

CS 258: Quantum Cryptography

Mark Zhandry

Final Logistics

Available on Gradescope from 12/8 – 12/12

Any 3 hour increment

Completed individually (including no AI)

Handwritten expected. Open computer, notes, internet

Today: quantum crypto beyond CS 258

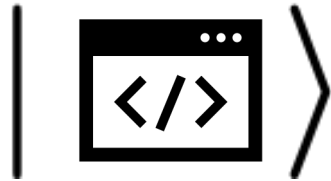
Other Topics in Unclonable Cryptography

(Public Key) Quantum Money are states that are
unclonable (and publicly verifiable)

But what if we want the unclonable states to
actually do something?

Quantum Copy Protection

Able to evaluate program on
arbitrary inputs



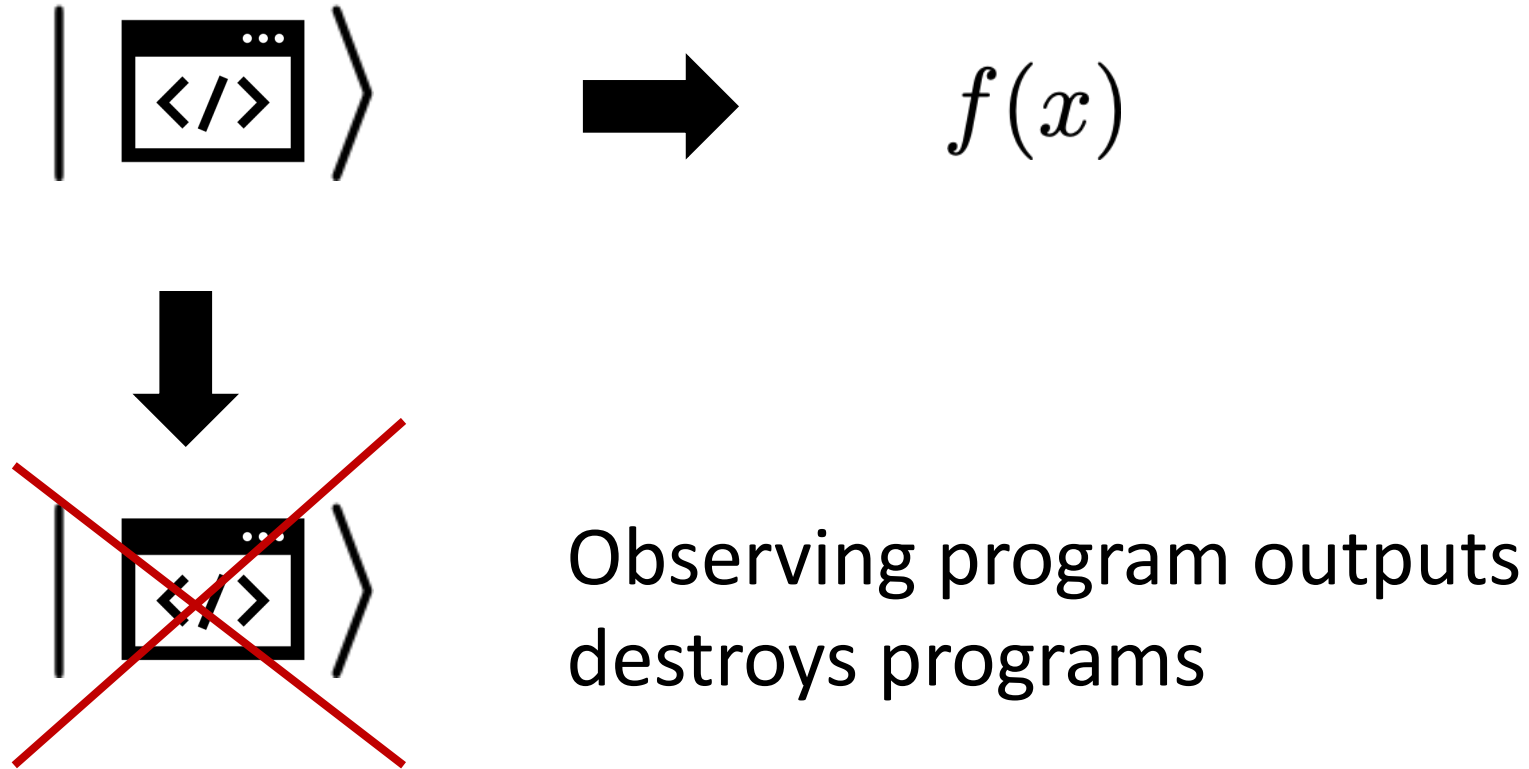
But unable to produce two
working copies of program

Cannot literally just encode classical program description in a ket. Instead need an inherently quantum state that hides underlying program

Have constructions for specific functions using classical obfuscation + quantum money ideas, but for general programs, area fairly open.

Also know some impossibilities for general programs

Quantum One-time Programs



Thm: Quantum OTP impossible for deterministic functionalities

Idea: Gentle measurements mean we can learn $f(x)$ without actually perturbing original program

Next goal: OTP for probabilistic functionalities

We know a few specific functionalities that can be OTP'ed, but otherwise pretty wide open

Even a general meaningful
definition is still largely open

Example: $f(x) = (g(x), r)$ where r random for each evaluation

Given quantum program that computes $f(x)$, can devise quantum program that computes $g(x)$.

