Advanced Cryptography
CS 655

Week 11:

- Indistinguishability Obfuscation + Applications

Course Project Report: Due Thursday, March 23 @ 11:59PM via E-mail

Spring 2023

Course Progress Report

- **Due:** Thursday, March 23 @ 11:59PM via E-mail
- **Pages:** 5-6
- **Contents:**
  - Motivation
  - Define the problem(s) you are working on clearly
  - Related Work
  - Preliminary Results
    - What have you tried?
    - What barriers have you encountered (if any)?

Obfuscation

- An obfuscator takes as input a program/circuit C and a security parameter λ and outputs a new program/circuit $C' = \mathrm{Obf}\left(1^{\lambda}, C\right)$

- **Efficiency:** The function obfuscate should run in polynomial time in the size of the input program/circuit |C| and in the size of security parameter λ

- **Correctness:** C' should be equivalent to C i.e., for all inputs x we have
$$C'(x) = C(x)$$

Security?

Virtual Blackbox Obfuscation

- **VBB Security Definition:** For all PPT attackers $\mathcal{A}$ there exists a simulator $\mathcal{S}$ such that for all programs $\{P_n\}$ and all security parameters λ
$$\left|\Pr\left[\mathcal{A}\left(\mathrm{Obf}(1^\lambda, P_n)\right)\right] - \Pr[\mathcal{S}^{P_n(\cdot)}(1^\lambda, |P_n|)]\right| \leq \mathrm{negl}(\lambda)$$

- **Intuition:** Anything an attacker could learn from the description of the obfuscated circuit $C' = \mathrm{Obf(C)}$ the attacker could have learned if they had oracle access to the circuit $C(x)$ as a blackbox

- **Pro:** Very strong security notion for obfuscation! ☺
- **Con:** Impossible to achieve ☹

Virtual Blackbox Obfuscation

- **VBB Security Definition:** For all PPT attackers $\mathcal{A}$ there exists a simulator $\mathcal{S}$ such that for all programs $\{P_n\}$ and all security parameters λ

$$\left|\Pr\left[\mathcal{A}\left(\mathrm{Obf}(1^{\lambda}, P_n)\right)\right] - \Pr[\mathcal{S}^{P_n(\cdot)}(1^{\lambda}, |P_n|)]\right| \leq \mathrm{negl}(\lambda)$$

- **Impossibility:** Let $\alpha, \beta, \gamma \in \{0,1\}^{\lambda}$ be uniformly random strings and define the following program

$$P_{\alpha,\beta,\gamma}(x) = \begin{cases} \beta & \text{if } x = \alpha \\ \gamma & \text{if } x(\alpha) = \beta \\ \perp & \text{otherwise} \end{cases}$$

View string x as description of a program. $x(\alpha)$ denotes the output of this program on input α

Virtual Blackbox Obfuscation

- **Impossibility:** Let $\alpha, \beta, \gamma \in \{0,1\}^\lambda$ be uniformly random strings and define the following program

$$P_{\alpha,\beta,\gamma}(x) = \begin{cases} \beta & \text{if } x = \alpha \\ \gamma & \text{if } x(\alpha) = \beta \\ \perp & \text{otherwise} \end{cases}$$

- **Observation 1 (blackbox queries hide $\boldsymbol{\alpha, \beta, \gamma}$):** If $\alpha, \beta, \gamma \in \{0,1\}^\lambda$ are uniformly random and $\mathcal{S}^{P_n(\cdot)}$ makes at most q queries then all of the responses will be ⊥ except with probability $2q2^{-\lambda}$

(Proof Sketch)

- $\Pr[x_i = \alpha \mid P_{\alpha,\beta,\gamma}(x_1) = \cdots = P_{\alpha,\beta,\gamma}(x_{i-1}) = \perp] \leq 2^{-\lambda}$
- $\Pr[x_i(\alpha) = \beta \mid P_{\alpha,\beta,\gamma}(x_1) = \cdots = P_{\alpha,\beta,\gamma}(x_{i-1}) = \perp] \leq 2^{-\lambda}$

Virtual Blackbox Obfuscation

- **Impossibility:** Let $\alpha, \beta, \gamma \in \{0,1\}^\lambda$ be uniformly random strings and define the following program

$$P_{\alpha,\beta,\gamma}(x) = \begin{cases} \beta & \text{if } x = \alpha \\ \gamma & \text{if } x(\alpha) = \beta \\ \perp & \text{otherwise} \end{cases}$$

- **Observation 2** (easy to extract α, β, γ from any obfuscation of $P_{\alpha,\beta,\gamma}$)
  - Let $P = \mathrm{Obf}\left(1^\lambda, P_{\alpha,\beta,\gamma}(x)\right)$ and consider running P on input P.

$$P(P) = \underbrace{P_{\alpha,\beta,\gamma}(P)}_{\text{Obfuscation correctness}} = \gamma$$

Since $P(\alpha) = P_{\alpha,\beta,\gamma}(\alpha) = \beta$

Definition of $P_{\alpha,\beta,\gamma}$

Obfuscation correctness

VBB Impossibility for Circuits

- Challenge: Cannot feed circuit as input to itself
- Impossibility for Circuits given Fully Homomorphic Encryption

- $C_{\alpha,\beta,\gamma}(x) = \begin{cases} \mathrm{Enc}_{\mathrm{pk}}(\alpha) & \text{if } x = 0 \\ \beta & \text{if } x = \alpha \\ \gamma & \text{if } \mathrm{Dec}_{\mathrm{sk}}(x) = \beta \\ \perp & \text{otherwise} \end{cases}$

- **Observation 1:** Oracle access to $C_{\alpha,\beta,\gamma}$ will still hide α, β, γ
- **Observation 2:** Given $\mathrm{C}' = \mathrm{Obf}(C_{\alpha,\beta,\gamma})$ we can extract $\mathrm{Enc}_{\mathrm{pk}}(\alpha)$ and then obtain an encryption $\mathrm{Enc}_{\mathrm{pk}}(\beta)$ of β by evaluating C' homomorphically on $\mathrm{Enc}_{\mathrm{pk}}(\alpha)$. Finally we can run $\mathrm{C}'\left(\mathrm{Enc}_{\mathrm{pk}}(\beta)\right) = \gamma$

Indistinguishability Obfuscation

Two circuits C and C' are equivalent if

1) They have the same size i.e.,$|C| = |C'|$ for all n
2) They have equivalent input/output behavior i.e., for all inputs x we have

$$C(x) = C'(x)$$

Definition: For all pairs of equivalent circuits and all PPT distinguishers $\mathcal{A}$ we have

$$\left|\Pr\left[\mathcal{A}\left(\mathrm{iO}(1^\lambda, C)\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathrm{iO}\left(1^\lambda, C'(x)\right)\right) = 1\right]\right| \leq \mathrm{negl}(\lambda)$$

Con: Weaker Promise ☹

- **Pro:** Achievable! ☺
  - **Con:** Current constructions are not practically efficient. ☹
- **Pro:** Still very useful ☺

Indistinguishability Obfuscation: Best Possible

- ``On Best Possible Obfuscation” [TCC’07]

- Suppose obfuscator iO satisfies security notion of Indististinguishability Obfuscation
- Suppose obfuscator Obf satisfies some other security notion
- Observe that

1) $\mathrm{Obf}'(\mathrm{C}) \coloneqq \mathrm{iO}(\mathrm{Obf}(\mathrm{C}))$ cannot be weaker obfuscation scheme than Obf

2) $\mathrm{C}' = \mathrm{Obf}(\mathrm{C})$ is functionally equivalent to C

3) C' is equivalent to Pad(C) i.e., pad description length of C so that circuits have the same size

4) $\mathrm{Obf}'(\mathrm{C}) \coloneqq \mathrm{iO}(\mathrm{Obf}(\mathrm{C}))$ is indistinguishable from $\mathrm{iO}(\mathrm{Pad}(\mathrm{C}))$ (by IO security)

Indistinguishability Obfuscation

- Constructing iO:
  - ``Candidate Indistinguishability Obfuscation and Functional Encryption for all Circuits” [FOCS’13]
  - Many constructions based on new assumptions
    - Many papers breaking assumptions (or constructions!) and many fixes
  - Recent progress “Indistinguishability Obfuscation from Well-Founded Assumptions.” [STOC’21]
  - Constructs iO from sub-exponential security of well studies crypto assumptions
    - Learning With Errors (LWE), Learning Parity with Noise (LPN) over prime fields, PRG in NC0, and Decision Linear (DLIN) assumption for symmetric bilinear groups of prime order
- Applications of iO (our focus): Witness Encryption, Short Signatures, Proofs of Human Work, Universal Samplers, ….

