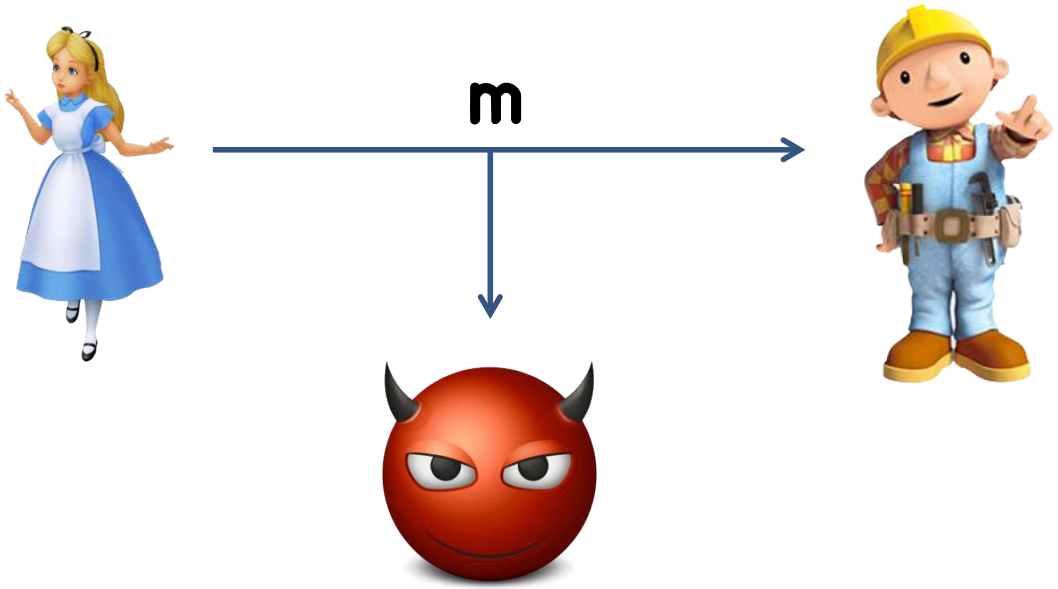
Cryptography in the Age of Quantum Computers

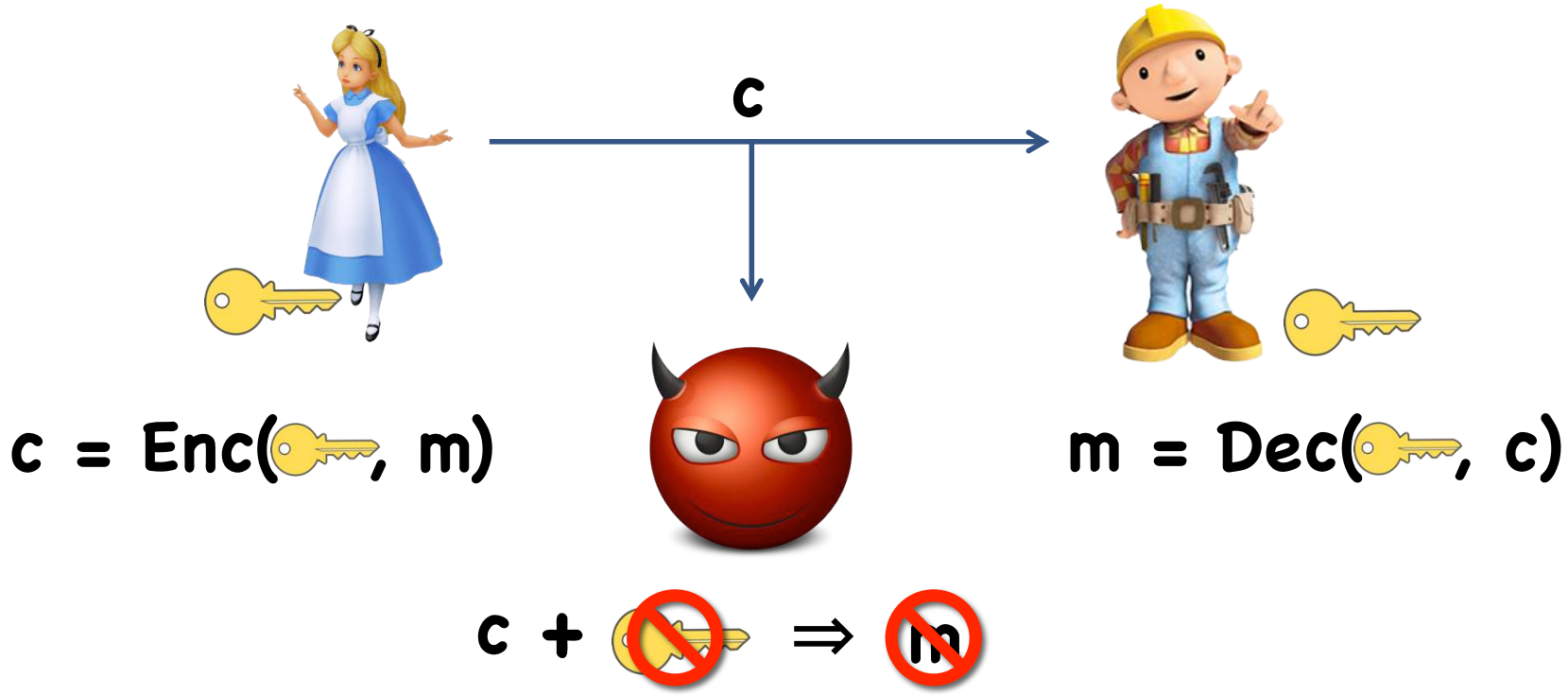
Mark Zhandry – Stanford University

Based on joint works with:
Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner

Typical Crypto Application



Solution: (Private Key) Encryption



Major question: How is security defined?

Definition 1: 1-time security

For any m_0, m_1:

$$c_0 = \mathrm{Enc}(k, m_0) \approx c_1 = \mathrm{Enc}(k, m_1)$$

Statistical security: statistical closeness

- [Sha'49]: $|k| \geq |m|$

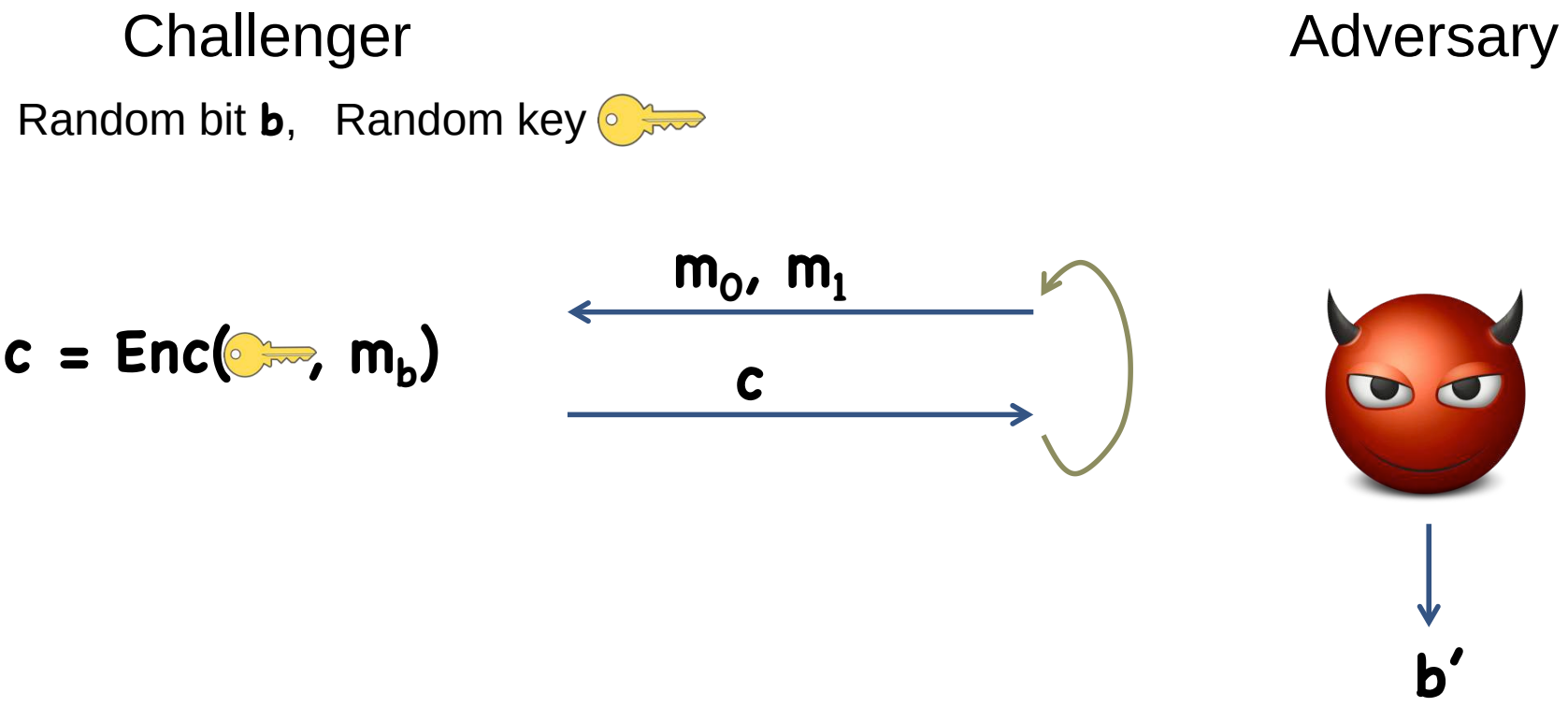
Computational security: computational indistinguishability

- Restrict adversaries running efficiently
- Now possible to have $|k| << |m|$

Question: what if I encrypt a second message?

Definition 2: CPA Security

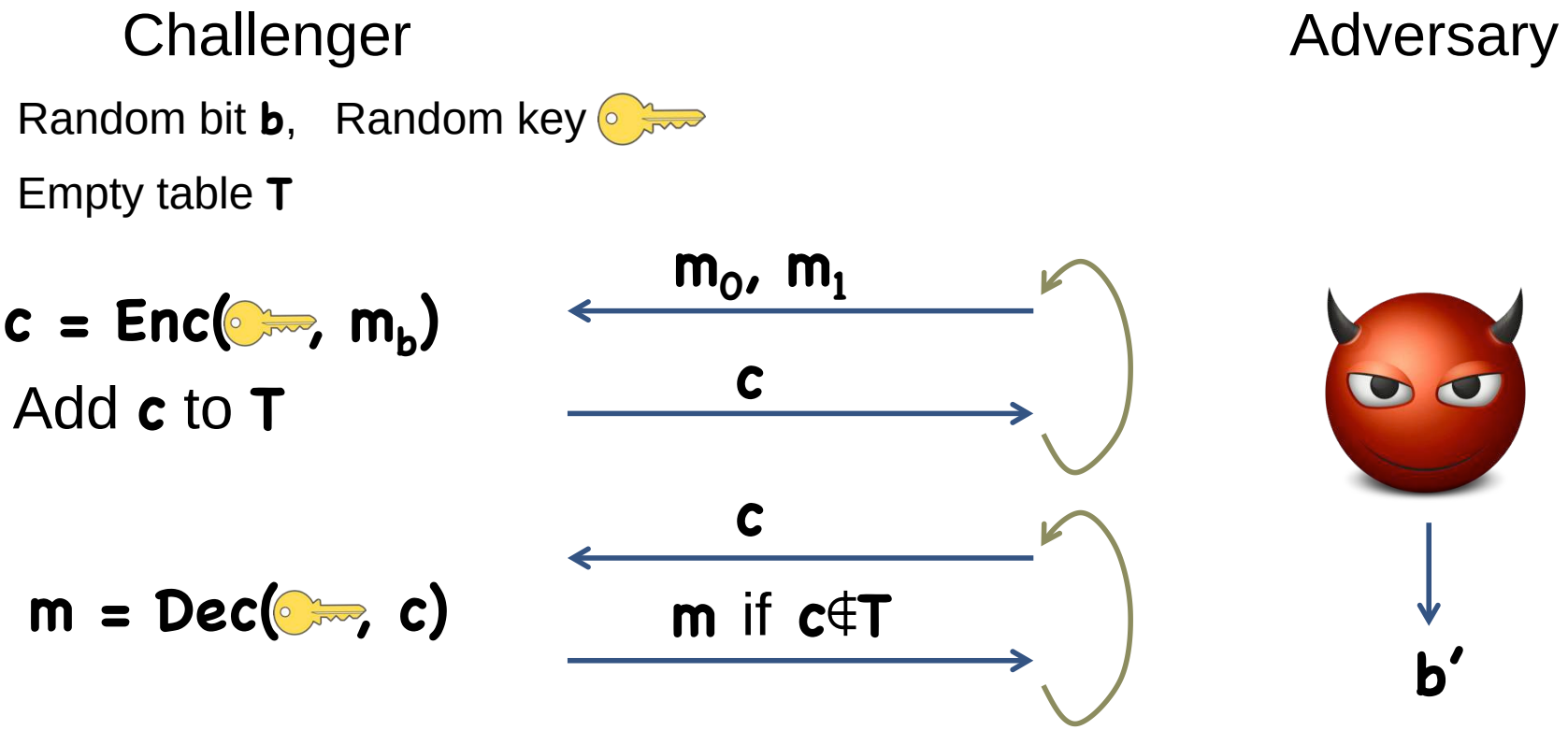
Indistinguishability under *chosen plaintext attack*



Def: CPA-Security ⇒ ∀ efficient , | Pr[b′=b] − ½ | < negl

Definition 3: CCA Security

Indistinguishability under *chosen ciphertext attack*



Def: CCA-Security ⇒ ∀ efficient , | Pr[b'=b] − ½ | < negl

Other Scenarios

Circular security:

Side-channel attacks:

Takeaway:
Models should give adversary as much power as possible

Quantum Computers

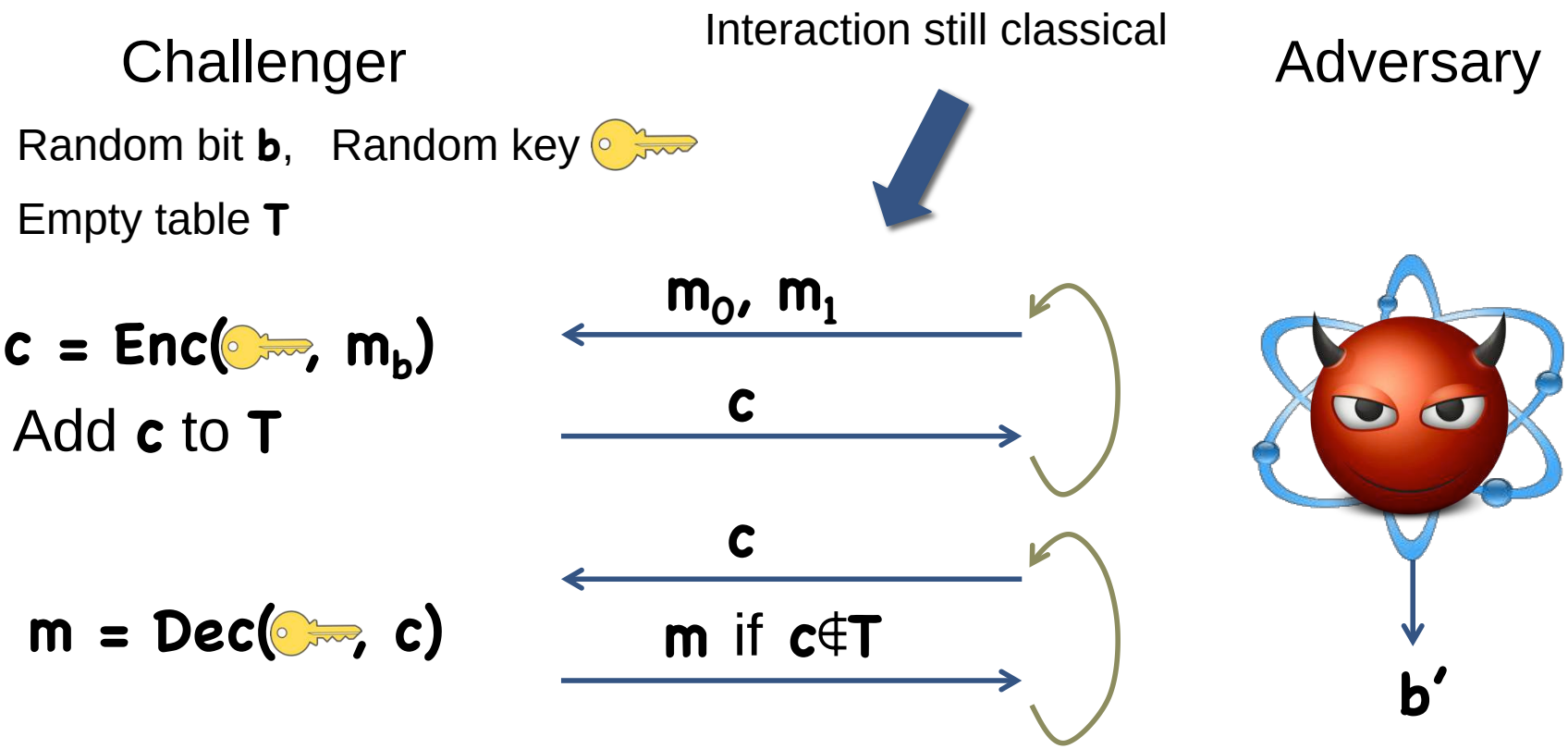
So far, assumed adversary obeys classical physics

What about quantum physics?

