h$
- Hence, the root of $\mathrm{M}^H$ must be hard to compute, because $h$ is collision-free!

*Main idea of the construction:*

- Massive iteration of $H$ can be used to compute global minima and maxima of certain (cleverly chosen) functions $f^h\colon \{0,1\}^k \to \{0,1\}^k$
- Global minimum (maximum) operation can be used to invert $h$
- Inverting $h$ can be used to find collisions for $h$

# How to find global minimum for a function $F$?




Define: $H(x\|b_1, y\|b_2) \stackrel{\text{def.}}{=} \begin{cases} \min\{F(x), F(y)\}\|1 & \text{if } b_1 = b_2 = 0 \\ \min\{x, y\}\|1 & \text{if } b_1 = b_2 = 1 \\ 1^{k+1} & \text{otherwise.} \end{cases}$

Then $\mathrm{M}^H_{k+1}(0, \ldots, 2^{k+1} - 1) = \min_x F(x)$.

# Inverting $f$ by using max and min

For any $f\colon \{0,1\}^k \to \{0,1\}^k$ define functions $F_f^{\mathsf{min}}$ and $F_f^{\mathsf{max}}$ of type $\{0,1\}^{2k} \to \{0,1\}^k$ as follows:

$$F_f^{\mathsf{min}}(x,y) = \begin{cases} 1^k & \text{if } f(x) \neq y \\ x & \text{if } f(x) = y \end{cases} \qquad F_f^{\mathsf{max}}(x,y) = \begin{cases} 0^k & \text{if } f(x) \neq y \\ x & \text{if } f(x) = y \end{cases}$$

Let $y \in \{0,1\}^k$ be a fixed bitstring. It is clear that

$$\min_x F_f^{\mathsf{min}}(x,y) = \begin{cases} 1^k & \text{if } y \notin f(\{0,1\}^k) \\ \min f^{-1}(y) & \text{if } y \in f(\{0,1\}^k) \end{cases}$$

and

$$\max_x F_f^{\mathsf{max}}(x,y) = \begin{cases} 0^k & \text{if } y \notin f(\{0,1\}^k) \\ \max f^{-1}(y) & \text{if } y \in f(\{0,1\}^k) \end{cases}$$

# Finding collisions for $h$ by using min and max

Take two distinct bit-strings $c_1, c_2 \in \{0,1\}^k$ and try to invert $f_1(\cdot) = h(\cdot, c_1)$ and $f_2(\cdot) = h(\cdot, c_2)$ realtive to $x' \leftarrow \{0,1\}^k$. For $f_1$ we obtain

$$x_1^{\mathsf{min}} = \min_x F_{f_1}^{\mathsf{min}}(x, f_1(x')), \qquad x_1^{\mathsf{max}} = \max_x F_{f_1}^{\mathsf{max}}(x, f_1(x')) \ .$$

With probability 1, $f_1(x') = f(x_1^{\mathsf{min}}) = f(x_1^{\mathsf{max}})$.

In case both $f_1$ and $f_2$ are “almost permutations”, i.e.

$$\Pr[|\, f_1^{-1}(f_1(x'))\,| \geq 2] = k^{-\omega(1)} \qquad \text{and} \Pr[|\, f_1^{-1}(f_1(x'))\,| \geq 2] = k^{-\omega(1)}$$

then with high probability, $f_1$ and $f_2$ can be inverted simultaneously on a uniformly selected output $y \leftarrow \{0,1\}^k$.

All in all, the probability of finding a collision for $h$ is at least $\frac{1}{3}$.

# Construction of $H$

Let $z \in \{0,1\}^k$ and for $i = 1,2$ define

$$\varphi_z^{i,\mathrm{min}}(x) = \begin{cases} 1^k & \text{if } f_i(x) \neq z \\ x & \text{if } f_i(x) = z. \end{cases} \qquad \varphi_z^{i,\mathrm{max}}(x) = \begin{cases} 0^k & \text{if } f_i(x) \neq z \\ x & \text{if } f_i(x) = z. \end{cases}$$

For $i = 1,2$ define $h_z^{i,\mathrm{min}}\colon \{0,1\}^{2(k+1)} \to \{0,1\}^{k+1}$ as follows:

$$h_z^{i,\mathrm{min}}(x\|b_1, y\|b_2) = \begin{cases} \min\{\varphi_z^{i,\mathrm{min}}(x), \varphi_z^{i,\mathrm{min}}(y)\}\|1 & \text{if } b_1 = b_2 = 0 \\ \min\{x,y\}\|1 & \text{if } b_1 = b_2 = 1 \\ 1^{k+1} & \text{otherwise.} \end{cases}$$

$$h_z^{i,\mathrm{max}}(x\|b_1, y\|b_2) = \begin{cases} \min\{\varphi_z^{i,\mathrm{max}}(x), \varphi_z^{i,\mathrm{max}}(y)\}\|1 & \text{if } b_1 = b_2 = 1 \\ \min\{x,y\}\|0 & \text{if } b_1 = b_2 = 0 \\ 0^{k+1} & \text{otherwise.} \end{cases}$$

Define: $H_z = h_{f(z)}^{1,\mathrm{min}} \times h_{f(z)}^{1,\mathrm{max}} \times h_{f(z)}^{2,\mathrm{min}} \times h_{f(z)}^{2,\mathrm{max}} \times h_z^{1,\mathrm{min}} \times h_z^{2,\mathrm{max}}$

# Conclusions

There seem to be no easy ways of ”abusing” non-complete hash trees $T^H$ for finding collisions for $h$ in a similar way ...

How to construct $H = P^h$ so that a massive iteration of $H$ always (or with high probability) gives a collision for $h$?

Can we find ”natural” (local,statistical,...) properties of $h$ that (together with collision-resistance) imply chain-resistance.