Powerful Tool for iO: Puncturable PRF

- Three algorithms KeyGen, Puncture, and Eval

- $F_K(x) \coloneqq Eval(K, x)$ is a pseudorandom function

- $\mathrm{Puncture}(\mathrm{K}, \mathrm{x}')$ takes as input a key K and an input x and outputs a new punctured key $K\{x'\}$
  - **Correctness:** $\mathrm{Eval}(\mathrm{K}\{\mathrm{x}'\}, \mathrm{x}) = \mathrm{Eval}(\mathrm{K}, \mathrm{x})$ for all inputs $x \neq x'$ and $\mathrm{Eval}(\mathrm{K}\{\mathrm{x}'\}, \mathrm{x}') = \perp$
  - **Security:** $\mathrm{K}\{\mathrm{x}'\}$ leaks no information about $F_K(x)$ i.e., all PPT distinguishers $\mathcal{A}$ we have

$$\left|\Pr\left[\mathcal{A}\left(\mathrm{K}\{\mathrm{x}'\}, F_K(x')\right) = 1\right] - \Pr[\mathcal{A}(\mathrm{K}\{\mathrm{x}'\}, r) = 1]\right| \leq \mathrm{negl}(\lambda)$$

(where r is a random string)

- **Intuition:** $K\{x'\}$ allows us to evaluate $F_K(x)$ on all inputs $x \neq x'$ except for x' while ensuring that $F_K(x')$ is still indistinguishable from random.

GGM: PRFs from PRGs

Theorem: Suppose that there is a PRG G with expansion factor $\ell(\boldsymbol{\lambda}) = 2\boldsymbol{\lambda}$. Then there is a secure PRF.

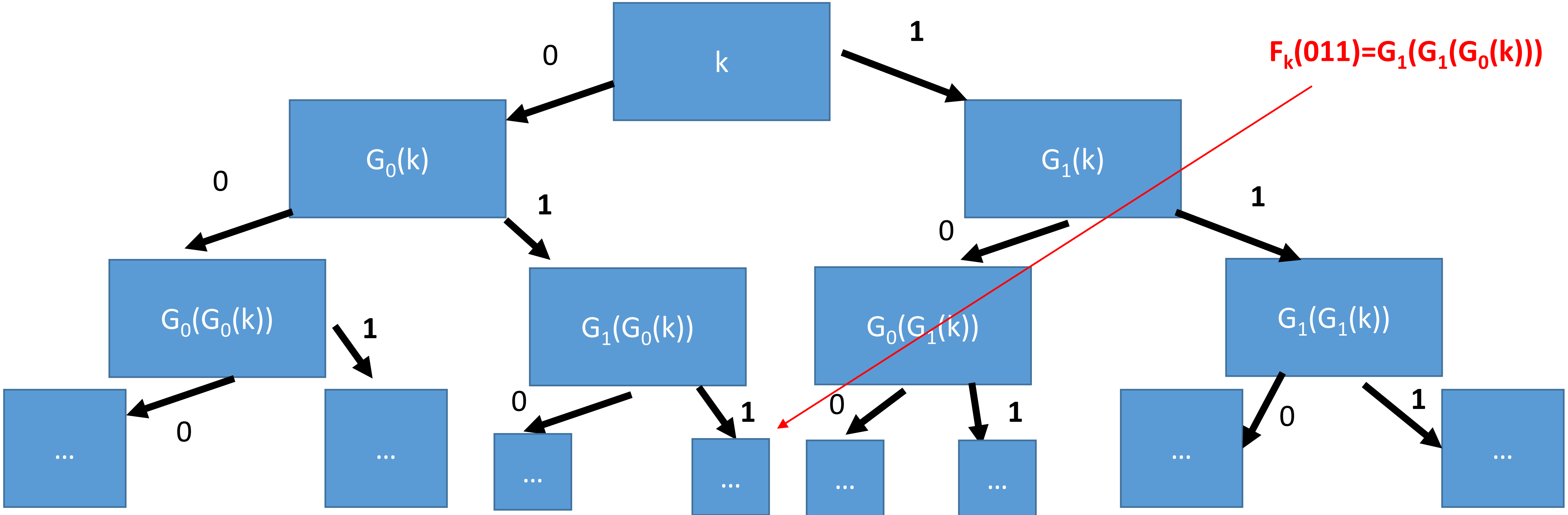
Let $\mathrm{G(x) = G_0(x)||G_1(x)}$ (first/last λ bits of output)

$$\boldsymbol{F_K(x_1, \dots, x_n) = G_{x_n}\left(\dots\left(G_{x_2}\left(G_{x_1}(K)\right)\right)\dots\right)}$$

PRFs from PRGs

$$G(x) := \overbrace{G_0(x)}^{\text{n-bits}} \,||\, \overbrace{G_1(x)}^{\text{n-bits}}$$

Theorem: Suppose that there is a PRG G with expansion factor $\ell(\lambda) = 2\lambda$. Then there is a secure PRF.



PRFs from PRGs

Theorem: Suppose that there is a PRG G with expansion factor $\ell(\boldsymbol{\lambda}) = 2\boldsymbol{\lambda}$. Then there is a secure PRF.

Proof:

Claim 1: For any t(n) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(\lambda)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(\lambda)}}\right)\right)\right]\right| < \boldsymbol{negl}(\boldsymbol{\lambda})$$

PRFs from PRGs

Claim 1: For any t(λ) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}[\boldsymbol{A}(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(\lambda)}})] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}(\boldsymbol{s_{t(\lambda)}})\right)\right]\right| < \boldsymbol{negl}(\lambda)$$

Proof Sketch (by Triangle Inequality): Fix j

$$\boldsymbol{Adv_j} = \left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{j+1}} \parallel \boldsymbol{G}(\boldsymbol{s_{j+2}}) \ldots \parallel \boldsymbol{G}(\boldsymbol{s_{t(\lambda)}})\right)\right]\right|$$

PRFs from PRGs

Claim 1: For any t(n) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(\lambda)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(\lambda)}}\right)\right)\right]\right| < \boldsymbol{negl}(\boldsymbol{\lambda})$$

Proof Sketch

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(\lambda)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(\lambda)}}\right)\right)\right]\right|$$

$$\leq \sum_{j<t(\lambda)} \boldsymbol{Adv_j}$$

$$\leq \boldsymbol{t}(\boldsymbol{\lambda}) \times \boldsymbol{negl}(\boldsymbol{\lambda}) = \boldsymbol{negl}(\boldsymbol{\lambda}) \qquad (\boldsymbol{QED})$$

PRFs from PRGs

Claim 1: For any t(λ) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(\lambda)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(\lambda)}}\right)\right)\right]\right| < \boldsymbol{negl}(\boldsymbol{\lambda})$$

Proof

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(\lambda)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(\lambda)}}\right)\right)\right]\right|$$

$$\leq \sum_{j<t(\lambda)} \boldsymbol{Adv_j}$$

$$\leq \boldsymbol{t}(\boldsymbol{\lambda}) \times \boldsymbol{negl}(\boldsymbol{\lambda}) = \boldsymbol{negl}(\boldsymbol{\lambda})$$

PRFs from PRGs

Claim 1: For any t(n) and any PPT attacker A we have

$$\left|Pr\left[A\left(r_1 \parallel \cdots \parallel r_{t(n)}\right)\right] - Pr\left[A\left(G(s_1) \parallel \cdots \parallel G\left(s_{t(n)}\right)\right)\right]\right| < negl(n)$$

Proof

$$\left|Pr\left[A\left(r_1 \parallel \cdots \parallel r_{t(n)}\right)\right] - Pr\left[A\left(G(s_1) \parallel \cdots \parallel G\left(s_{t(n)}\right)\right)\right]\right|$$

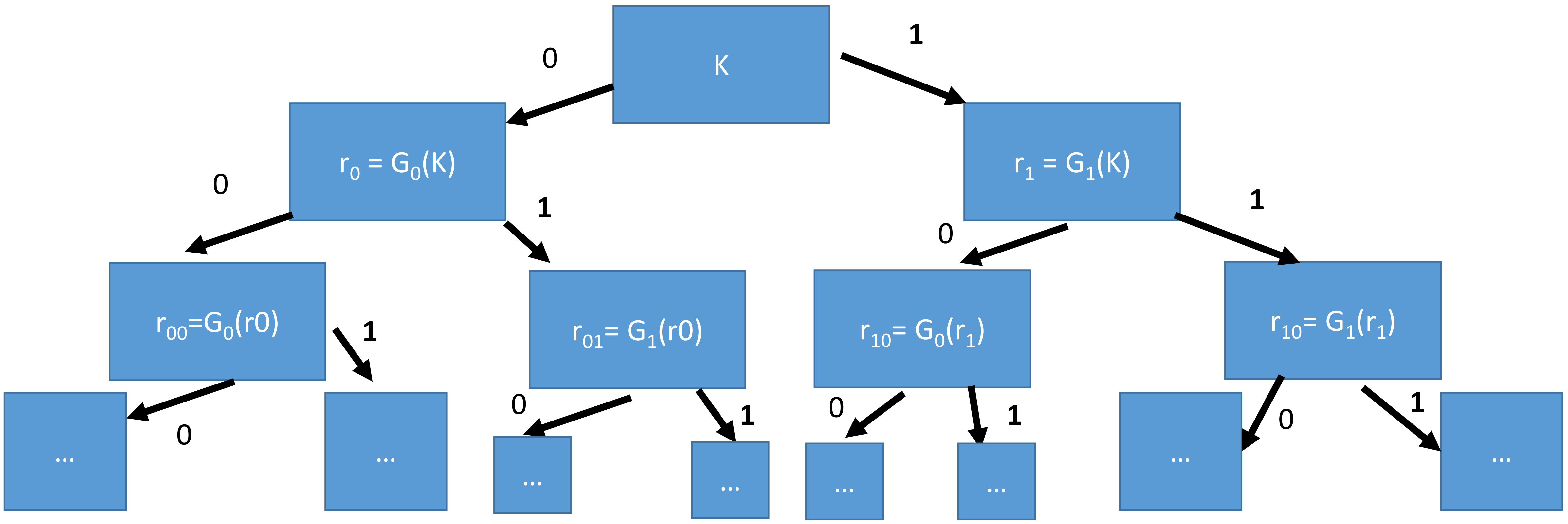
$$\leq \sum_{j<t(n)} Adv_j$$

$$\leq t(n) \times negl(n) = negl(n)$$

(QED, Claim 1)