Quantum computing = using quantum physics to perform certain computations

- Active research area
- [Sho’94]: quantum computers can break lots of crypto

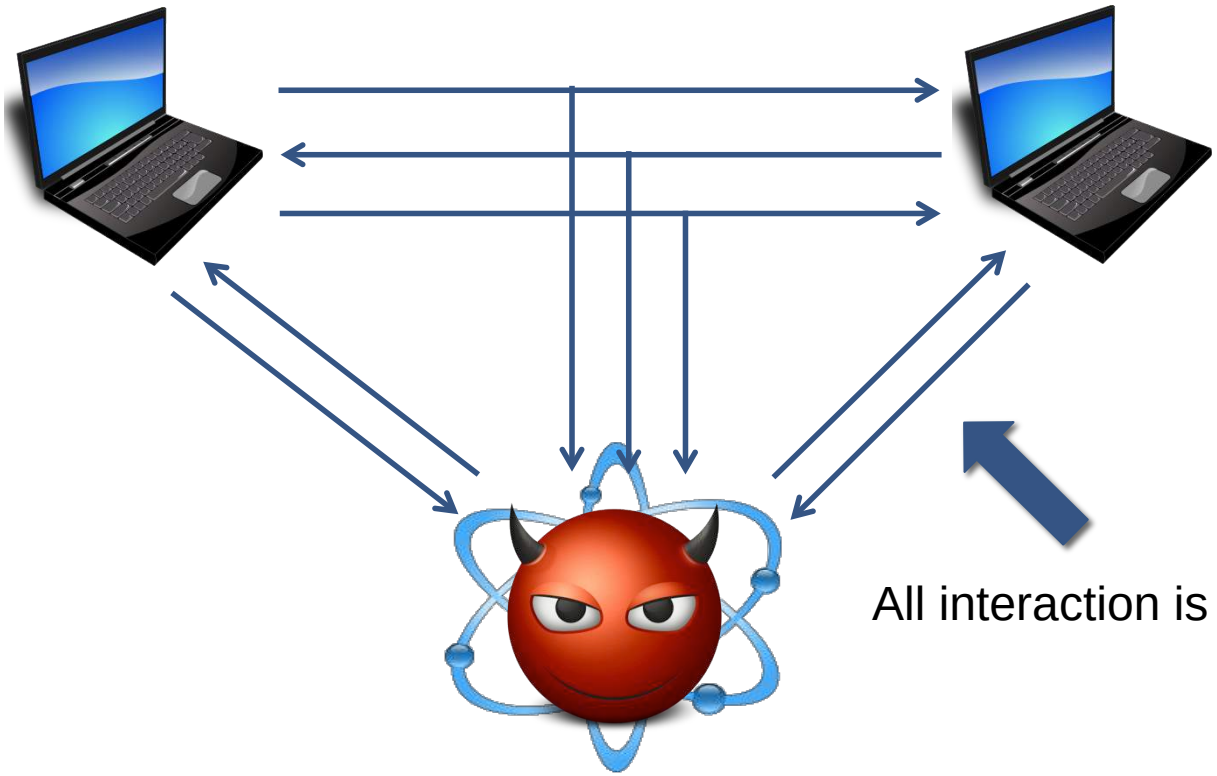
Post-Quantum CCA Security



Def: CCA-Security ⇒ ∀ efficient | Pr[b'=b] − ½ | < negl

Post-Quantum Security

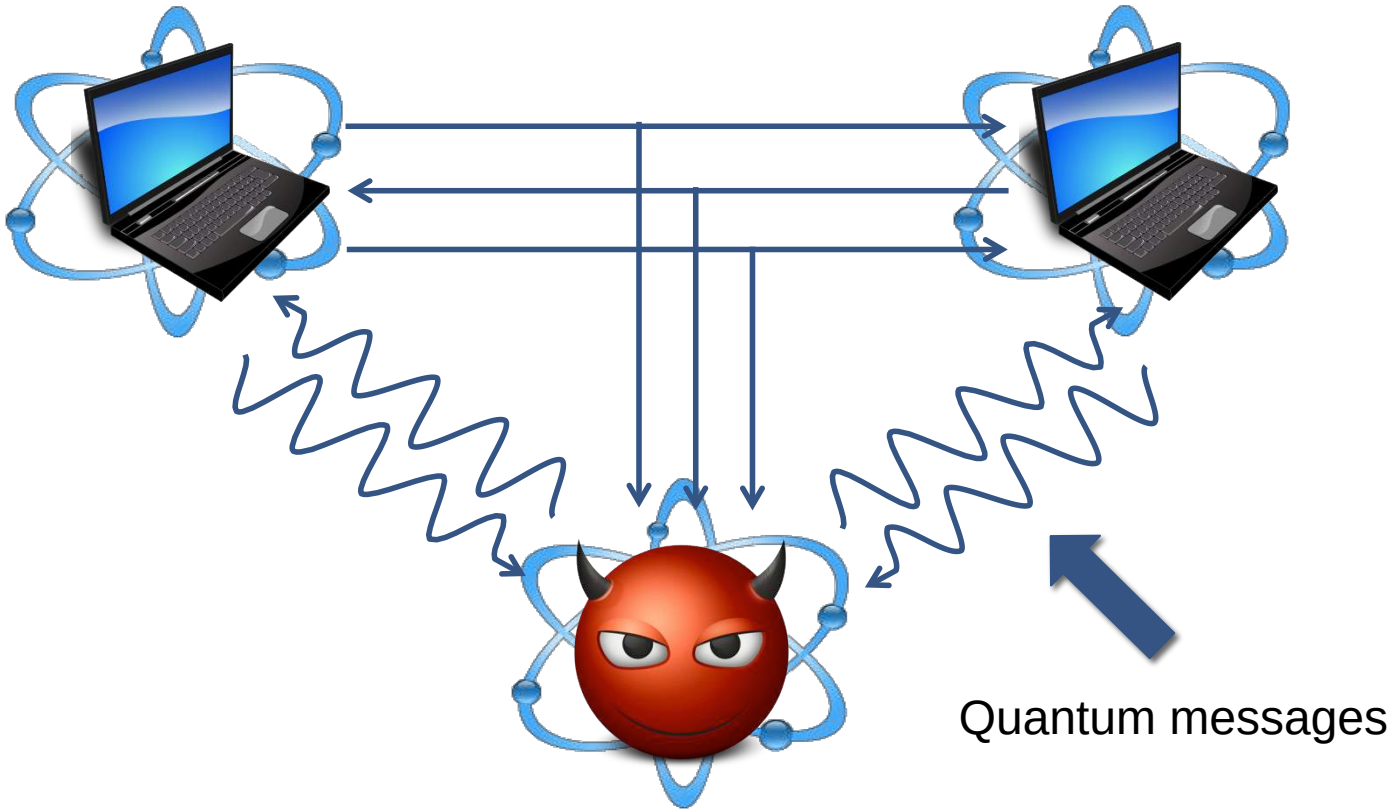
Post-quantum = end-users are classical



All interaction is classical

Full Quantum Security

Full quantum = end-users are *quantum*



Quantum Background

Quantum states:

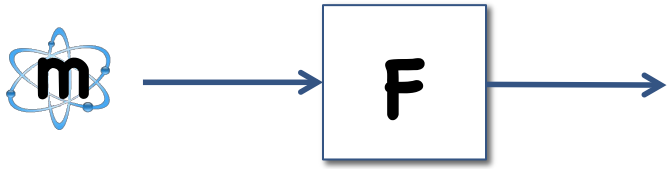
$= \text{superposition of } \mathbf{all} \text{ messages}$

$= \Sigma\, \alpha_m |m\rangle \qquad (\Sigma\, |\alpha_m|^2 = 1)$

Measurement:

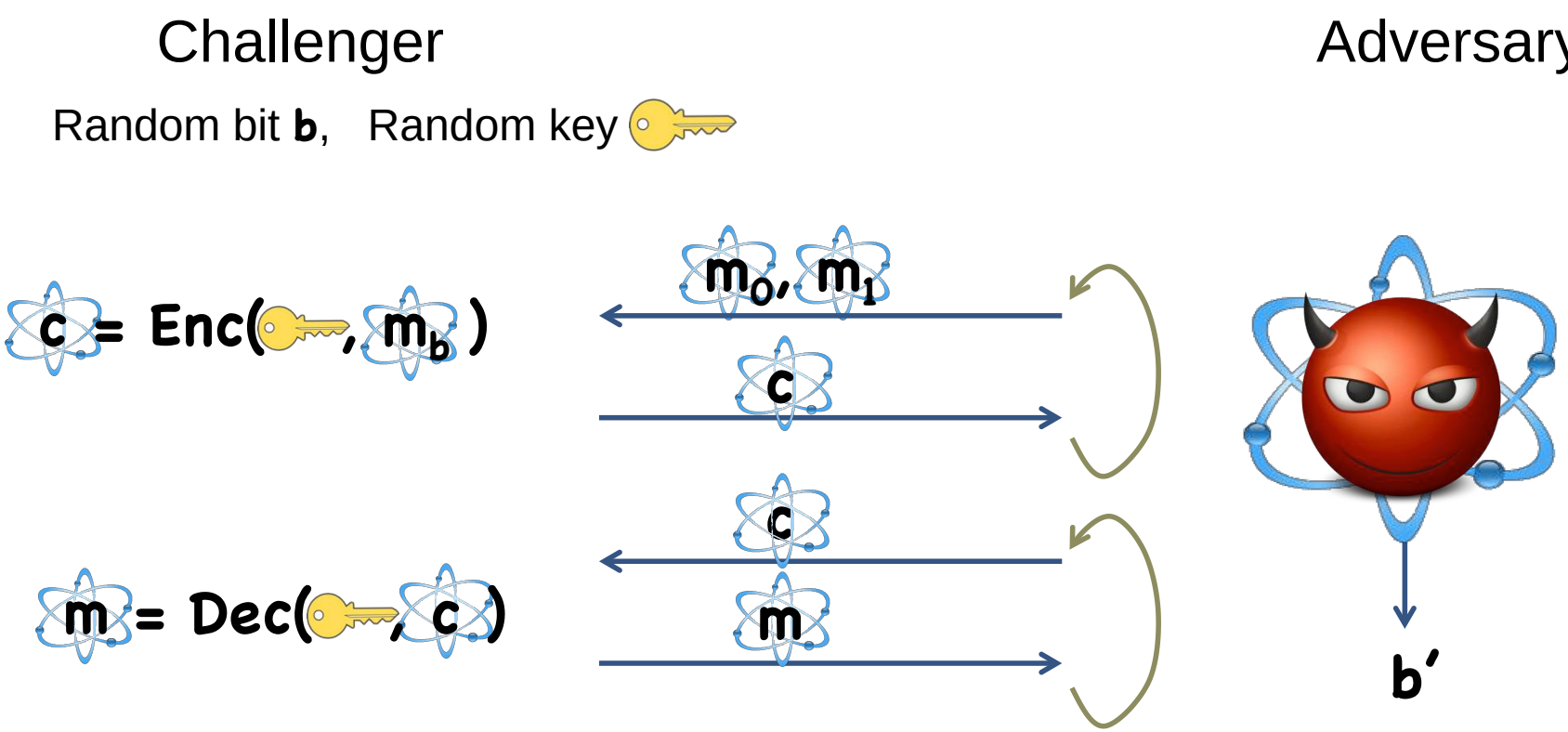
m with probability $|\alpha_m|^2$

Simulate classical ops in superposition:



$= \Sigma\, \alpha_m |F(m)\rangle$

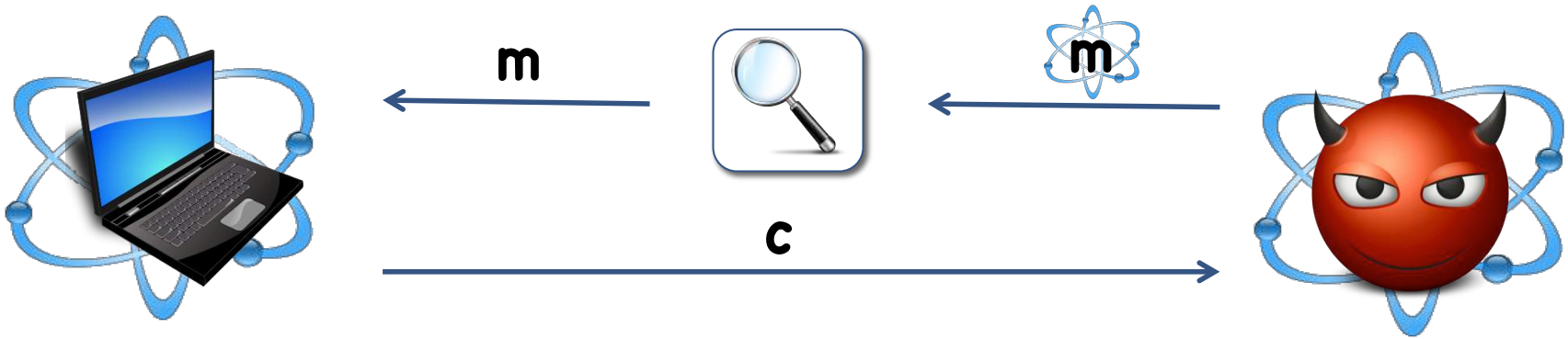
Full Quantum CCA Security?



Def: CCA-Security ⇒ ∀ efficient | Pr[b'=b] − ½ | < negl

Are Full Quantum Attacks Plausible?

Objection: can always “classicalize” by sampling



⇒ Reduce attack to post-quantum attack!