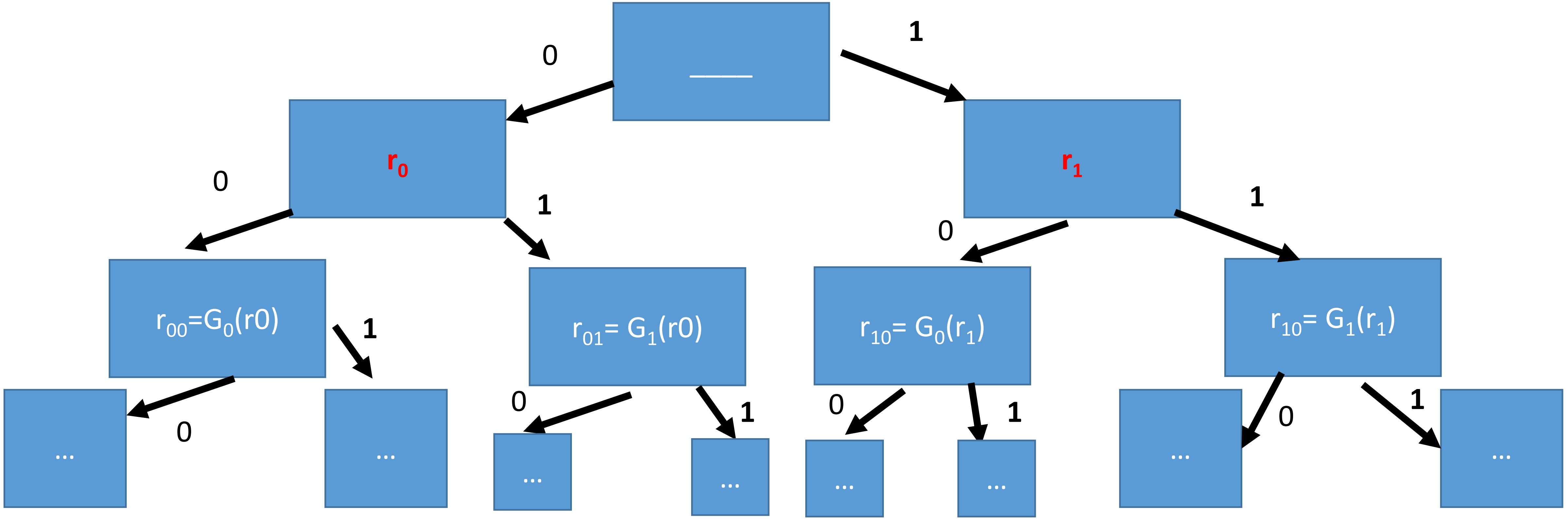
Hybrid H_1 and H_2

- Original Construction: Hybrid H_1



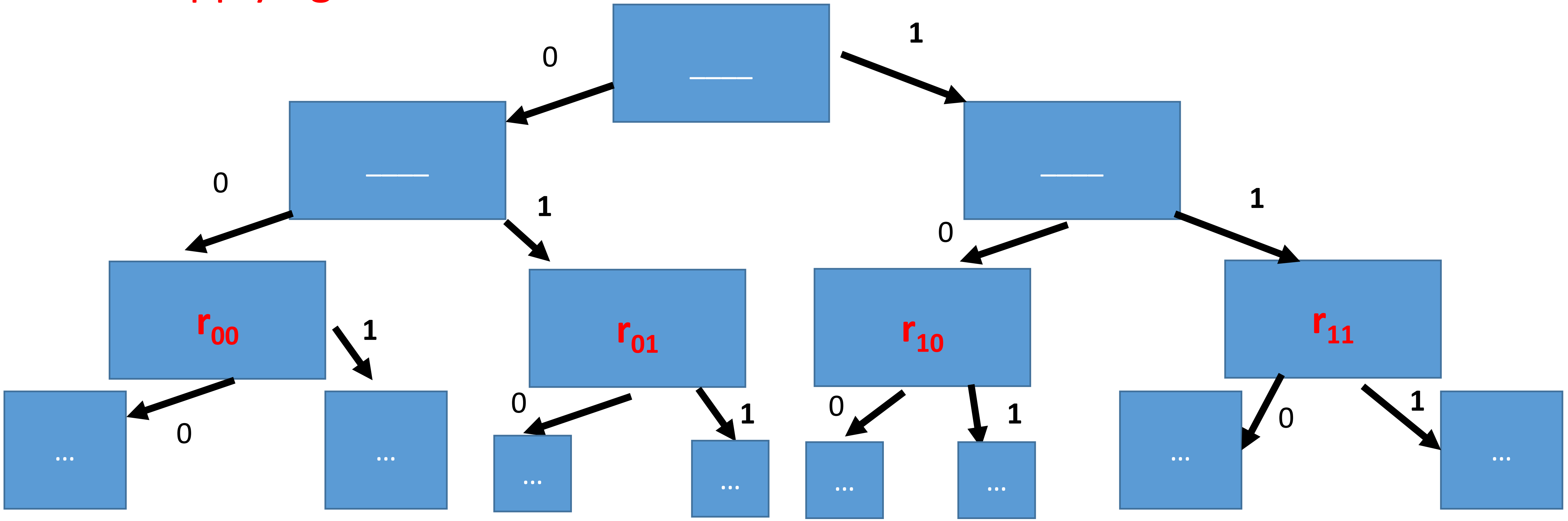
Hybrid H_1 and H_2

- Modified Construction H_2: Pick r_0 and r_1 randomly instead of $r_i = G_i(K)$



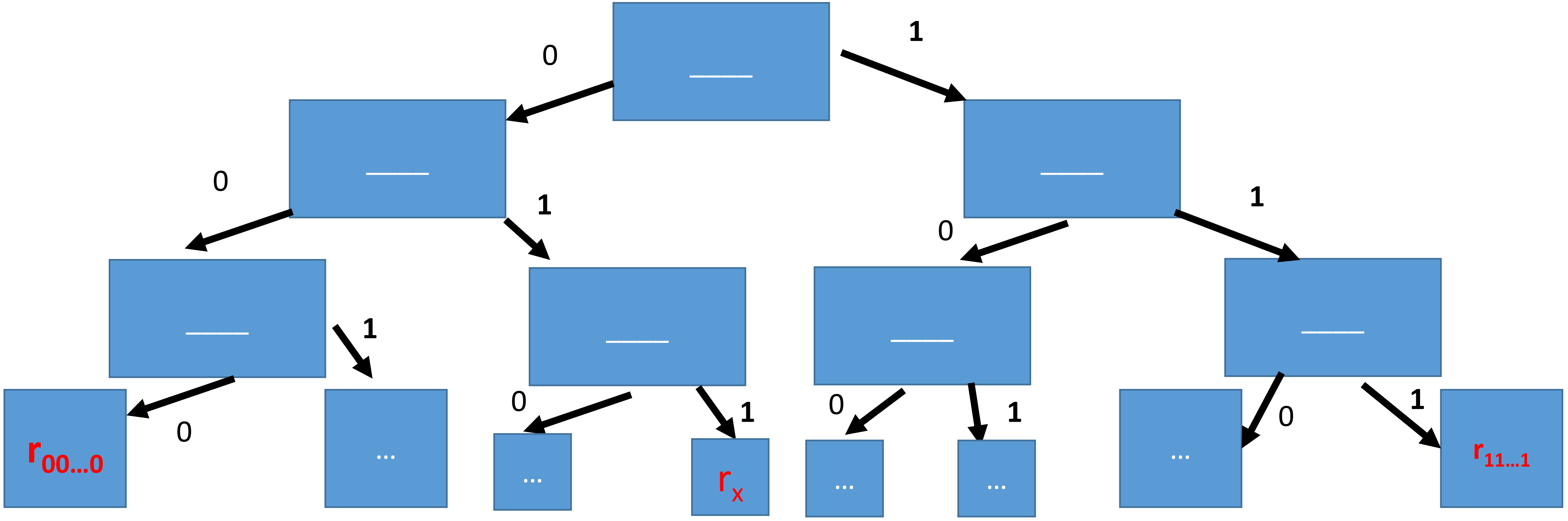
Hybrid H_3

- Modified Construction H_3: Pick r_{00} , r_{01} , r_{10} and r_{11} randomly instead of applying PRG



Hybrid H_n

- Truly Random Function: All output values r_x are picked randomly



Hybrid H_1 vs H_2

Claim 1: For any t(n) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(n)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(n)}}\right)\right)\right]\right| < \boldsymbol{negl}(\boldsymbol{\lambda})$$

Claim 2: ***Attacker who makes t(λ) queries to F_k (or f) cannot distinguish H_2 from the real game (except with negligible probability).***

Proof Intuition: Follows by Claim 1

Hybrid H_i vs H_i

Claim 1: For any t(n) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(\lambda)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(\lambda)}}\right)\right)\right]\right| < \boldsymbol{negl}(\boldsymbol{n})$$

Claim 3: Attacker who makes t(n) queries to F_k (or f) cannot distinguish H_i from H_{i-1} the real game (except with negligible probability).

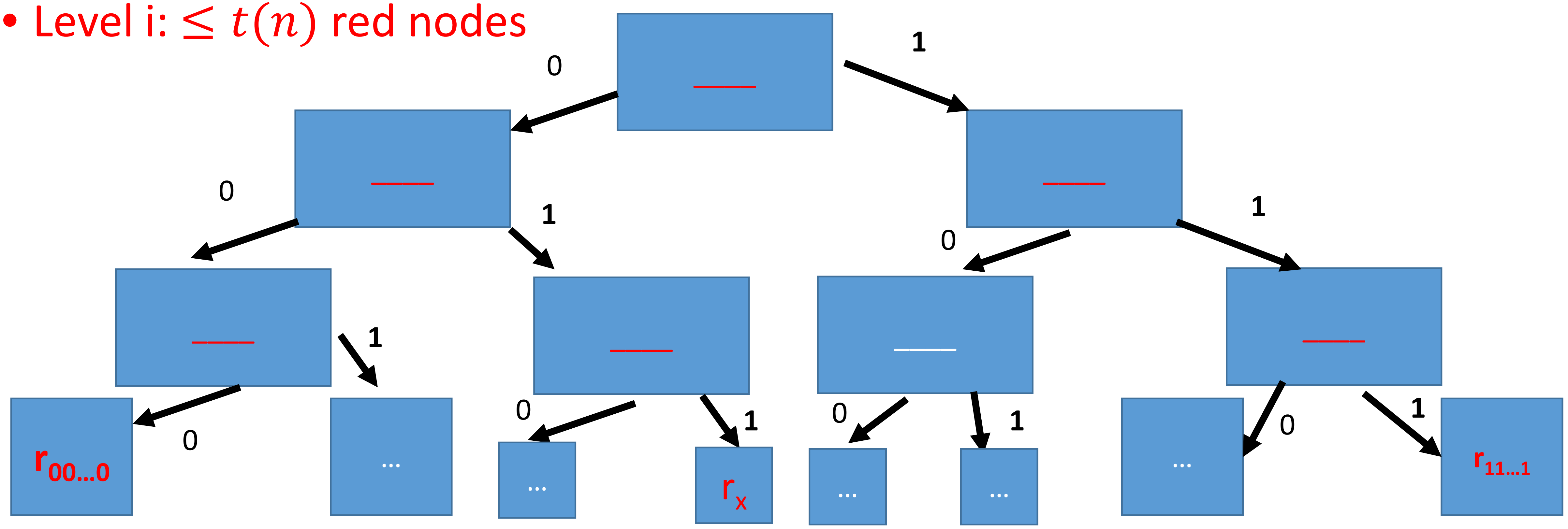
Challenge: Cannot replace 2^i pseudorandom values with random strings at level i

2^i negl($\boldsymbol{\lambda}$) is not necessarily negligible if $i = \frac{\boldsymbol{\lambda}}{2}$

Key Idea: Only need to replace t($\boldsymbol{\lambda}$) values (note: $t(\boldsymbol{\lambda})$negl($\boldsymbol{\lambda}$) is negligible).

Hybrid H_i

- Red Leaf Nodes: Queried $F_k(x)$ (at most t(n) red leaf nodes)
- Red Internal Nodes: On path from red leaf node to root
- Level i: $\leq t(n)$ red nodes



Hybrid H_1 vs H_2

Claim 1: For any t(n) and any PPT attacker A we have

$$\left|Pr[A(r_1 \parallel \cdots \parallel r_{t(\lambda)})] - Pr\left[A\left(G(s_1) \parallel \cdots \parallel G(s_{t(\lambda)})\right)\right]\right| < negl(\lambda)$$

Claim 2: *Attacker who makes $t(\lambda)$ oracle queries to our function cannot distinguish H_i from H_{i+1} (except with negligible probability).*

Proof: Indistinguishability follows by Claim 1

Let $x_1,\ldots x_t$ denote the t queries. Let $y_1,\ldots,y_t$ denote first i bits of each query.

(H_{i+1} vs H_i: replaced $G(r_{y_i})$ with $r_{y_i\parallel 0} \parallel r_{y_i\parallel 1}$)

Hybrid H_i vs H_i

Claim 1: For any t(λ) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(\lambda)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(\lambda)}}\right)\right)\right]\right| < \boldsymbol{negl}(\boldsymbol{\lambda})$$

Claim 3: Attacker who makes t($\boldsymbol{\lambda}$) queries to F_k (or f) cannot distinguish H_i from H_{i-1} the real game (except with negligible probability).

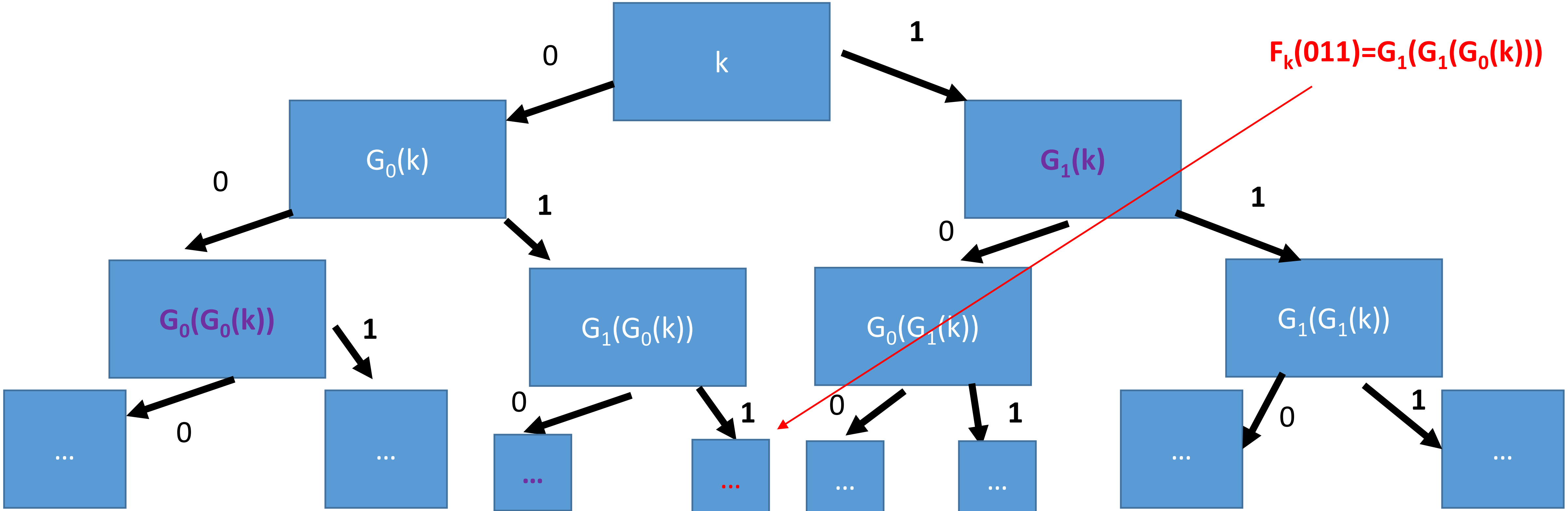
Triangle Inequality: Attacker who makes t($\boldsymbol{\lambda}$) queries to F_k (or f) *cannot* distinguish H_1 (real construction) from H_n (truly random function) except with negligible probability.