Reasons to still use full quantum notions:

- Classicalization is burden on hardware designer
- What if adversary can bypass?
- Classicalization amounts to a *hardware* assumption

This Work

[BDFLSZ'11,Zha'12a,Zha'13]: Quantum random oracle model

[Zha'12b]: Pseudorandom functions

[BZ'13a]: Message Authentication Codes

[BZ'13b]: Digital signatures and encryption

Theorem: Full-quantum security > Post-quantum security

Theorem (Informal): Full-quantum security can be obtained with "minimal" overhead w.r.t. post-quantum security

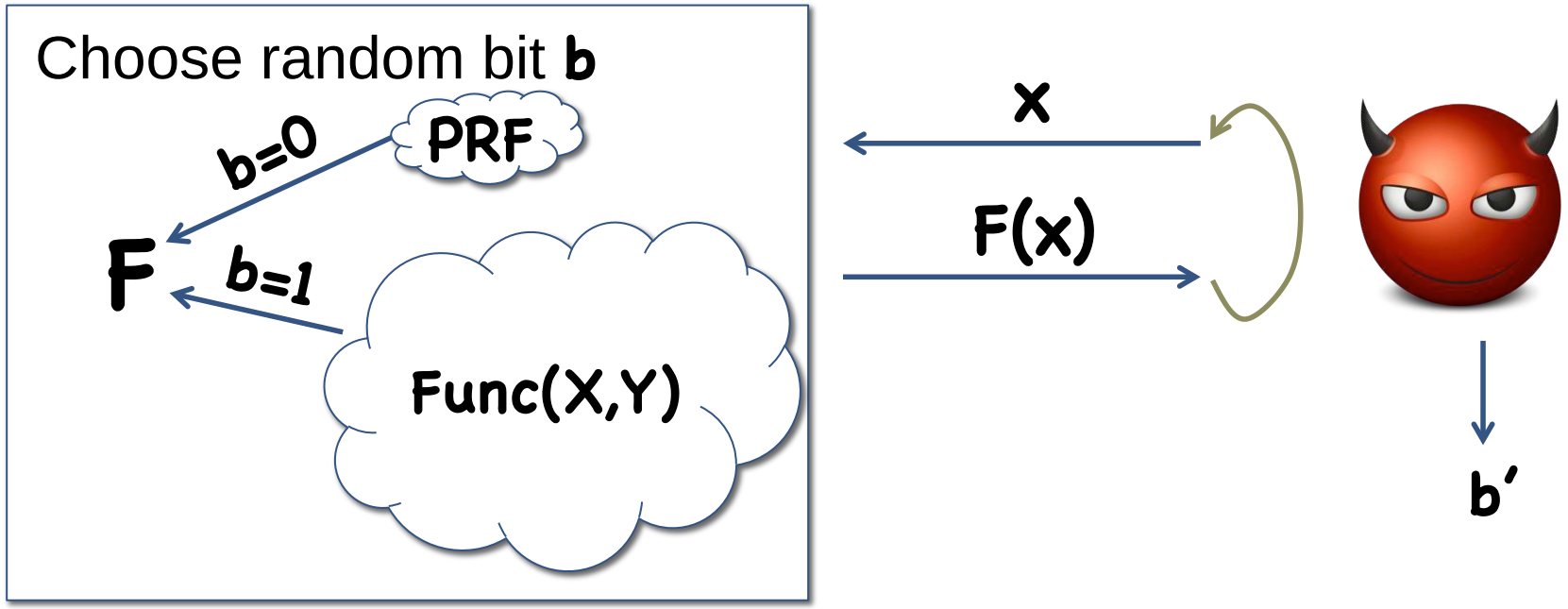
Example: Pseudorandom Functions

[GGM'84]

Efficient keyed functions that "look like" random functions

- Fundamental building block in symmetric crypto

Classical security:



Def: Security ⇒ ∀ efficient  , | Pr[b'=b] − ½ | < negl

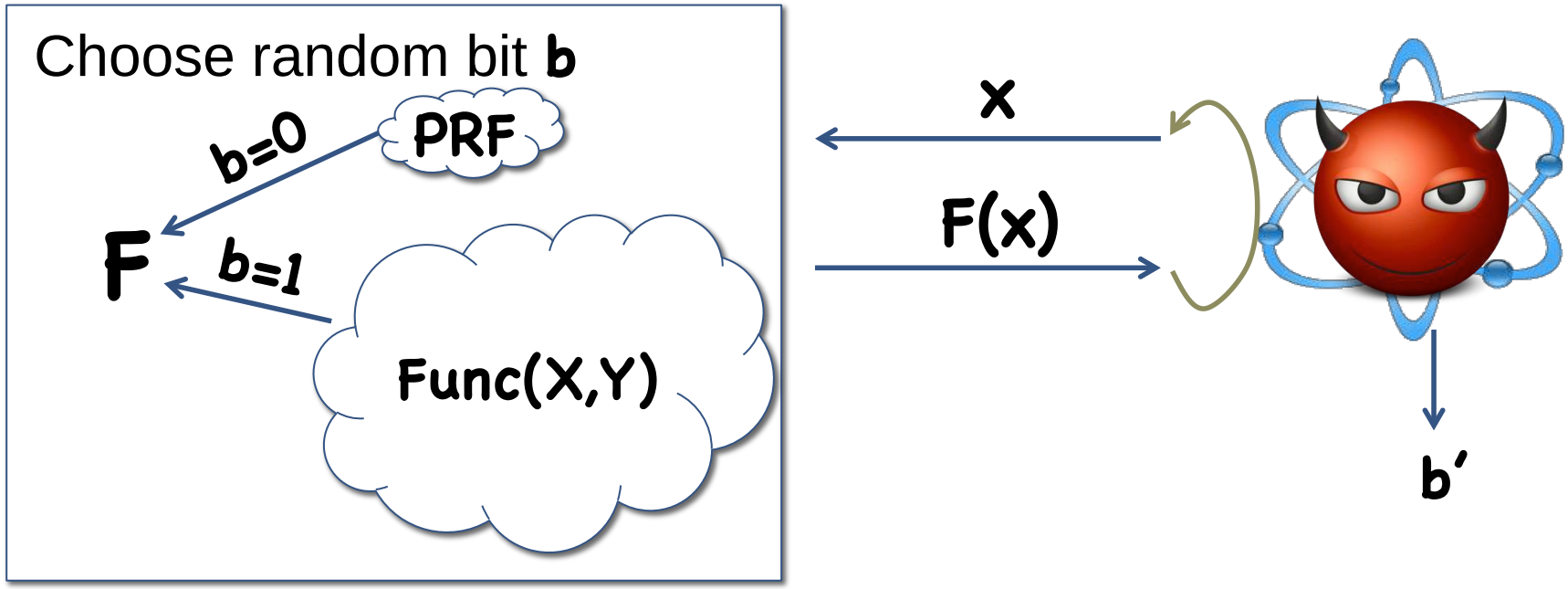
Example: Pseudorandom Functions

[GGM'84]

Efficient keyed functions that "look like" random functions

- Fundamental building block in symmetric crypto

Post-quantum security:



Def: PQ-Security ⇒ ∀ efficient , | Pr[b'=b] − ½ | < negl

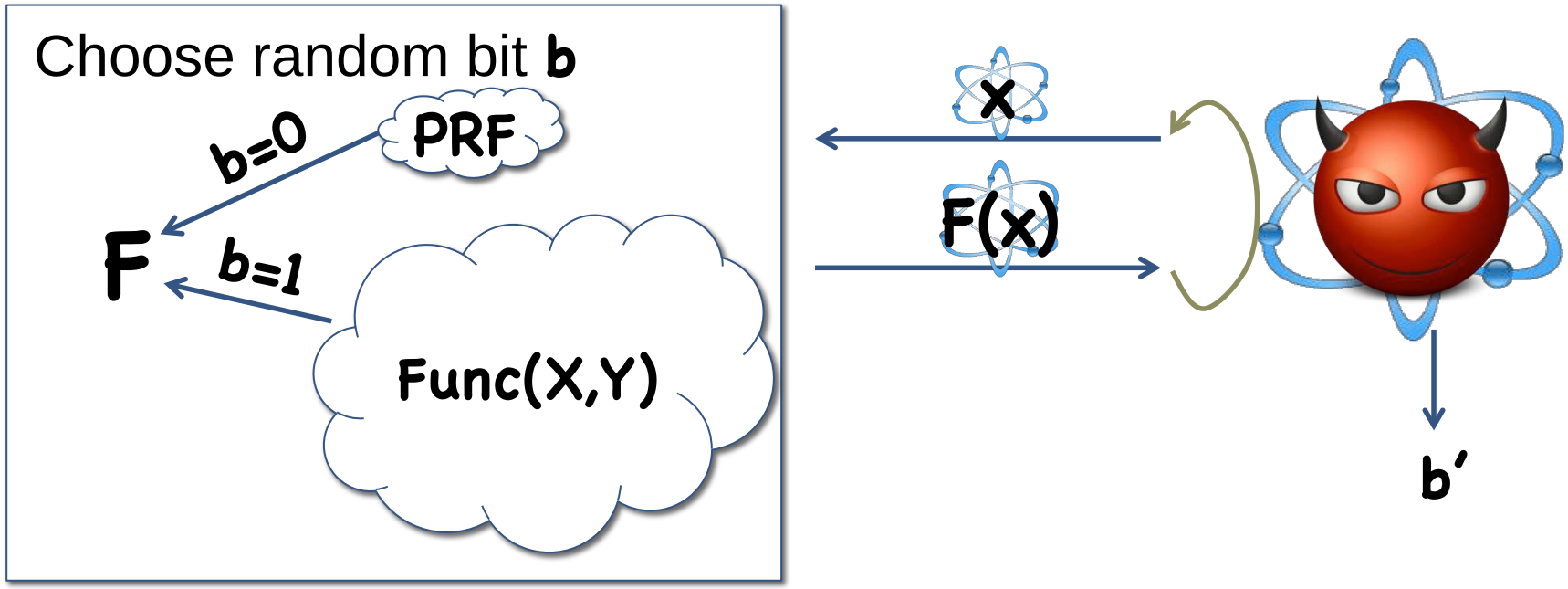
Example: Pseudorandom Functions

[GGM'84]

Efficient keyed functions that "look like" random functions

- Fundamental building block in symmetric crypto

Full-quantum security:



Def: FQ-Security ⇒ ∀ efficient , | Pr[b'=b] − ½ | < negl

How to build QPRFs?

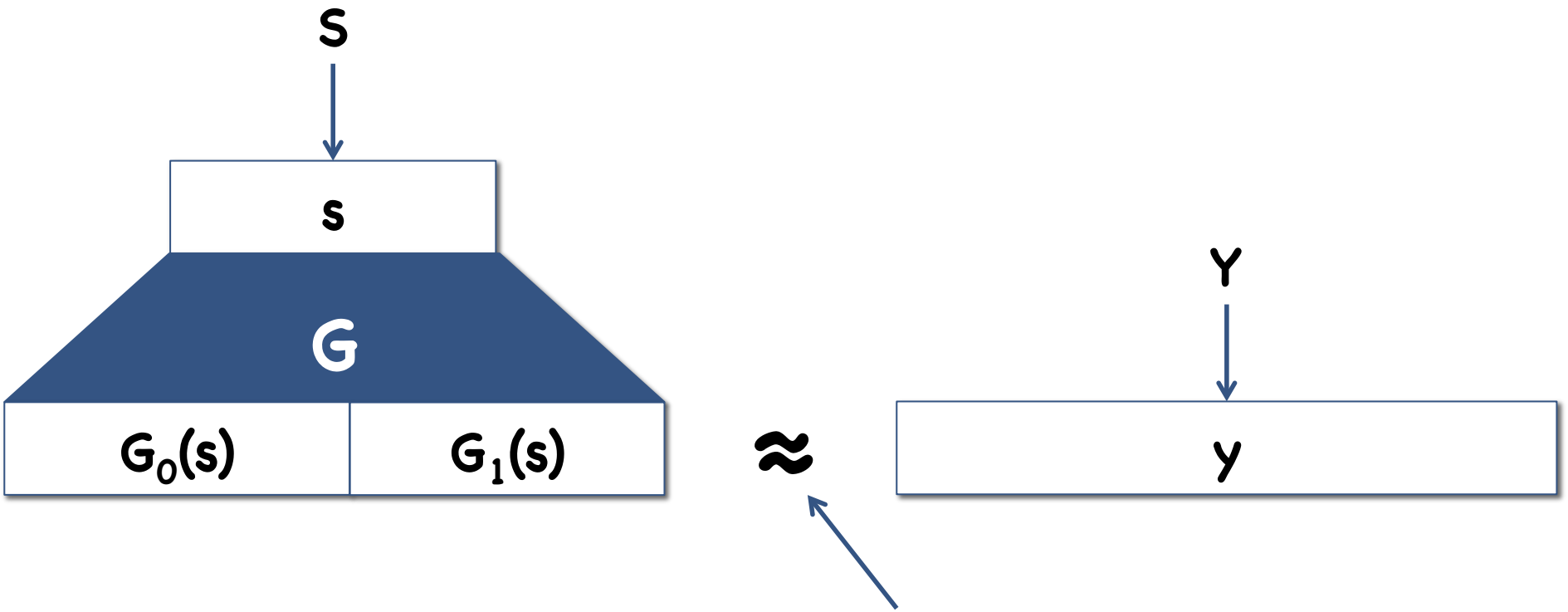
Hope that existing PQ-secure PRFs are FQ secure

Examples: GGM, NR, BPR

Questions:

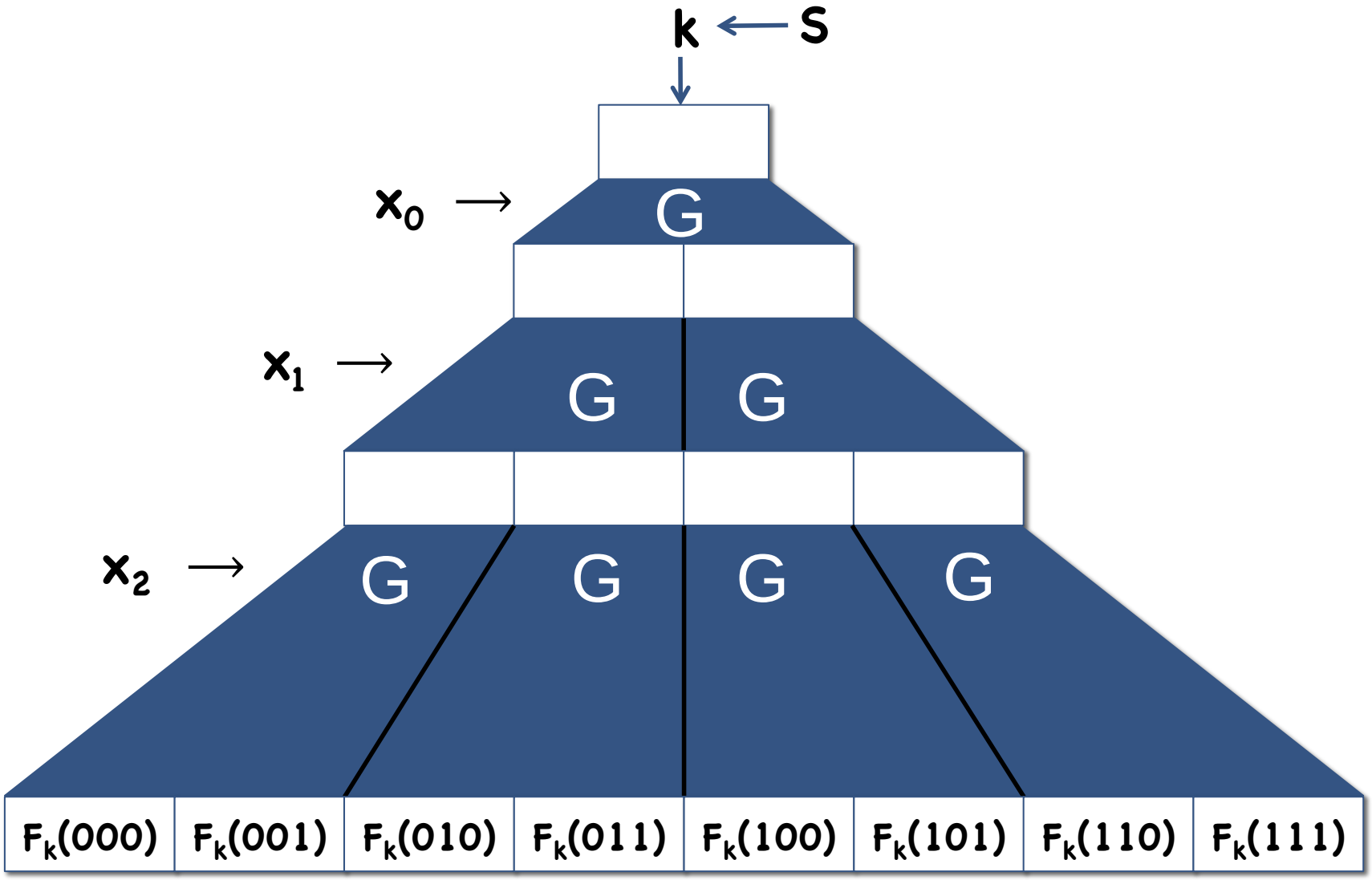
- Do classical security analyses carry over?
- If not, what new tools are needed?