Punctured Key (Example)

$$G(x) := \underbrace{G_0(x)}_{\text{n-bits}} \,||\, \underbrace{G_1(x)}_{\text{n-bits}}$$

$K\{011\} = G_1(k)$, $G_0(G_0(k))$ and $G_0(G_1(G_0(k)))$

$G_1(k)$ ➔ Can evaluate $F_k(1x')$ for any input x' in $\{0,1\}^2$

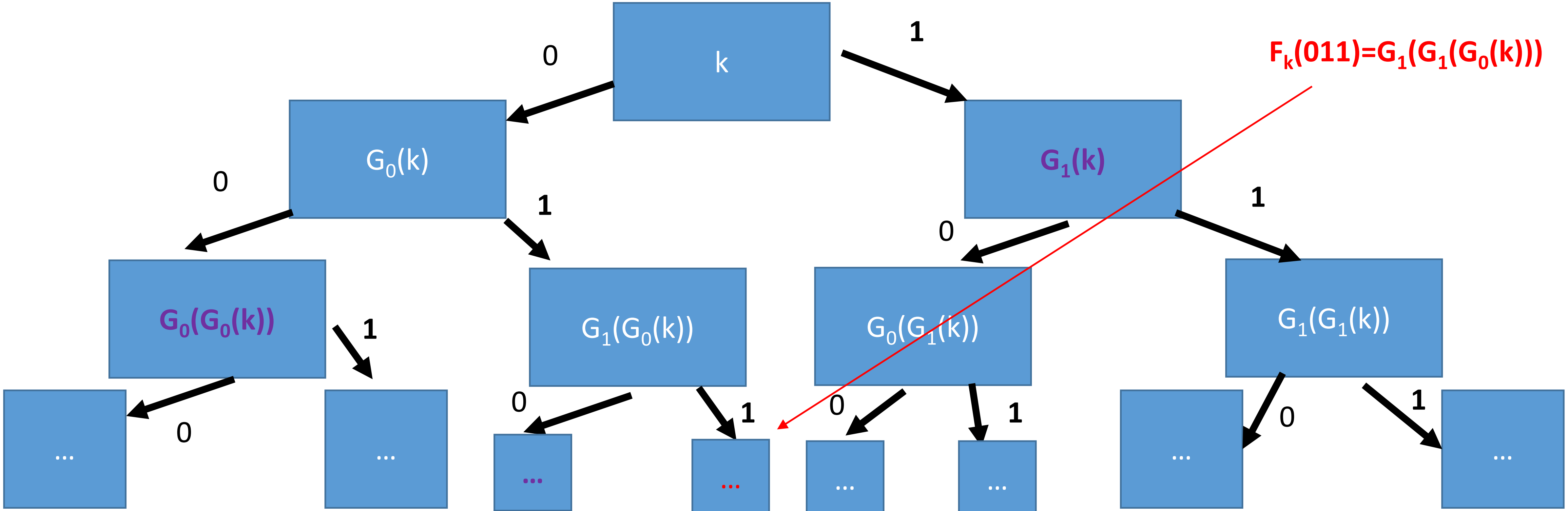


Punctured Key (Example)

$$G(x) := \underbrace{G_0(x)}_{\text{n-bits}} \,||\, \underbrace{G_1(x)}_{\text{n-bits}}$$

$K\{011\} = G_1(k)$, $G_0(G_0(k))$ and $G_0(G_1(G_0(k)))$

$G_0(G_0(k))$ ➔ Can evaluate $F_k(00x')$ for any bit x'

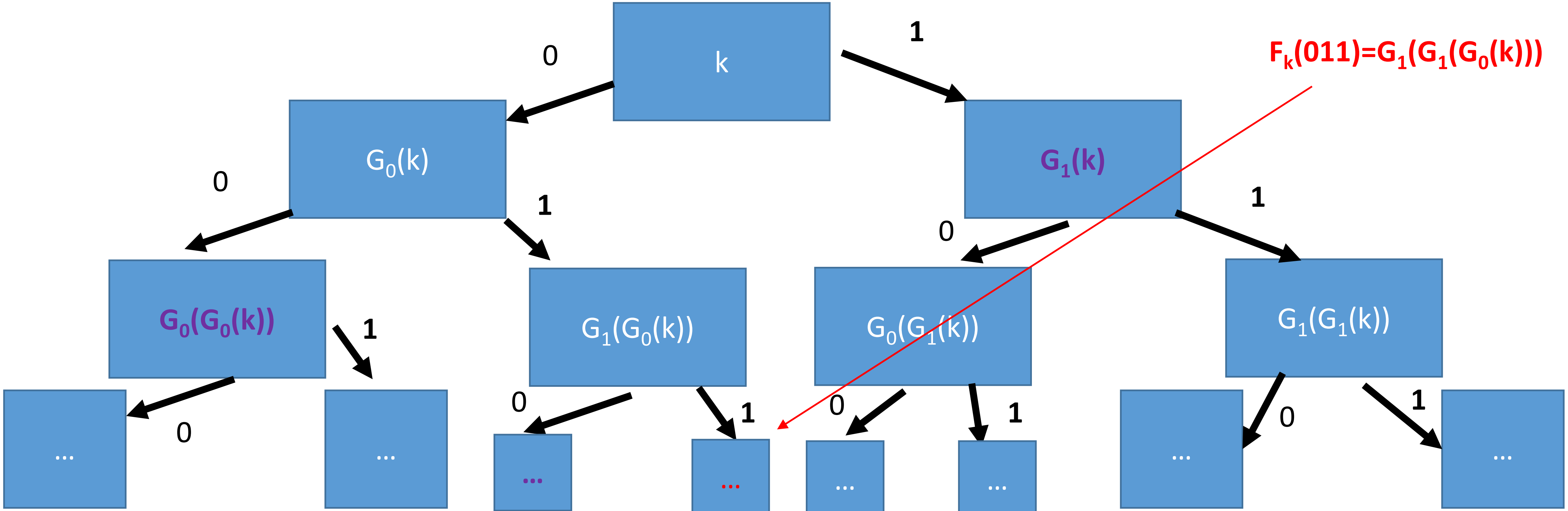


Punctured Key (Example)

$$G(x) := \underbrace{G_0(x)}^{\text{n-bits}} \,||\, \overbrace{G_1(x)}^{\text{n-bits}}$$

$K\{011\} = G_1(k)$, $G_0(G_0(k))$ and $G_0(G_1(G_0(k)))$

$G_0(G_1(G_0(k)))$ ➔ Can evaluate $F_k(001)$



GGM Puncturable PRF

- $\boldsymbol{F_K}: \{\mathbf{0},\mathbf{1}\}^{n} \rightarrow \{\mathbf{0},\mathbf{1}\}^{\lambda}$
- ***GGM tree has depth-proportional to length of PRF input n***
- ***Punctured Key Stores: n pseudorandom strings of length λ***
- ***Punctured Key K{x} has size O(n λ***)
- Security follows similar hybrid argument

Digital Signatures from iO

- Define a circuit $C_K(\sigma, m) = \begin{cases} 1 & \text{if } \mathrm{Eval}(\mathrm{K}, \mathrm{m}) = \sigma \\ 0 & \text{otherwise} \end{cases}$

- **Key Generation:** Pick a random Puncturable PRF key K and output public key $\mathrm{pk} = \mathrm{iO}(C_K)$ and secret key $sk = \mathrm{K}$

- **Signing:** $\mathrm{Sign}(\mathrm{sk}, \mathrm{m}) = \mathrm{Eval}(\mathrm{K}, \mathrm{m}) = \sigma$
- **Signature Verification:** Just run obfuscated program $\mathrm{pk}(.\,,.\,)$ with inputs σ and m.

Digital Signatures from iO

- Define a circuit $C_K(\sigma, m) = \begin{cases} 1 & \text{if Eval(K, m)} = \sigma \\ 0 & \text{otherwise} \end{cases}$

- **Selective Signature Forgery Game:** Fix a target message m^* and then generate (sk,pk).
  - Attacker may make q queries to signing oracle $\text{Sign}(\text{sk},\cdot)$ on any other message i.e., $m_i \neq m^*$ for all queries i.
  - **Attacker's goal:** output forgery $\sigma^*$for m^*
- Selective Security ➔ Adaptive Security (Union bound over all target messages m^*)
  - Union bound trick requires sub-exponential security of iO + PPRF
  - Standard Trick in many iO security proofs

Selective Security Proof: Hybrid Argument

$$C_K(\sigma, m) = \begin{cases} 1 & \text{if } \mathrm{Eval}(\mathrm{K}, \mathrm{m}) = \sigma \\ 0 & \text{otherwise} \end{cases}$$

- **Hybrid 1:** Replace $\mathrm{pk} = iO(C_K)$ *with* $\mathrm{pk} = iO(C_{f,K})$ *where*

  Define a circuit $C_{f,K}(\sigma, m) = \begin{cases} 1 & \text{if } f(\mathrm{Eval}(\mathrm{K}, \mathrm{m})) = f(\sigma) \\ 0 & \text{otherwise} \end{cases}$

Note 1: If f is a one-way permutation then $C_{f,K}$ and C_K are equivalent

Note 2: Hybrid 1 is indistinguishable from Hybrid 0 (original game) due to iO security (since $C_{f,K}$ and C_K are equivalent circuits)

Selective Security Proof: Hybrid Argument

$$C_{f,K}(\sigma, m) = \begin{cases} 1 & \text{if } f(\mathrm{Eval}(\mathrm{K}, \mathrm{m})) = f(\sigma) \\ 0 & \text{otherwise} \end{cases}$$

- Equivalent if f is a one-way permutation
- **Hybrid 2:** Replace $\mathrm{pk} = iO(C_{f,K})$ *with* $\mathrm{pk} = iO(C_{K\{m^*\},z^*})$ *where*

$$C_{K\{m^*\},z^*}(\sigma, m) = \begin{cases} 1 & \text{if } (f(\sigma), m) = (z^*, m^*) \\ 1 & \text{if } \mathrm{Eval}(K\{m^*\}, \mathrm{m}) = \sigma \\ 0 & \text{otherwise} \end{cases} \quad where\, \mathrm{z}^* = f(\sigma^*)$$

Note: Hybrid 2 is indistinguishable from Hybrid 1 due to iO security (since $C_{K\{m^*\},z^*}$ and $C_{f,K}$ are equivalent circuits)

Selective Security Proof: Hybrid Argument

- $C_{K\{m^*\},z^*}(\sigma, m) = \begin{cases} 1 & \text{if } (f(\sigma), m) = (z^*, m^*) \\ 1 & \text{if } \mathrm{Eval}(K\{m^*\}, \mathrm{m}) = \sigma \\ 0 & \text{otherwise} \end{cases}$ where $z^* = f(\sigma^*)$

- **Hybrid 3:** Replace $\mathrm{pk} = iO(C_{K\{m^*\},z^*})$ *with* $\mathrm{pk} = iO(C_{K\{m^*\},f(r)})$ *where r is a uniformly random string and*

$$C_{K\{m^*\},R}(\sigma, m) = \begin{cases} 1 & \text{if } (f(\sigma), m) = (R, m^*) \\ 1 & \text{if } \mathrm{Eval}(K\{m^*\}, \mathrm{m}) = \sigma \\ 0 & \text{otherwise} \end{cases} \quad \textit{where } R = f(r)$$

Note: Hybrid 3 is indistinguishable from Hybrid 1 due to security of the punctured PRF. Even if we reveal r attacker cannot distinguish random r from $\sigma^* = \mathrm{Eval}(K, m^*)$.

Selective Security Proof: Hybrid Argument

- **Hybrid 2:** Replace $\text{pk} = iO(C_{K\{m^*\},\sigma^*})$ *with* $\text{pk} = iO(C_{K\{m^*\},f(r)})$ *where r is a uniformly random string and*

$$C_{K\{m^*\},R}(\sigma, m) = \begin{cases} 1 & \text{if } (f(\sigma), m) = (R, m^*) \\ 1 & \text{if Eval}(K\{m^*\}, \text{m}) = \sigma \\ 0 & \text{otherwise} \end{cases} \quad \textit{where } R = f(r)$$

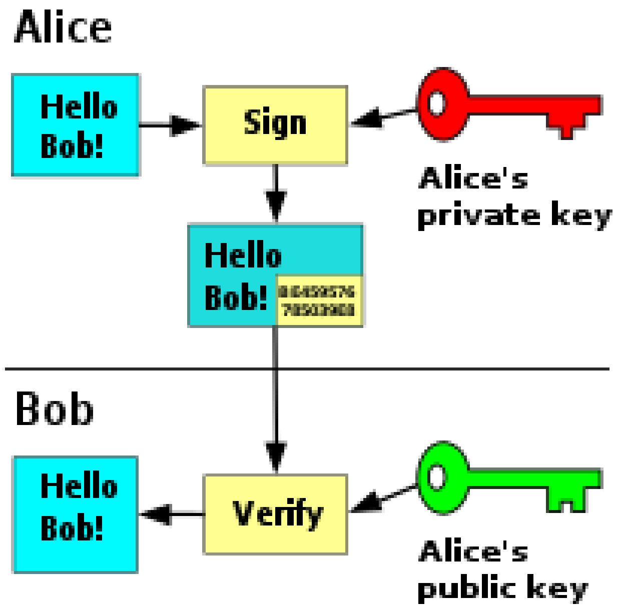
Signature Forgery in Hybrid 3? *Forging a signatures requires us to find some* σ^* such that $f(r) = f(\sigma^*)$. Difficulty follows from security of OWP. Obfuscated program only contains $R = f(r)$ *for uniformly random r.*

Short Signatures from iO

- The signature is just a PRF output ➔ we can hope for λ-bit signatures with λ-bit security
- Advantage of the iO based signature construction ☺
- Length of Public Key will be much longer longer (obfuscated program) ☺
- Not practically efficient (unless we get practically efficient iO) ☹

Short Signature Schemes

- **RSA-FDH:** $\omega(k)$-bits
- **EC-DSA:** 4k-bits
- **Schnorr:** 4k-bits
- **Short Schnorr Signature:** 3k-bits
  - Suggested in Schnorr's original paper
  - **Eurocrypt 2022:** provides k-bit security in idealized models (GGM+Random Oracle)
- **BLS:** 2k-bits
  - Bilinear Pairings for Verification
  - Shorter signatures, but higher computational overhead
- **iO Based Signatures:** k bits
  - Purely Theoretical Construction
  - No practical instantiation of indistinguishability obfuscation!



Witness Encryption

Recap: NP-Complete problems.

- Consider an NP-Complete problem e.g., CIRCUIT-SAT
- **Instance:** Circuit $C: \{0,1\}^n \rightarrow \{0,1\}$
- **Decision Problem:** Does there exist some input x such that $C(x) = 1$?
- **NP Certifier:** Given witness x it is easy to verify that the circuit is satisfiable e.g., $C(x) = 1$
- **NP Hard:** Polynomial time reduction from *any* other decision problem in NP (e.g., SAT, 3COLOR, CLIQUE) reduces to CIRCUIT-SAT.

Witness Encryption

- **Idea:** Use C as a public key and witness x as a secret key

$$\boldsymbol{Enc(C, m) = c}$$
$$\boldsymbol{Dec(x, c) = m}\ if\ C(x) = 0$$

- Any party can encrypt message using C.
- Ciphertext can only be decrypted if we know a witness x.
- If no witness exists then ciphertext is “permanently locked” i.e., attacker cannot distinguish between $Enc(C, m)$ and $Enc(C, m')$

Witness Encryption

- **Idea:** Use C as a public key and witness x as a secret key

$$\boldsymbol{Enc(C, m) = c}$$

$$\mathbf{Dec(x, c) = m} \text{ if } \mathrm{C(x) = 1}; \text{otherwise } \mathbf{Dec(x, c)} = \perp$$

Construction:

$$\mathrm{Enc}(C, m) = \mathrm{iO}\left(1^{\lambda}, \boldsymbol{D_{C,m}}\right)$$

Where $\boldsymbol{D_{C,m}}$ is a circuit such that

$$\boldsymbol{D_{C,m}(x)} = \begin{cases} \boldsymbol{m} & \boldsymbol{if\ C(x) = 0} \\ \perp & \boldsymbol{otherwise} \end{cases}$$

Witness Encryption

- **Idea:** Use C as a public key and witness x as a secret key

$$\boldsymbol{Enc(C, m) = c}$$

$$\mathbf{Dec}(\mathbf{x}, \mathbf{c}) = \mathbf{m} \text{ if } \mathrm{C}(\mathrm{x}) = 1; \text{otherwise } \mathbf{Dec}(\mathbf{x}, \mathbf{c}) = \perp$$

Construction:

$$\mathrm{Enc}(C, m) = \mathrm{iO}\left(1^{\lambda}, \boldsymbol{D_{C,m}}\right)$$

$$\boldsymbol{D_{C,m}(x)} = \begin{cases} \boldsymbol{m} & \boldsymbol{if\ C(x) = 0} \\ \perp & \boldsymbol{otherwise} \end{cases}$$

$$\mathrm{Dec}(x, c) = \mathrm{c}(x)$$

Witness Encryption

- **Idea:** Use C as a public key and witness x as a secret key

$$\boldsymbol{Enc(C, m) = c}$$

$$\mathbf{Dec(x, c) = m} \text{ if } \mathrm{C(x)} = 1; \text{otherwise } \mathbf{Dec(x, c)} = \perp$$

Security Analsis: If $\mathrm{C(x)} = 0$ for all inputs x then $\boldsymbol{D_{C,m}}$ is equivalent to the trivial circuit $\boldsymbol{D(x)} :=\perp$**.**

iO Security

➔$\mathrm{Enc}(C, m) = \mathrm{iO}\left(1^\lambda, \boldsymbol{D_{C,m}}\right)$ *cannot be distinguished from* $\mathrm{iO}\left(1^\lambda, \boldsymbol{D}\right)$

➔ $\mathrm{Enc}(C, m') = \mathrm{iO}\left(1^\lambda, \boldsymbol{D_{C,m'}}\right)$ *cannot be distinguished from* $\mathrm{iO}\left(1^\lambda, \boldsymbol{D_{C,m}}\right)$

$$\boldsymbol{D_{C,m}(x)} = \begin{cases} \boldsymbol{m} & \boldsymbol{if\ C(x) = 0} \\ \perp & \boldsymbol{otherwise} \end{cases}$$