Pseudorandom Generators

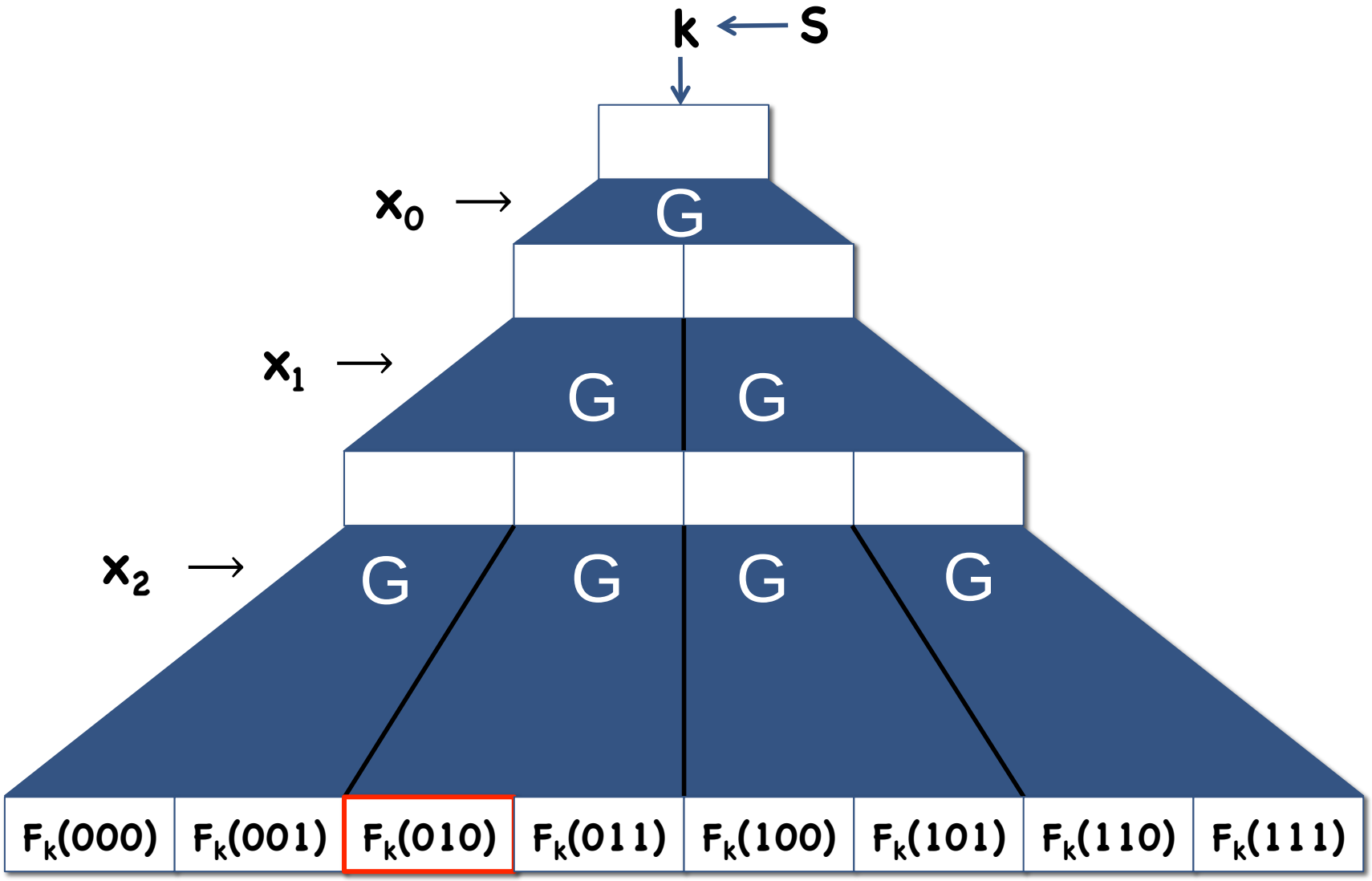


Indistinguishable by efficient quantum adversaries

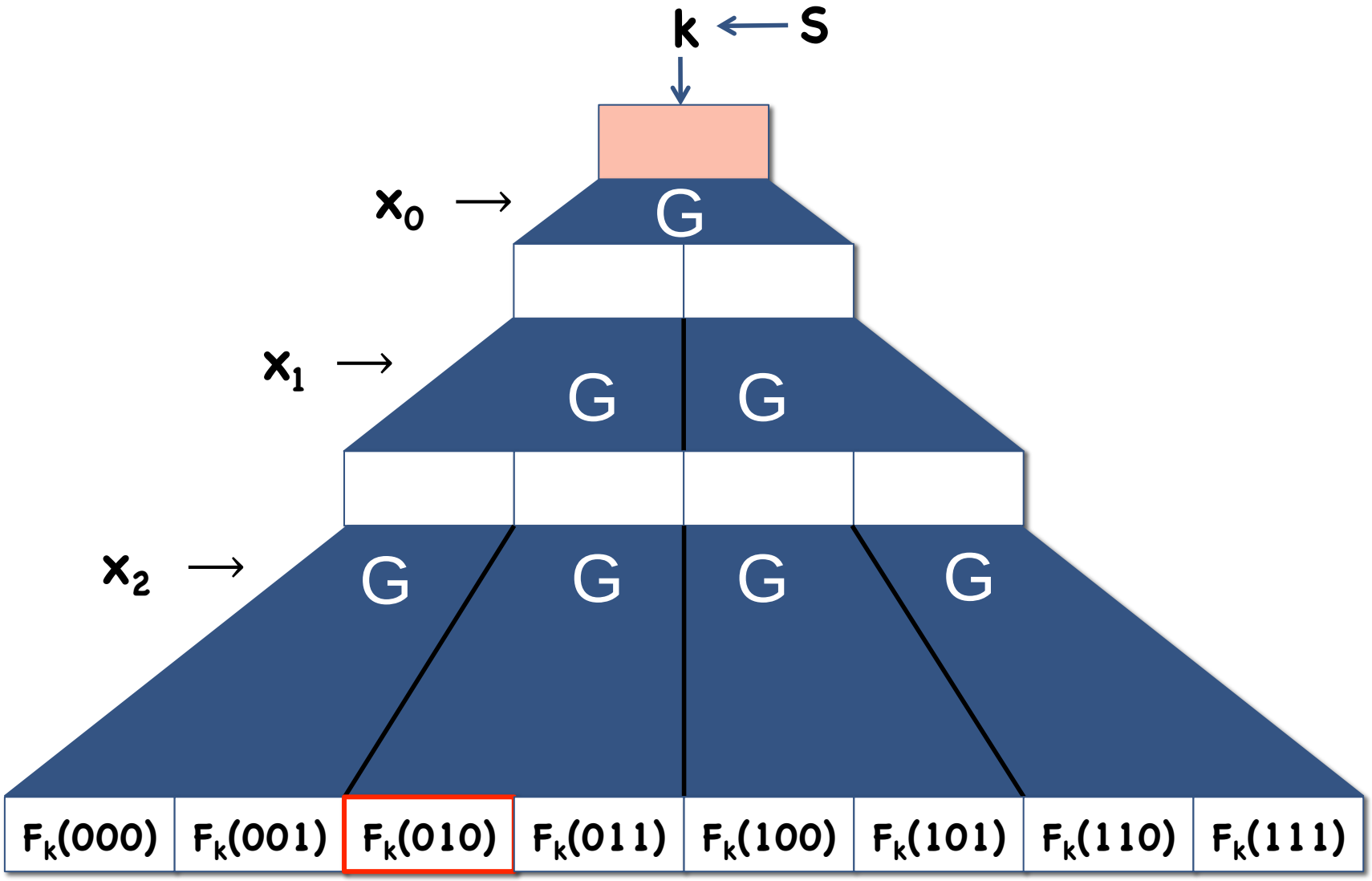
The GGM Construction



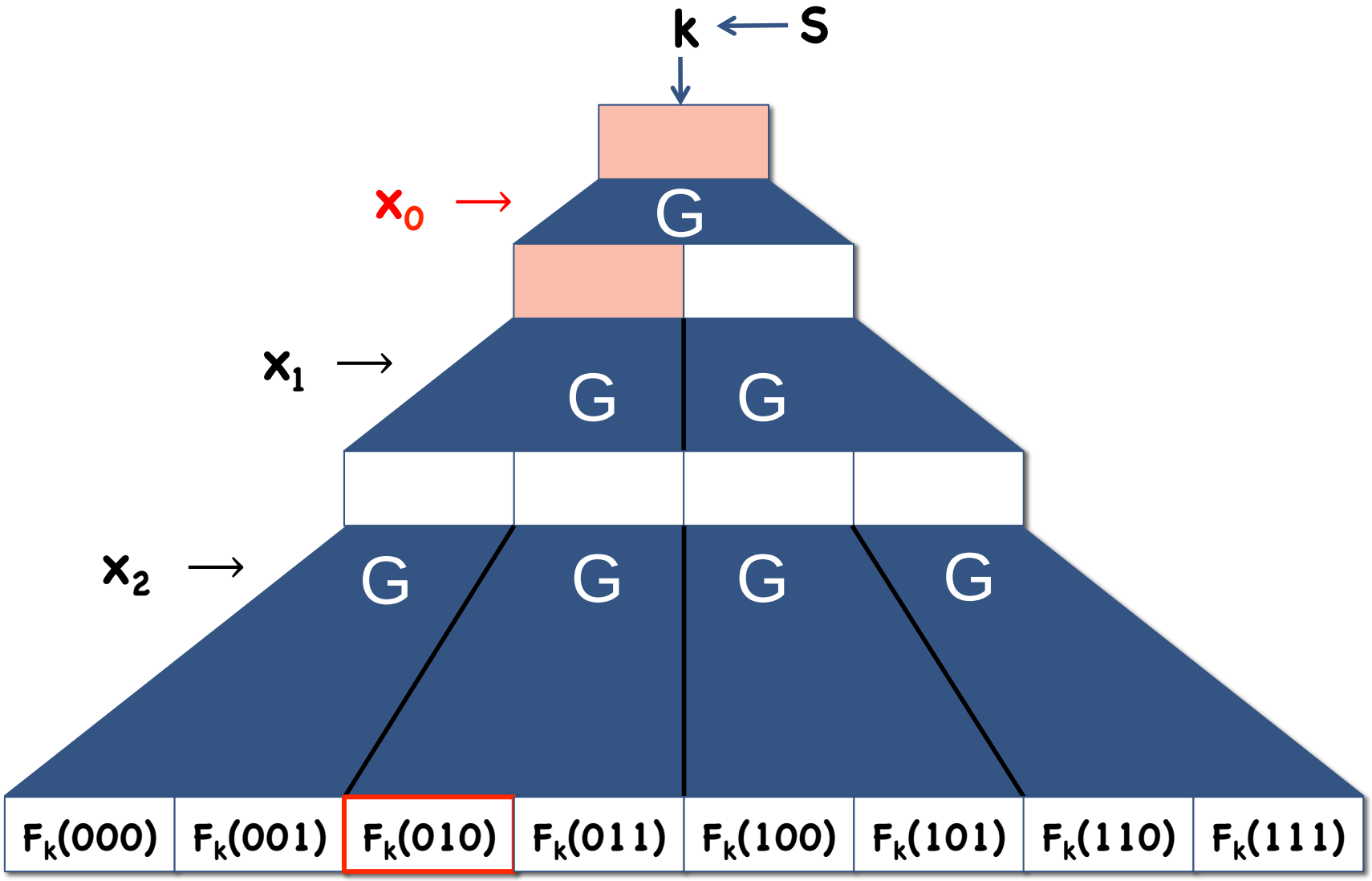
The GGM Construction



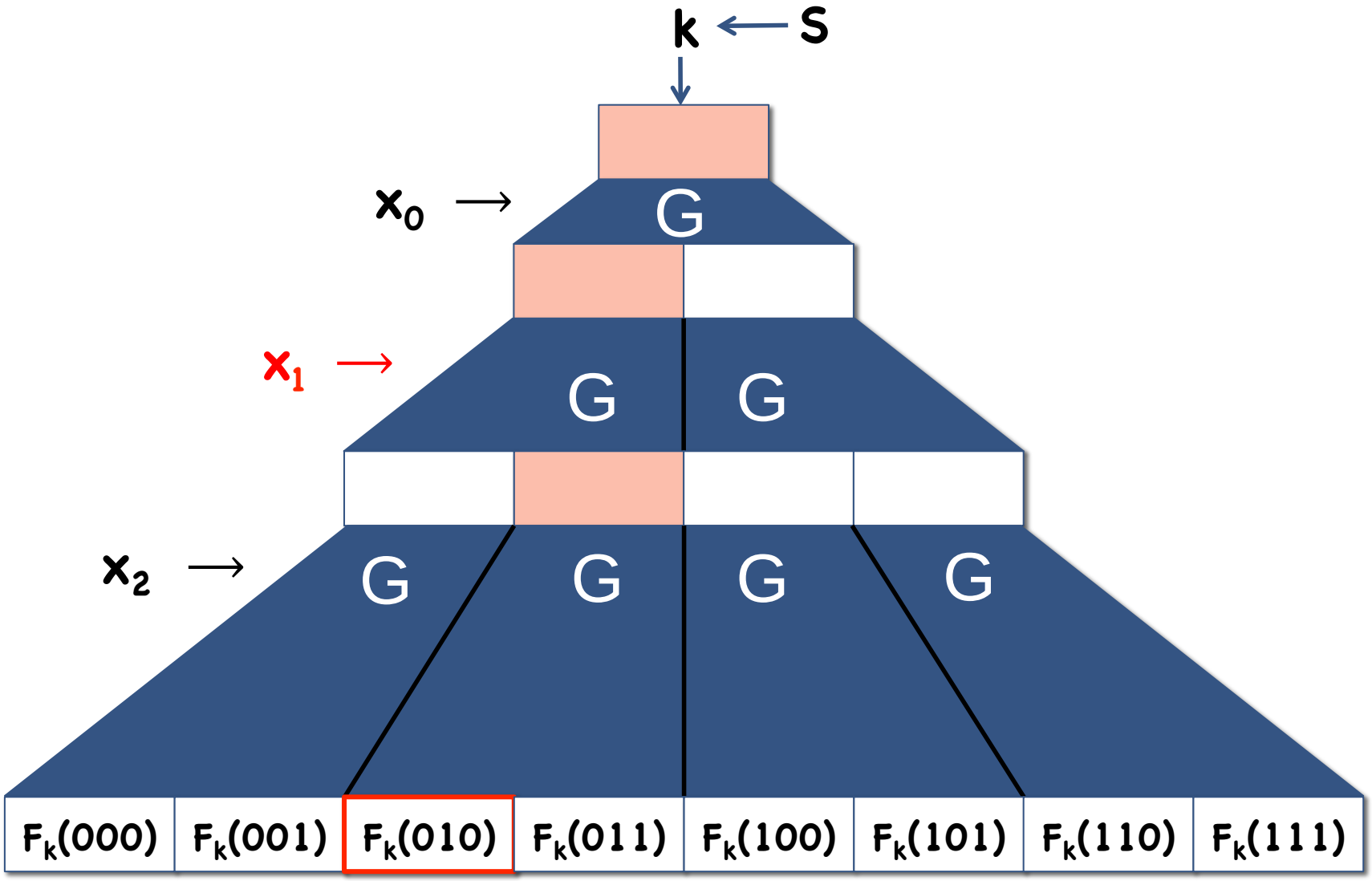
The GGM Construction



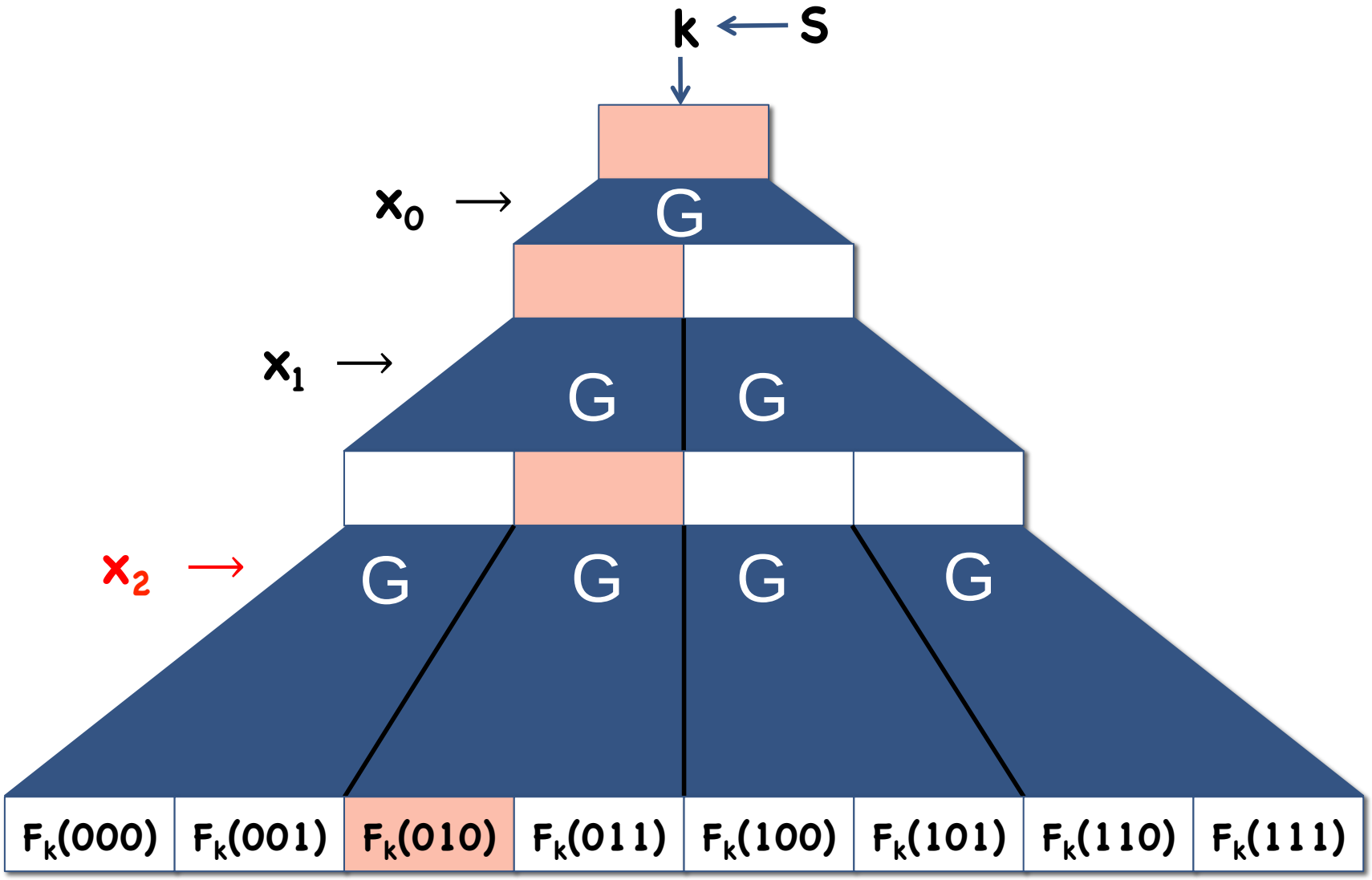
The GGM Construction



The GGM Construction



The GGM Construction



Quantum Security Proof?

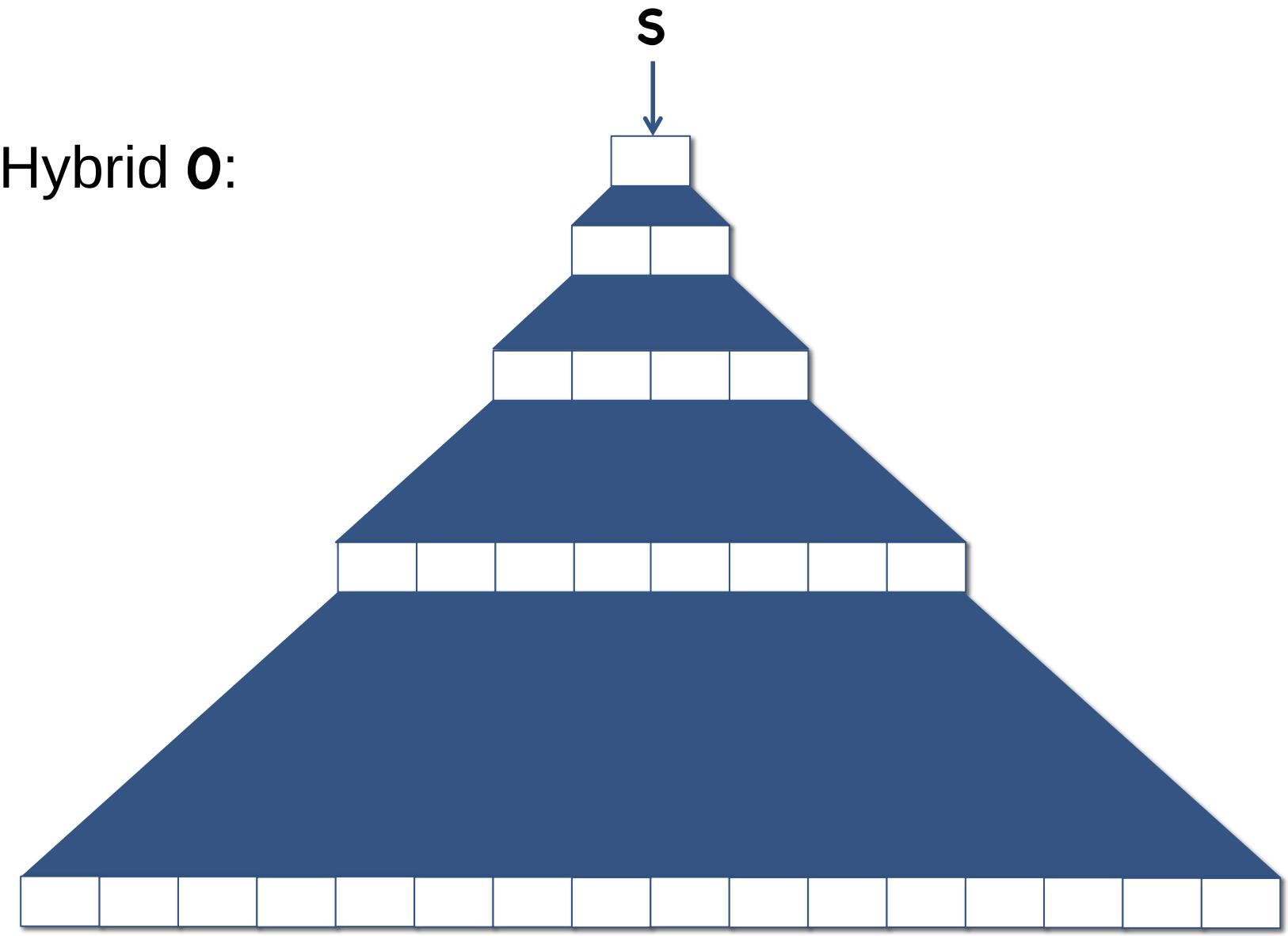
Idea: follow classical steps

- Turn PRF distinguisher into PRG distinguisher

Step 1: Hybridize over levels of tree

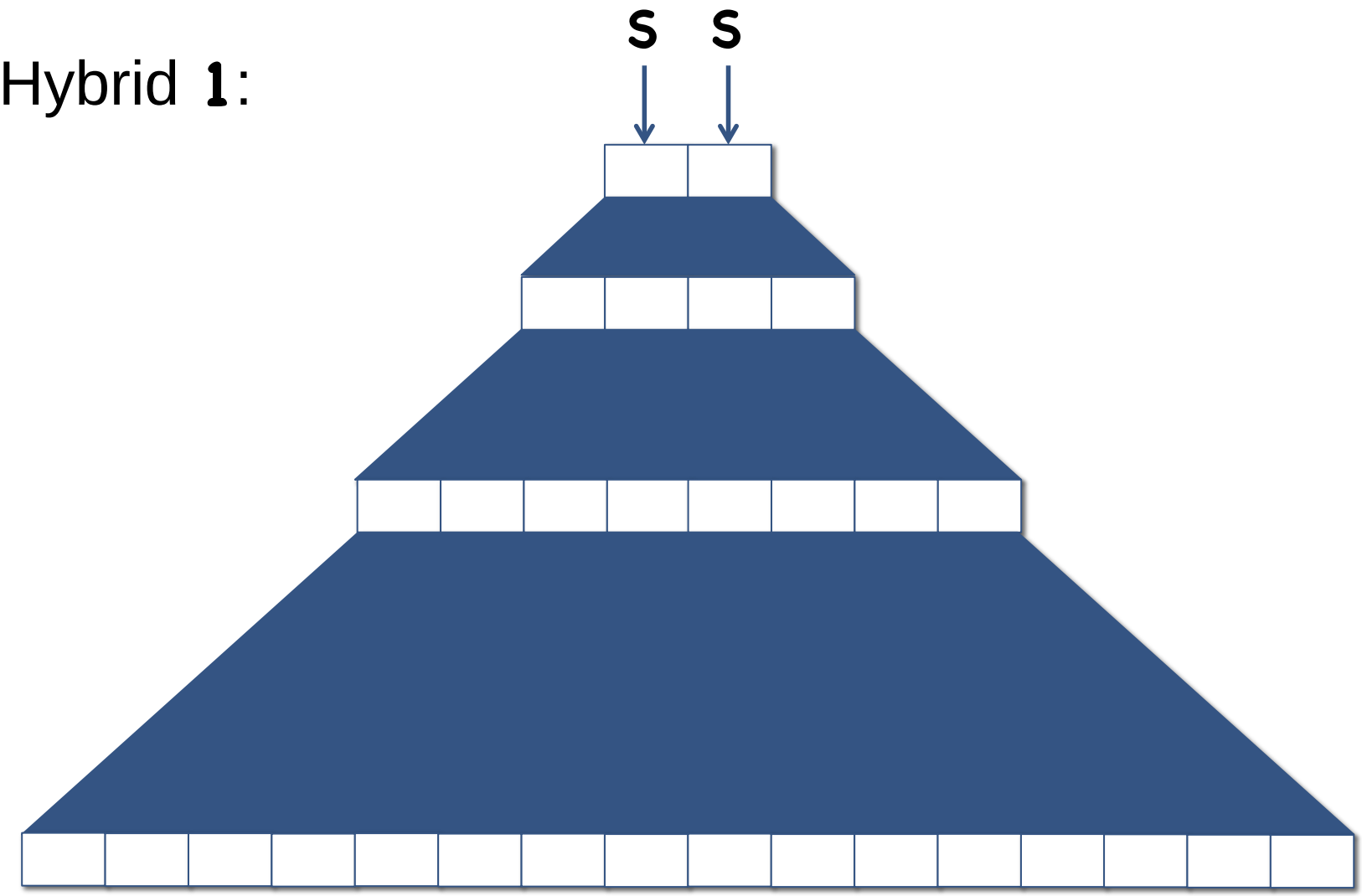
Hybridize Over Levels

Hybrid **0**:



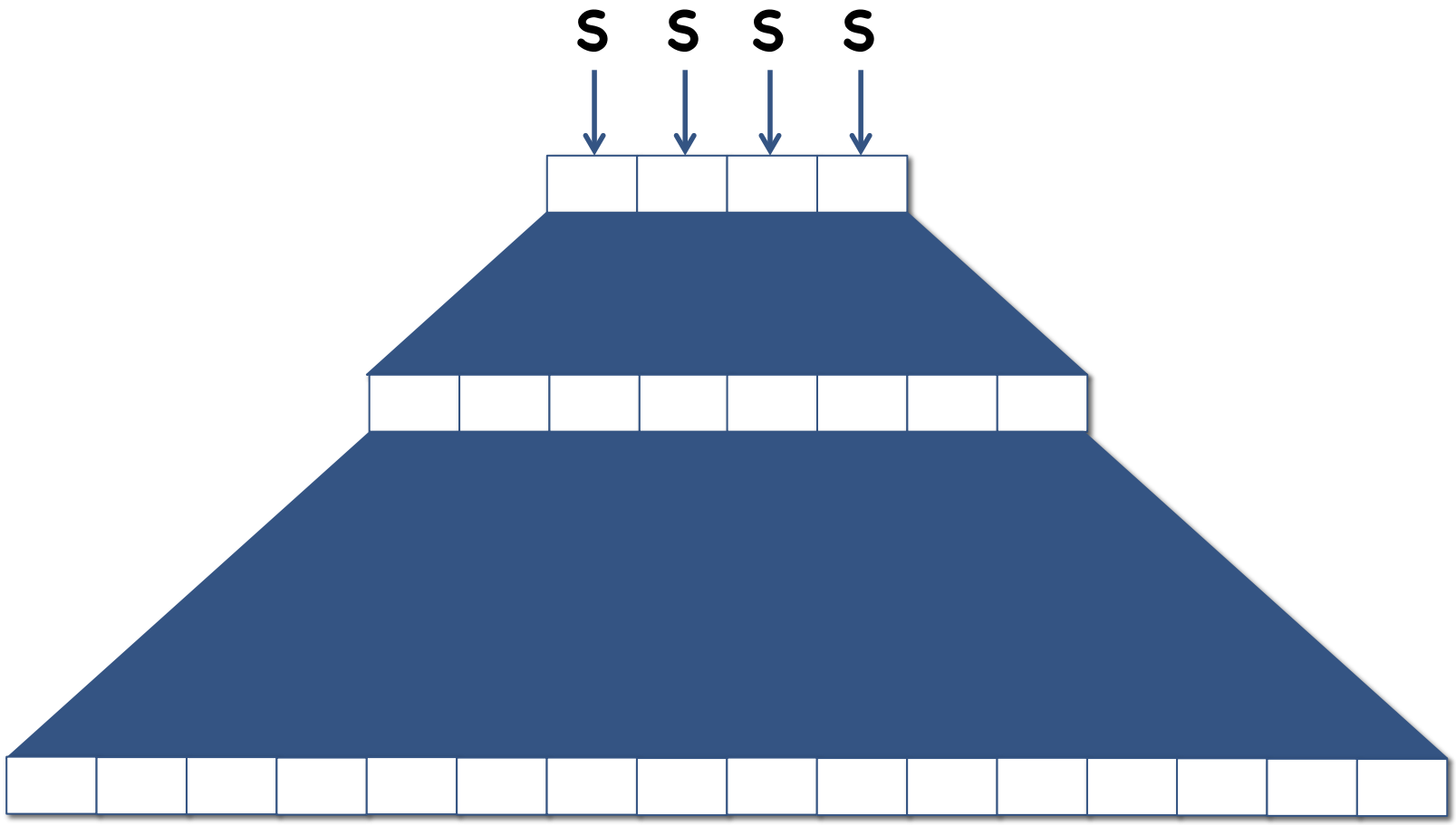
Hybridize Over Levels

Hybrid **1**:



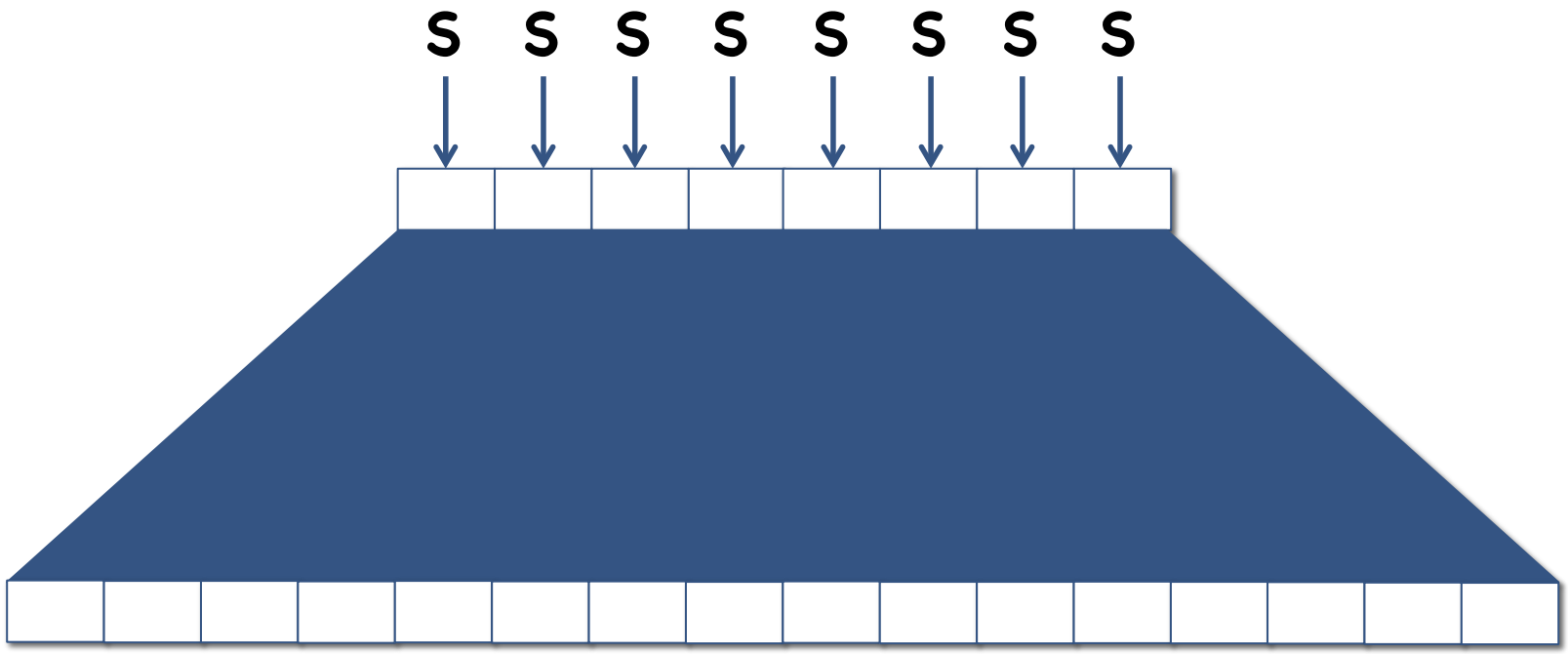
Hybridize Over Levels

Hybrid **2**:



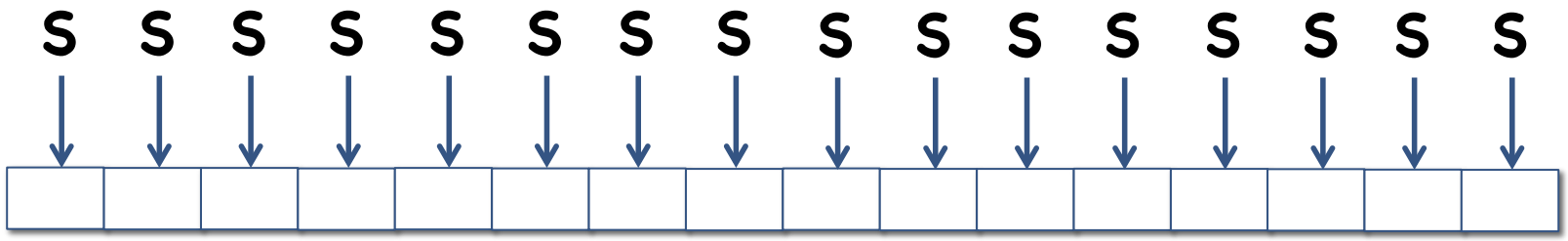
Hybridize Over Levels

Hybrid **3**:



Hybridize Over Levels

Hybrid **n**:



Hybridize Over Levels

Distinguish **PRF** from **Func(X,Y)** with adv. **ε**

↓

Distinguish two adjacent hybrids with adv. **ε/n**

n polynomial ⇒ acceptable loss

Hybridize Over Levels

Distinguish **PRF** from **Func(X,Y)** with adv. **ε**

↓

Distinguish two adjacent hybrids with adv. **ε/n**

n polynomial ⇒ acceptable loss

Argument carries over to quantum setting unmodified

Quantum Security Proof?

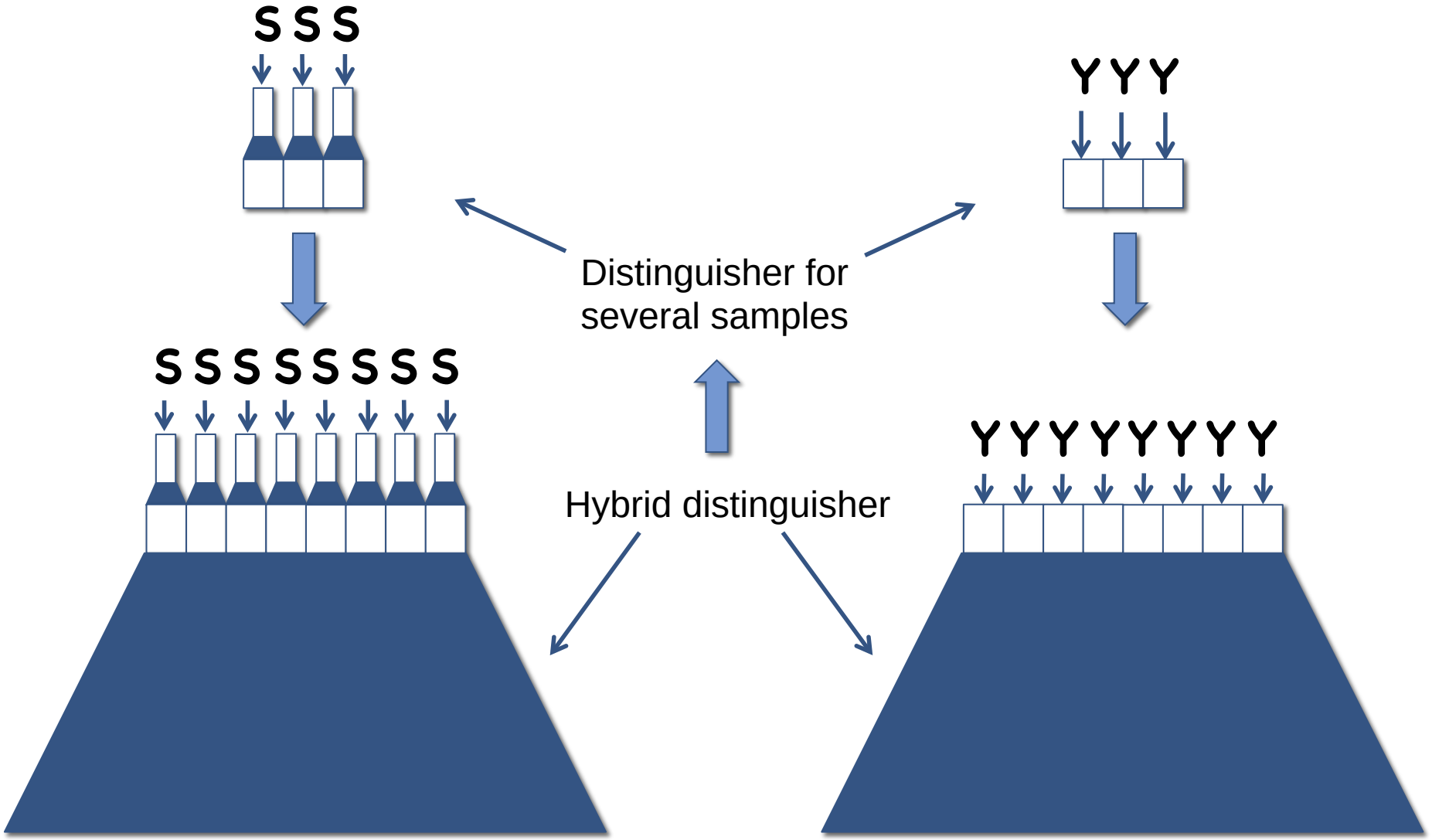
Idea: follow classical steps

- Turn PRF distinguisher into PRG distinguisher

Step 1: Hybridize over levels of tree ✓

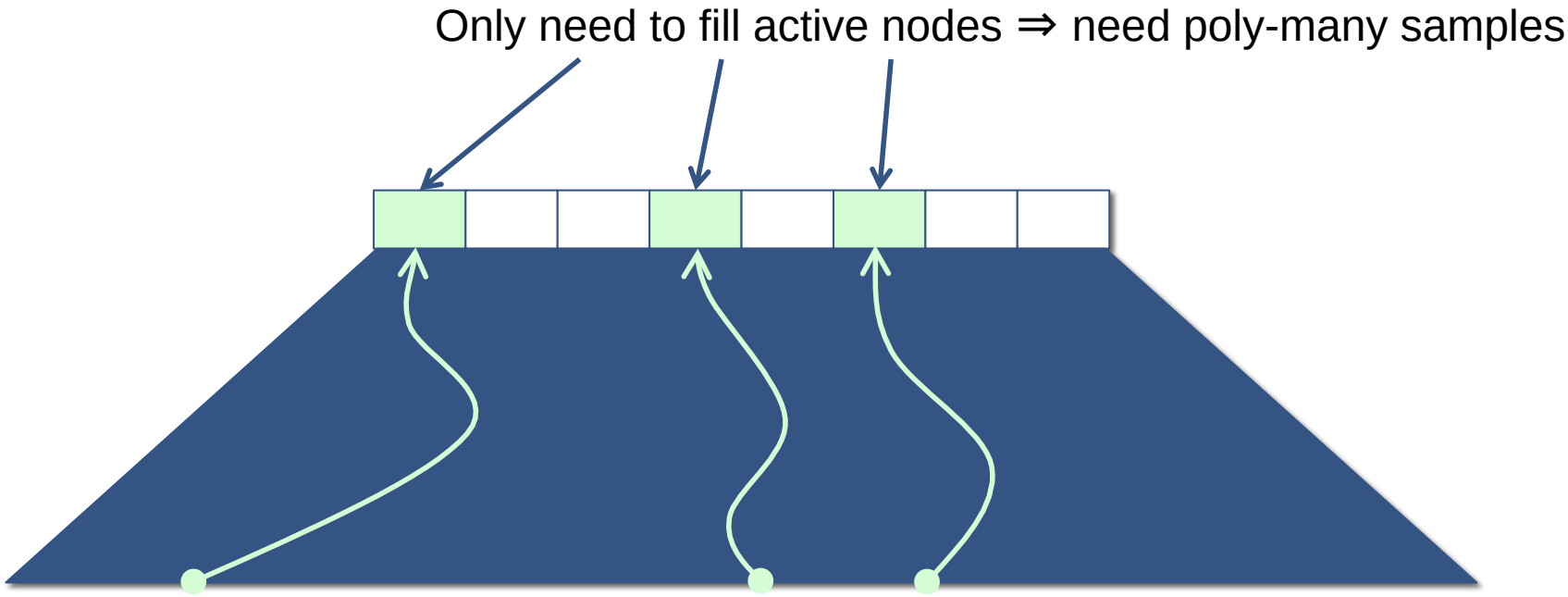
Step 2: Simulate hybrids using PRG/Random samples

Simulating Hybrids



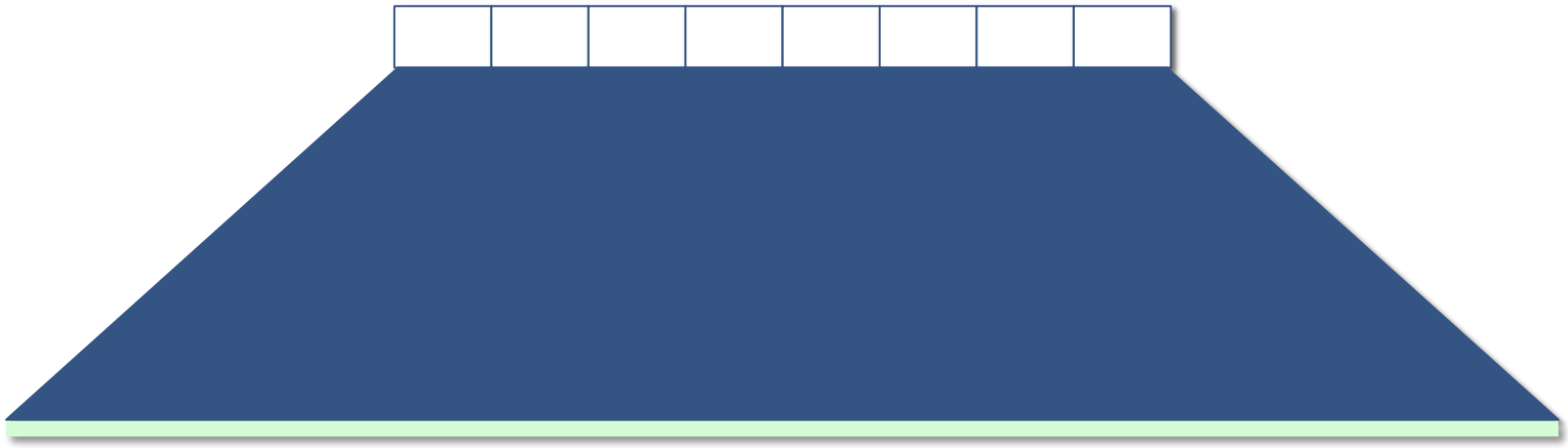
How It Was Done Classically

Active node: value used to answer query



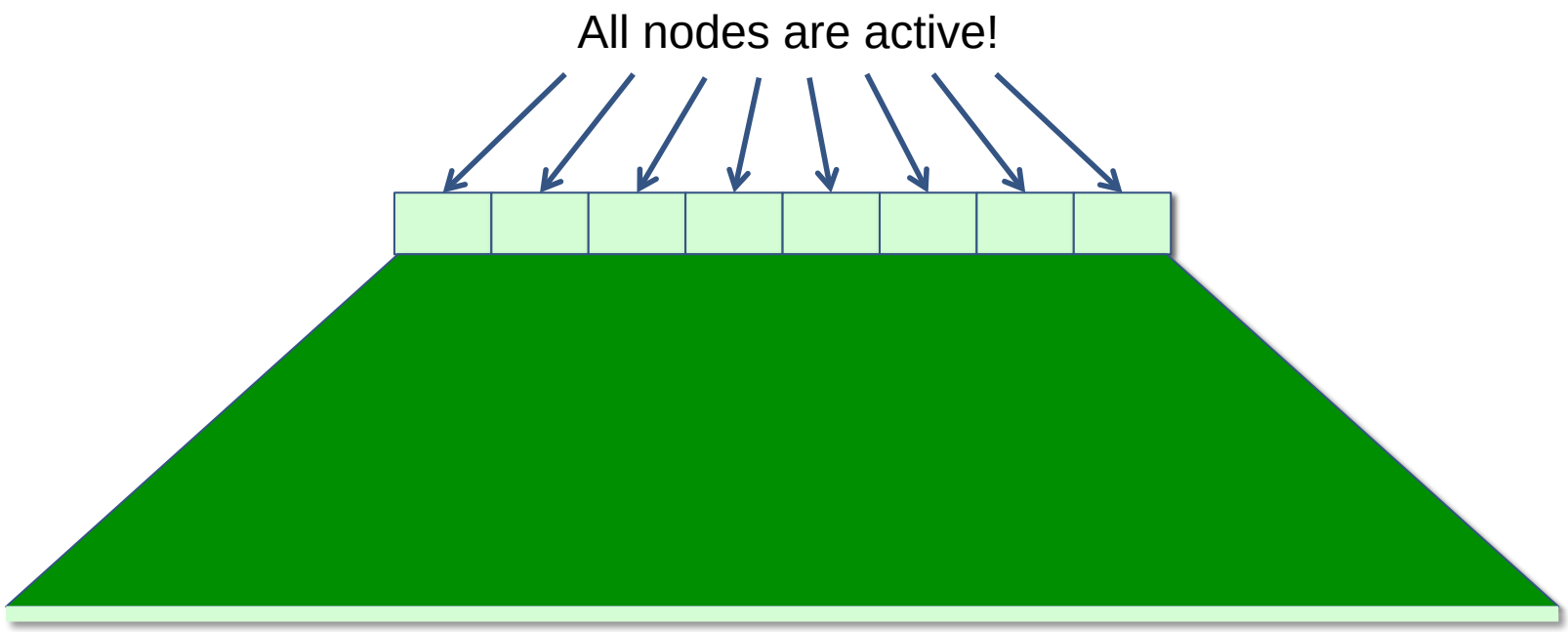
Adversary only queries polynomial number of points

Quantum Simulation?