Functional Encryption

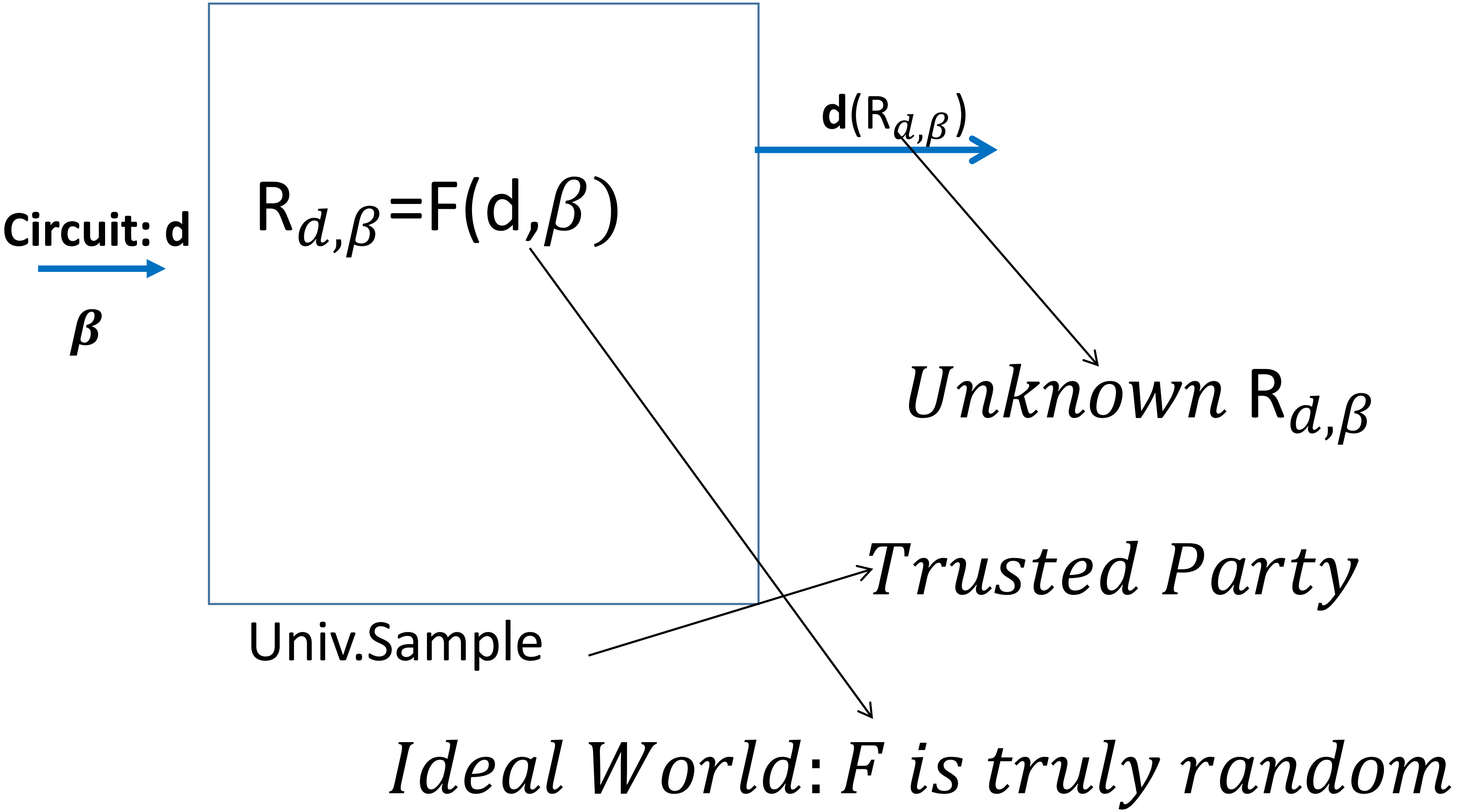
- **Public Key Encryption:** $c \coloneqq \mathrm{Enc}_{\mathrm{pk}}(m)$
- **Secret Key Used to Decrypt:** $\mathrm{Dec}_{\mathrm{sk}}(c) = m$
- Can generate special Secret Key for Circuit C: sk_C
  - **Correctness:** $\mathrm{Dec}_{\mathrm{sk}_\mathrm{C}}(c) = C\big(\mathrm{Dec}_{\mathrm{sk}}(c)\big) = C(m)$
  - **Security Goal (Intuition):** Cannot learn “more” than $C(m)$

Construction Idea (Over Simplified): $sk_C = \mathrm{iO}\big(1^\lambda, \boldsymbol{D_C}\big)$

$$\boldsymbol{D_C}\big(\mathrm{Enc}_{\mathrm{pk}}(m)\ \big) = \boldsymbol{C}(\mathrm{Enc}_{\mathrm{pk}}(m))$$

Full Construction/Proof: Uses Statistically Simulation Sound Non-Interactive Zero Knowledge Proofs.

Application: Universal Sampler [Hofheinz et al. 2016]



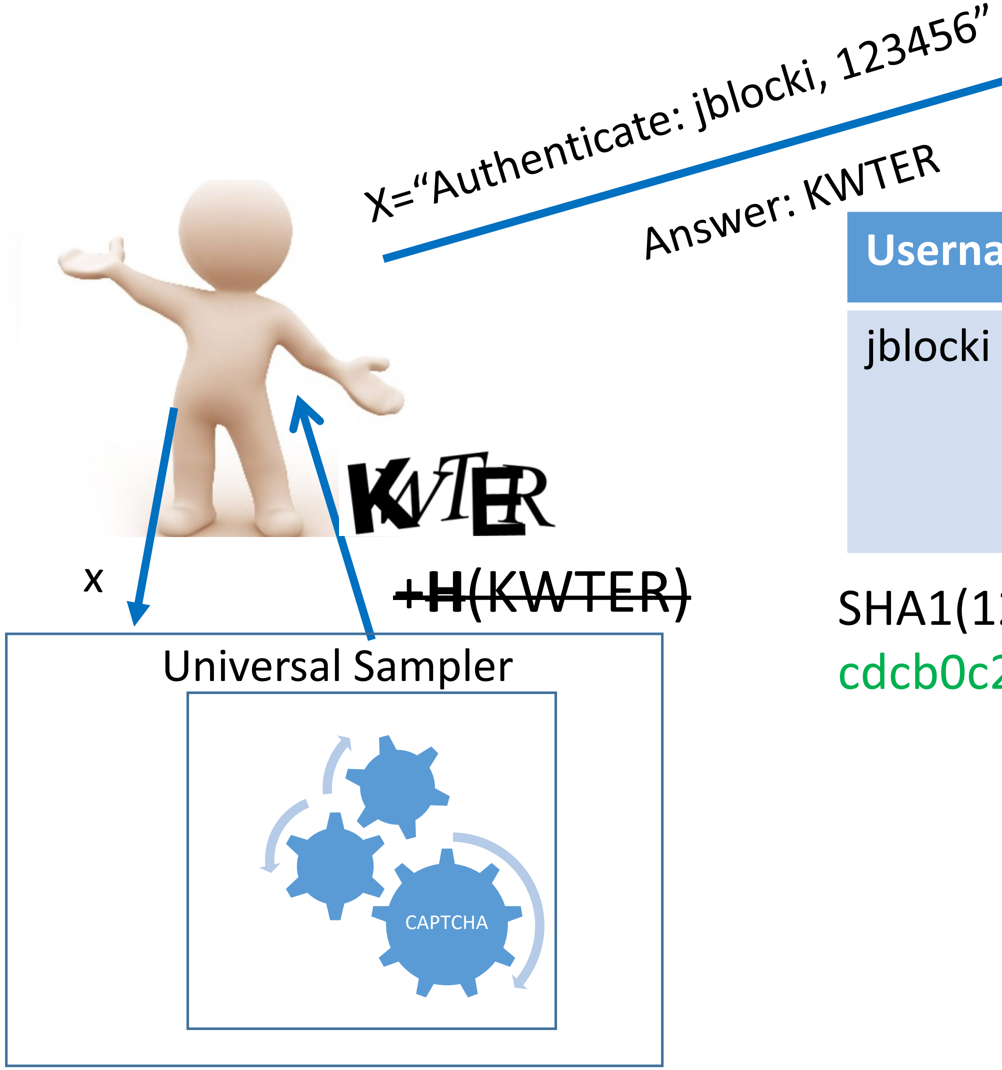
Application: Universal Sampler

- Setup
  - **Input:** 1^{λ} (e.g., size of crypo keys) and
  - **Output:** U (e.g., an obfuscated program)
- Sample
  - **Input**: U, d, β
    - d a polynomial size circuit
    - β randomness index
  - **Output**: $d(r_{\beta})$
    - Ideal World: Secret random string chosen once and for all for each given β

Universal Sampler [Hofheinz et al. 2016]

- Construction in Random Oracle Model
- Crypto Assumptions: iO + OWF
  - Random Oracle not queried inside iO
- Adaptive Security
  - “delayed backdoor programming” via Random Oracle