Adversary can query on all exponentially-many inputs

Quantum Simulation?



Adversary can query on all exponentially-many inputs

Need exponentially many samples to simulate!

Quantum Security Proof?

Idea: follow classical steps

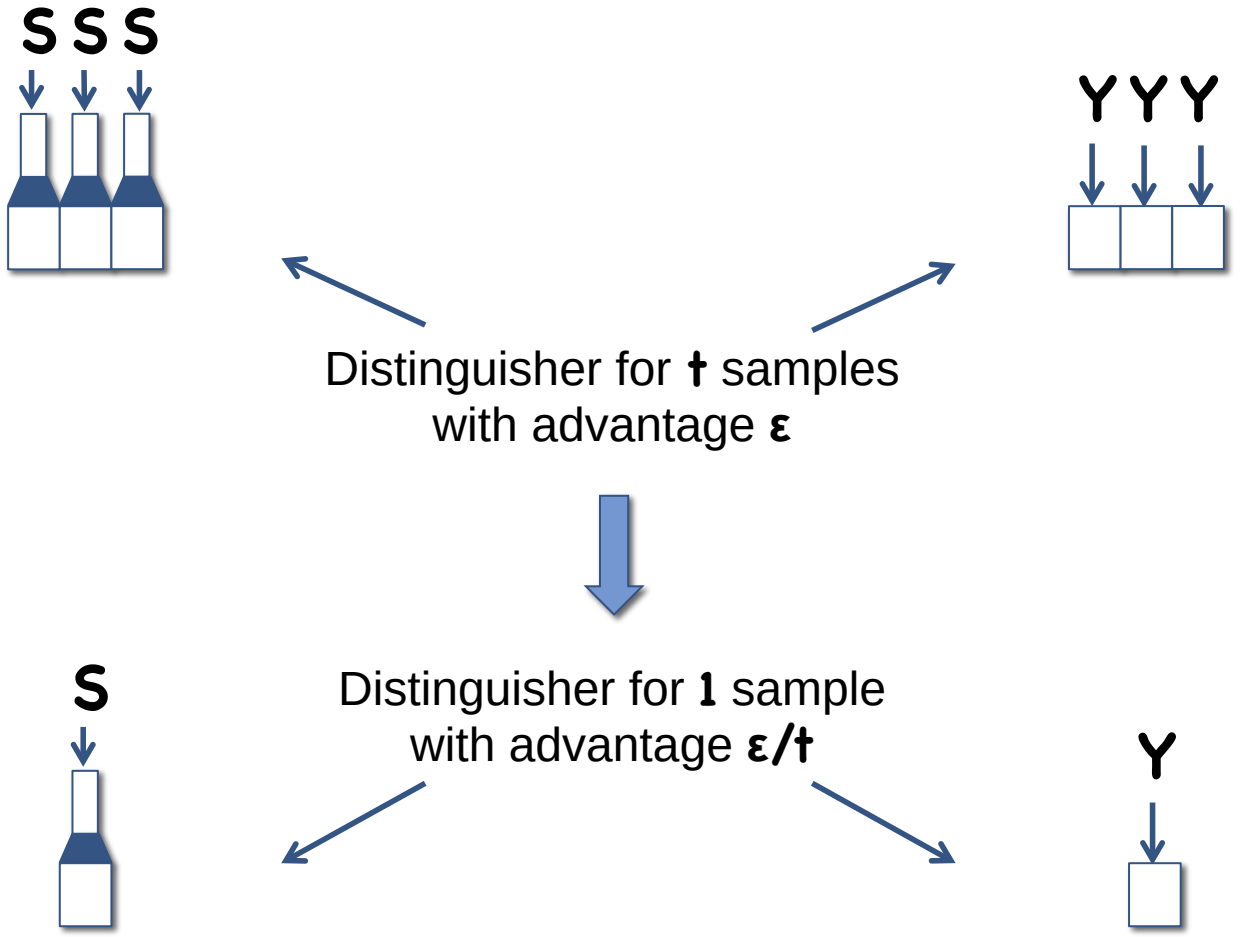
- Turn PRF distinguisher into PRG distinguisher

Step 1: Hybridize over levels of tree ✓

Step 2: Simulate hybrids using PRG/Random samples ?

Step 3: Hybrid over samples

Hybrid Over Samples



Argument carries over to quantum setting unmodified

Quantum Security Proof?

Idea: follow classical steps

- Turn PRF distinguisher into PRG distinguisher

Step 1: Hybridize over levels of tree ✓

Step 2: Simulate hybrids using PRG/Random samples ?

Step 3: Hybrid over samples ✓

- Exponential samples ⇒ exponential security loss
- Can only handle poly-many samples

Quantum Security Proof?

Idea: follow classical steps

- Turn PRF distinguisher into PRG distinguisher

Step 1: Hybridize over levels of tree ✓

Step 2: Simulate hybrids using PRG/Random samples ✗

Step 3: Hybrid over samples ✓

- Exponential samples ⇒ exponential security loss
- Can only handle poly-many samples

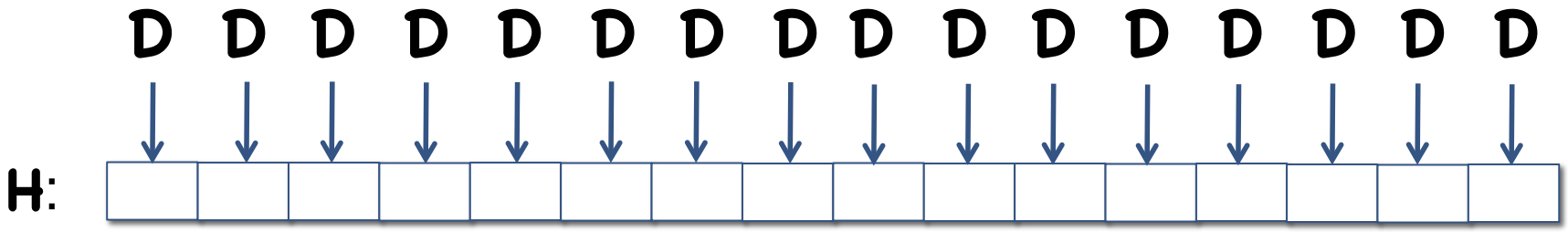
A Distribution to Simulate

Distribution D on Y $\Rightarrow$ induces distribution on functions

For all $x \in X$:

$y_x \leftarrow D$

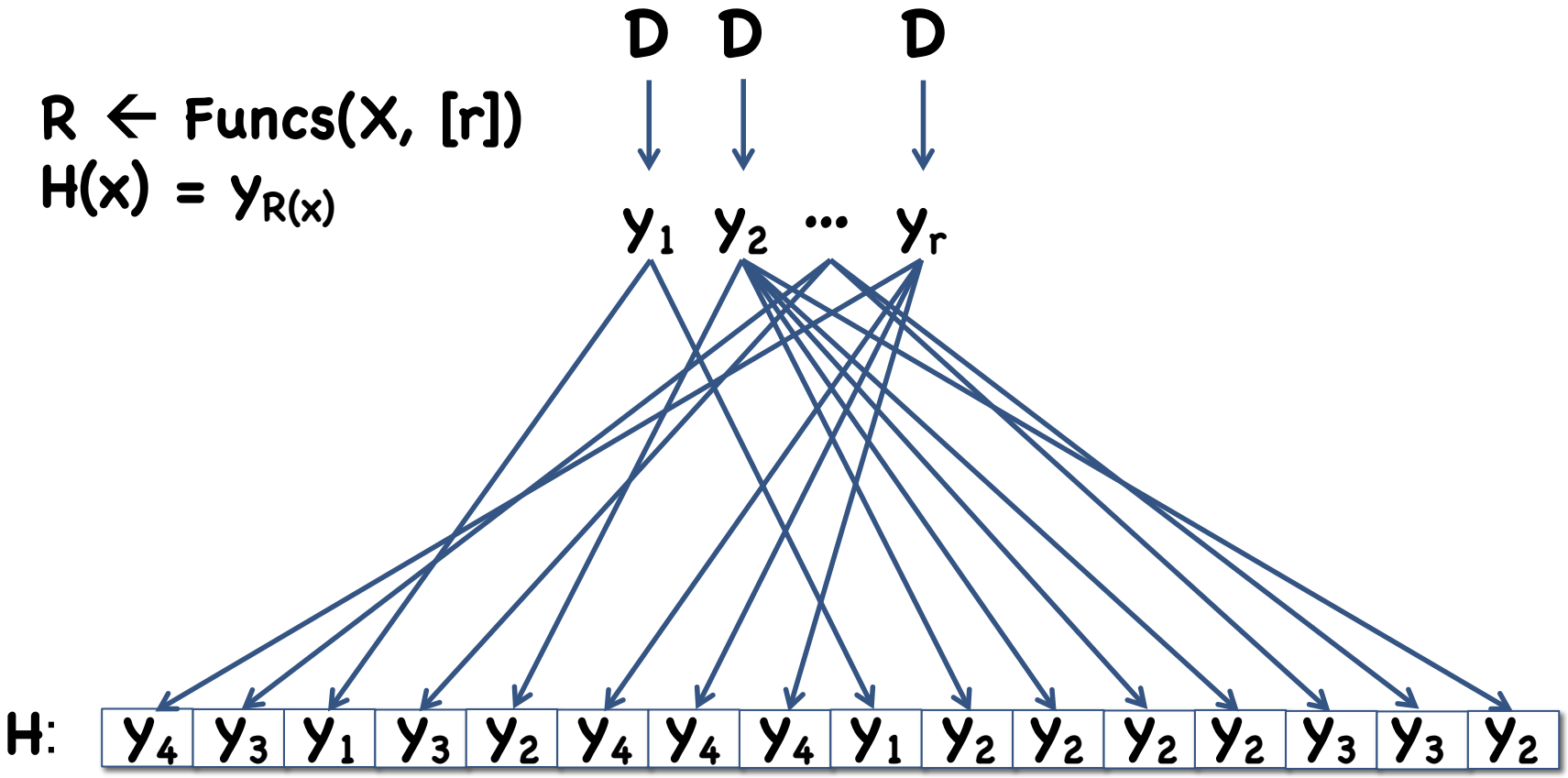
$H(x) = y_x$



$H \leftarrow D^X$

Goal: simulate using poly-many samples

Solution: Small-Range Distributions



Small-Range Distributions

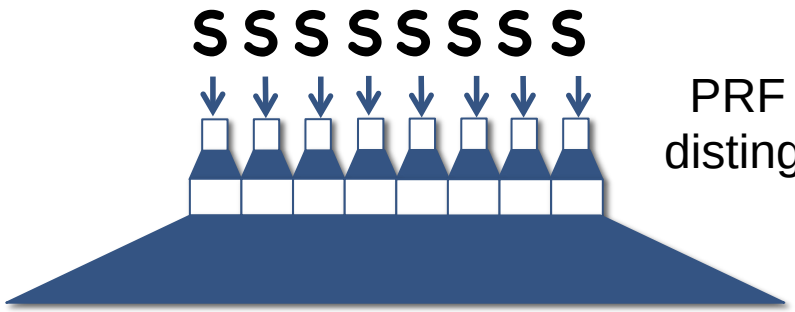
Theorem: $SR_r^X(D)$ is indistinguishable from D^X by any q-query quantum algorithm, except with advantage $O(q^3/r)$

Notes:

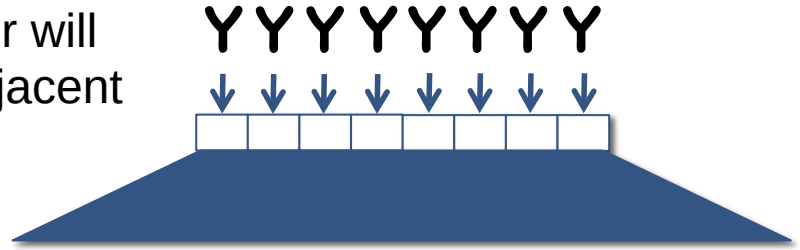
- Highly non-trivial
- Distinguishing prob not negligible, but good enough
  - We get to choose r
- Random function R not efficiently constructible

Theorem: Can simulate R using k-wise independence

Quantum GGM Proof

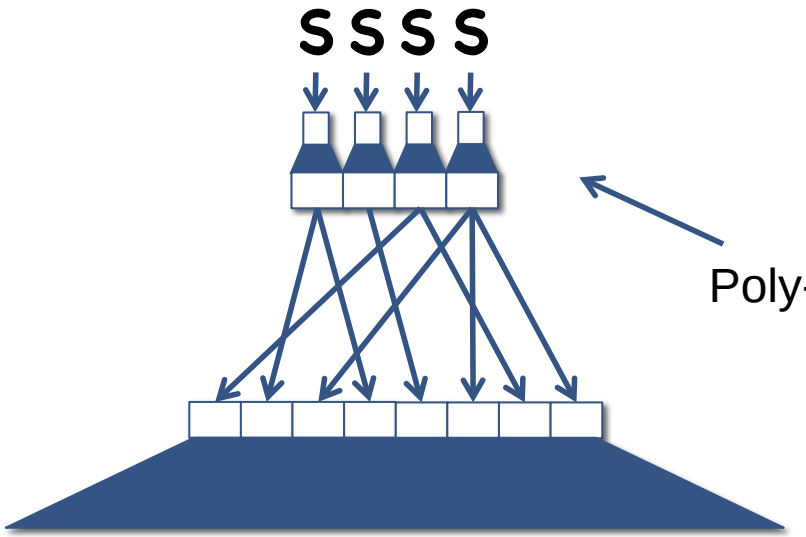


PRF distinguisher will distinguish two adjacent hybrids

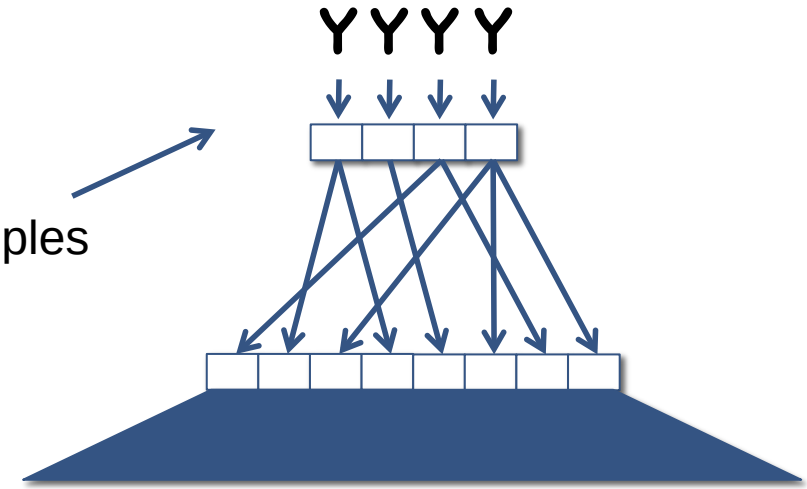


≈ (SR distributions)

≈ (SR distributions)



Poly-many samples



Quantum Security Proof

Idea: follow classical steps

- Turn PRF distinguisher into PRG distinguisher

Step 1: Hybridize over levels of tree ✓

Step 2: Approx. sim. hybrids using poly-many samples ✓

Step 3: Hybrid over samples ✓

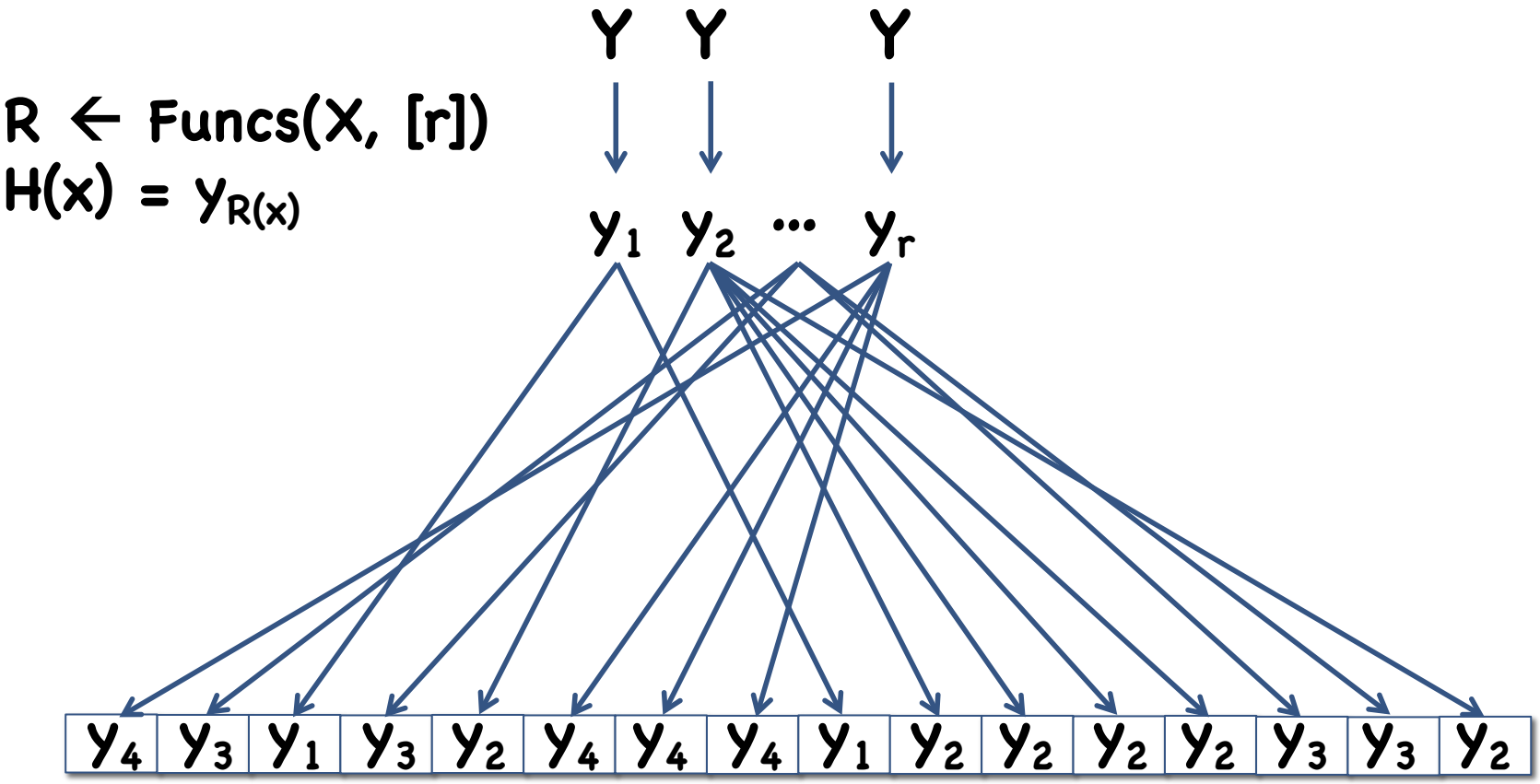
Result: PRG distinguisher

Impossible by assumption ⇒ PRF distinguisher impossible

Quantum Query Results

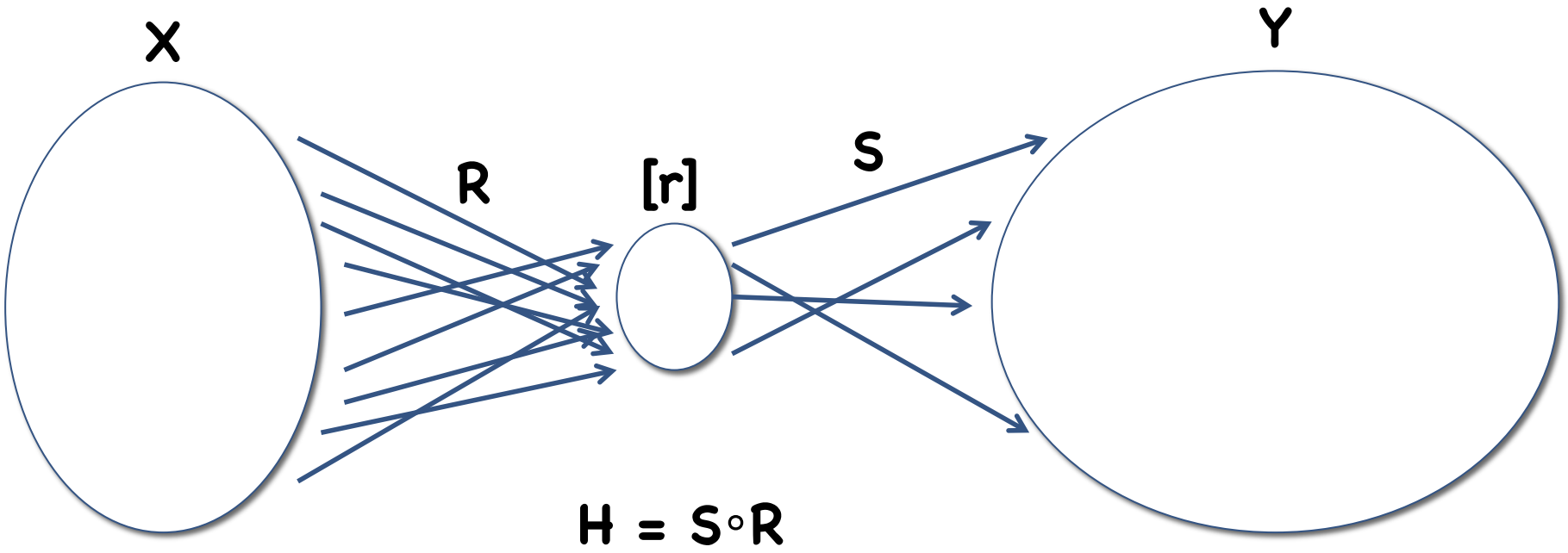
Quantum Collision Finding

Recall small-range distributions when **D** is uniform:



Quantum Collision Finding

Another view:



Theorem: H is indistinguishable from random by any q-query quantum algorithm, except with advantage $O(q^3/r)$

Corollary: If $|Y| >> |X|^2$, impossible to find collision in H unless $q \geq \Omega(r^{1/3})$

Quantum Collision Finding

Corollary: If $|Y| >> |X|^2$, impossible to find collision in H unless $q \geq O(r^{1/3})$

What about truly random functions with $|Y| << |X|^2$?

Theorem: $q \geq \Omega(r^{1/3})$ quantum queries are required to find collisions in a random function $R: X \rightarrow [r]$

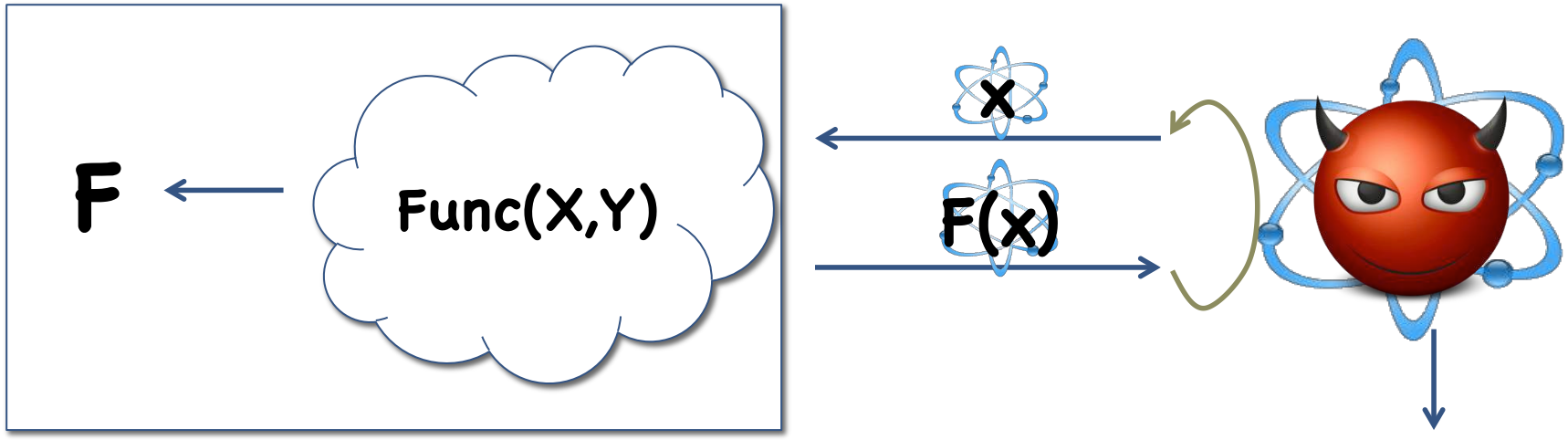
Previous $r^{1/3}$ lower bounds known for different settings

- E.g. k-to-1 functions [AS'01]
- All prior settings required $|Range| \geq |Domain|$
- Our works for all domain/range sizes

Bound is tight: [BHT'97] $q = O(r^{1/3})$

Quantum Oracle Interrogation

Using **q** queries, determine function at **k>q** points



$(x_1, F(x_1)), (x_2, F(x_2)), \ldots (x_k, F(x_k))$

Important for MAC, signature security

Quantum Oracle Interrogation

Classically: hard $\quad\quad\quad\quad\quad\quad\quad\quad$ **Adv = $1/|Y|^{k-q}$**

- Large outputs: **Adv = negl** even for **k=q+1**
- Small outputs: **Adv = negl** for **k = c q**

Quantum: not so fast

> **Theorem [vD'98]**: For **F:X→{0,1}**,
> **q** quantum queries ⇒ **k = 1.9q** points w.h.p

Also true for small ranges:

> **Theorem**: For **F:X→$\{0,1\}^2$**,
> **q** quantum queries ⇒ **k = 1.3q** points w.h.p

Question: What about large range sizes?

Quantum Oracle Interrogation

Theorem: For $F:X \to \{0,1\}^n$,
q quantum queries $\Rightarrow \Pr[k=q+1 \text{ points}] \leq (q+1)/2^n$

Highly non-trivial

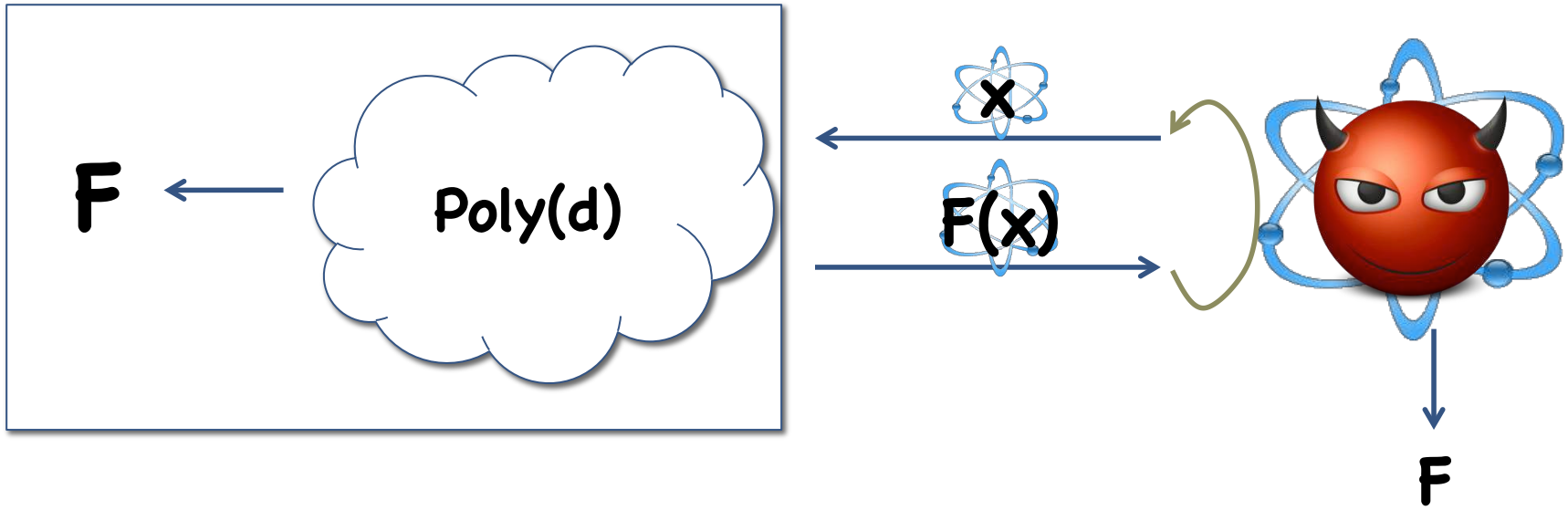
New quantum impossibility tool: **The Rank Method**

Therefore:

- Small range: $\Pr[q+1 \text{ points}]$ large
- Large range: $\Pr[q+1 \text{ points}]$ small

Quantum Polynomial Interpolation

Using **q** queries to a polynomial, determine polynomial



Classical: **q=d+1** → easy
q<d+1 → hard

Quantum: **q=d/2** → hard [KK'10]

Theorem: (quantum) **q=d** → easy

Theorem: (quantum) **q=(d+1)/2**, "large" **d** → hard

Conclusion

Studying full quantum security notions important

- Quantum computers seem inevitable
- Unclear what attacks are possible
- Strive for strongest definitions
- **Bonus:** quantum query complexity results

Future work: more advanced primitives

- Identity-based encryption
- Functional encryption
- Fully homomorphic encryption
- Other quantum query questions

Acknowledgements

?