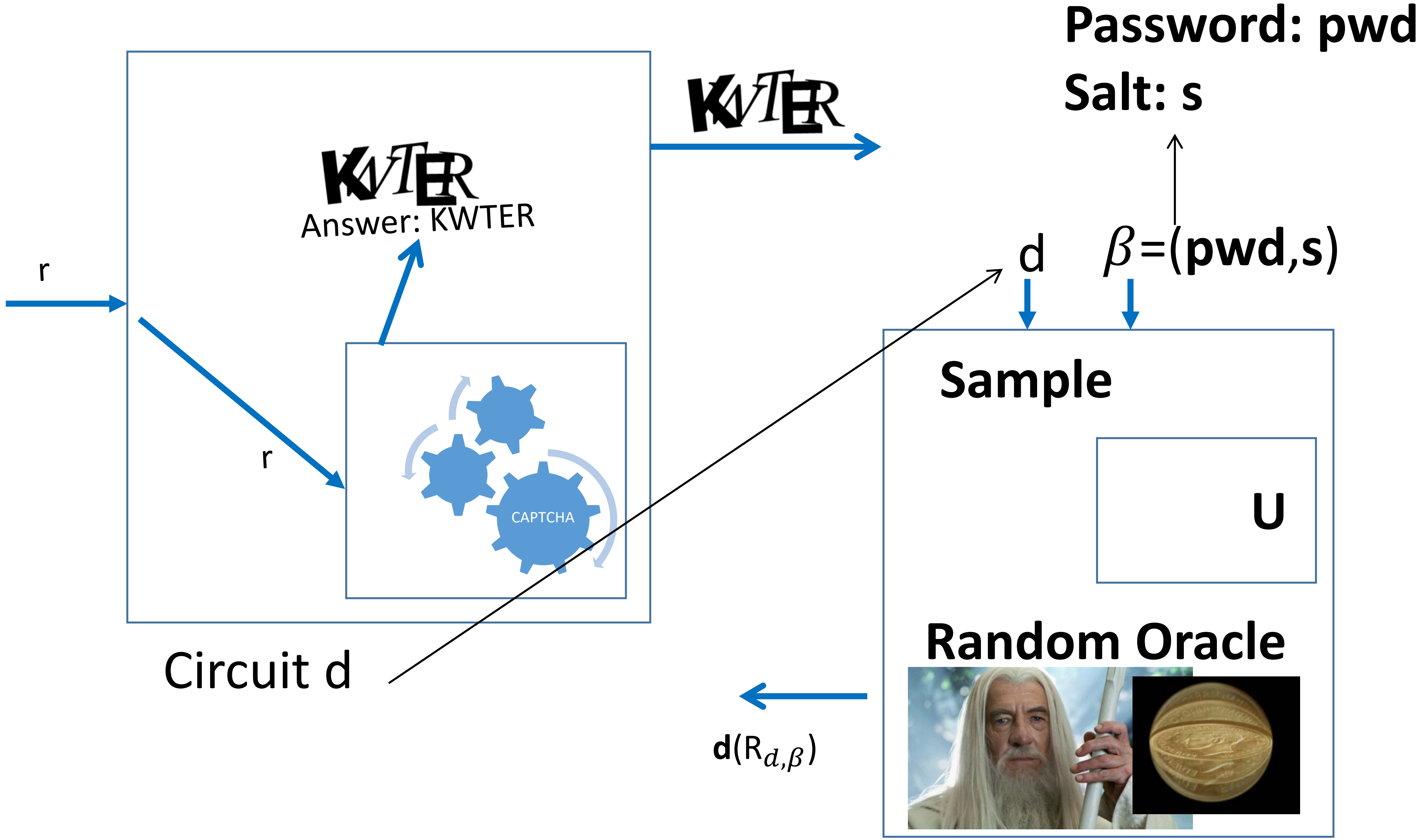
Application: CAPTCHAs in Password Storage



Username	Salt	Hash
jblocki	89d978034a3f6	1f88ecdcb0c25e8ae1ed1c9ce6f2e2e6dcfb0e21

SHA1(123456KWTER89d978034a3f6)=1f88ecdcb0c25e8ae1ed1c9ce6f2e2e6dcfb0e21

PoH Construction



Security Reduction

Main Theorem: Blackbox reduction transforms any **ppt** algorithm breaking PoH security into a **ppt** algorithm breaking CAPTCHA security.

(Assuming security of Universal Sampler)

Statement about human ignorance

Security Analysis

Thm (Informal): If UNI is adaptively secure universal sampler and CAPT is computer uncrackable CAPTCHA then password authentication scheme is *costly to crack.*

Costly to Crack: An adversary with *m human work units* can crack users password with probability at most

$$\lambda_m = \sum_{i=1}^{m} p_i + negligible$$

Security Analysis

Thm (Informal): d CAPT
is computer unc n
scheme is *costly*

Standard CAPTCHA assumption:
Adversary not given hashes answers to puzzles.

Costly to Crack: *An adversary with m `human work units' can crack users password with probability at most*

$$\lambda_m = \sum_{i=1}^{m} p_i + negligible$$

Security Analysis

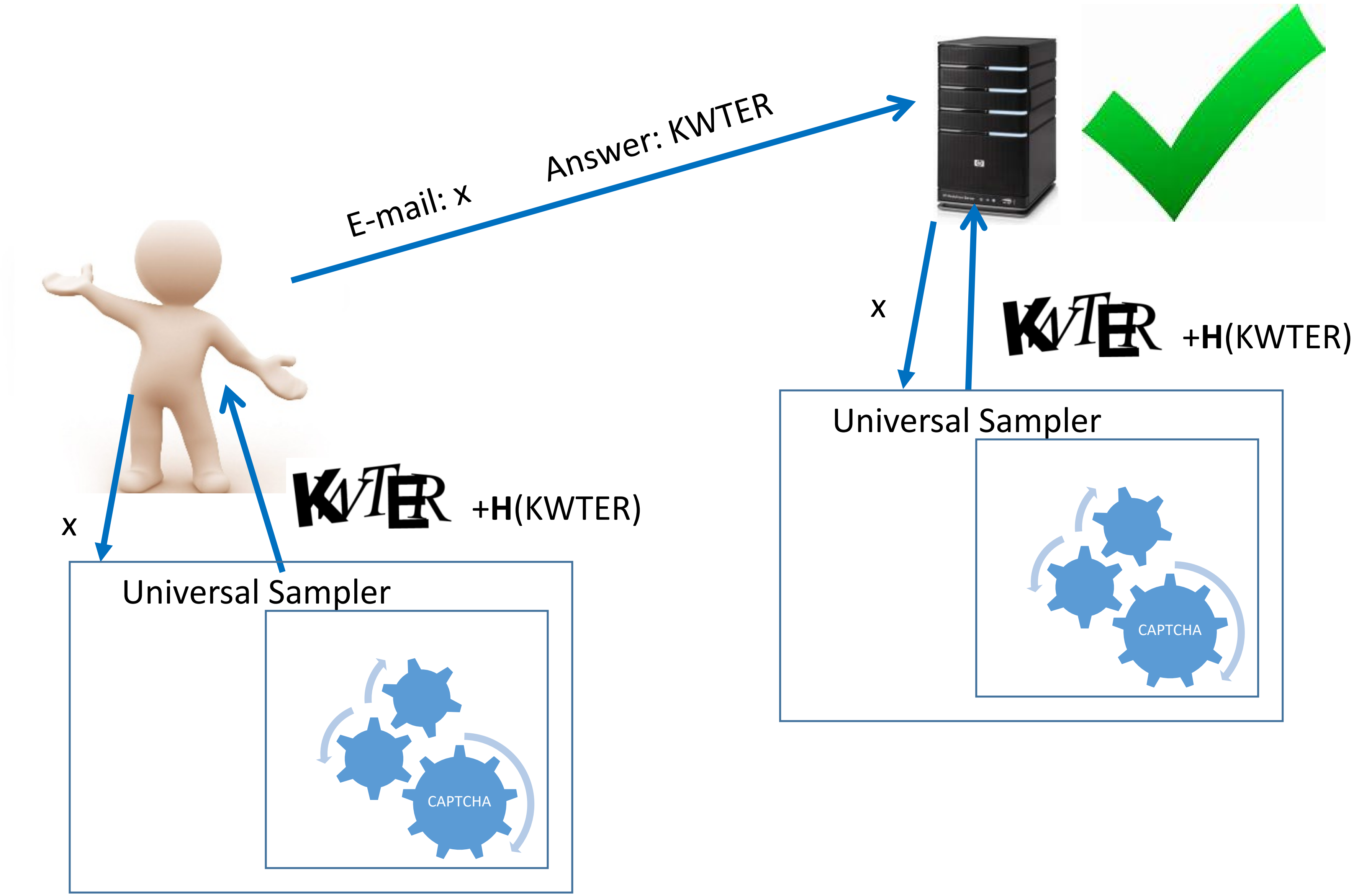
Thm (Informal): d CAPT
is computer unc n
scheme is *costly*

Standard CAPTCHA assumption: Adversary not given hashes answers to puzzles.

Costly to Crack*: An adversary with m `human work units' can crack*
users passwor

** Actually show blackbox reduction from ppt adversary breaking security of password scheme to ppt adversary breaking CAPTCHA security

PoH for E-mails



Thanks for Listening