Since g is deterministic, can apply Gentle Measurements on g

Thus, just insisting on probabilistic outputs is insufficient

One-time Signatures (aka Signature Tokens)

$$f(x) = \text{random signature on } x$$

Meaningful one-time security requirement: impossible to obtain signatures on two different messages

One-*Shot* Signatures

Adversary cannot produce signatures on two messages, even if the adversary devises the public key

Recall

Def (Commitment, Computational Sum-Binding): A commitment scheme is **classically/quantumly sum-binding** if, for all PPT/QPT adversaries \mathcal{A} , there exists a negligible function ϵ such that

$$\Pr[W_0] + \Pr[W_1] \leq 1 + \epsilon(\lambda)$$

where $W_b(\lambda)$ is the event that \mathcal{A} succeeds in the following:

- \mathcal{A} produces a commitment c and two msgs $m_0, m_1 \in \{0, 1\}^*$ *of the same length*
- Give b to \mathcal{A}
- \mathcal{A} tries to output $r \in \{0, 1\}^\lambda$ s.t. $c = \text{Com}(m_b, r)$

Hash functions are good commitments

$$\text{Com}(m, r) = H(m, r)$$

Lemma (informal): If H is collapsing, then Com is post-quantum sum-binding

Lemma turns out to be if and only if

Theorem: Collision-resistant but not collapsing hash \rightarrow One Shot Signatures

Key Gen: take $\sum_{x,z} \alpha_{x,z} |x, z\rangle$ produced by collapsing adversary

Measure $H(x) \rightarrow$ result y is public key

Leftover state $C \sum_{\substack{x,z \\ H(x)=y}} \alpha_{x,z} |x, z\rangle$ is money state

Theorem: Collision-resistant but not collapsing hash \rightarrow One Shot Signatures

Verification: Signature on message bit b = string s
such that $H(b, s) = y$

Signing: Use collapsing adversary to break sum-binding

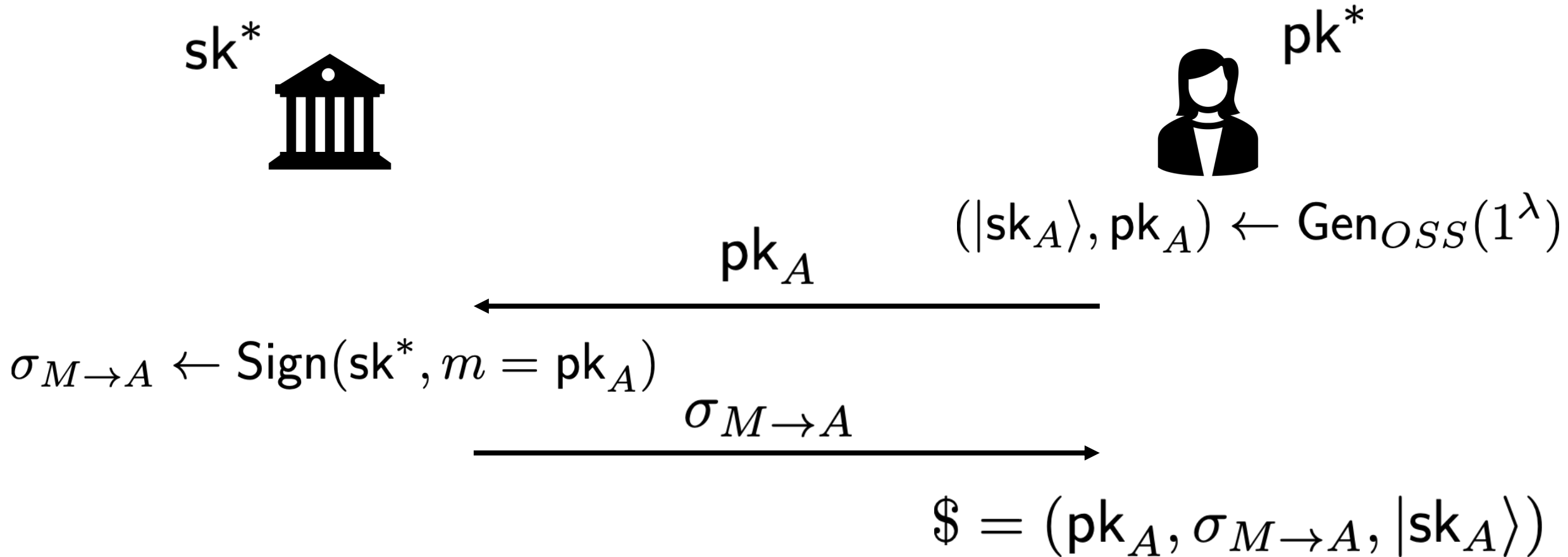
Quantum Money with Classical Communication

Mint's Key Gen: plain classical signature scheme

Mint keeps secret key sk^* to self

Mint publishes public key pk^*


Minting a banknote + sending to Alice



Notice that mint is completely classical!

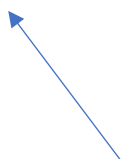
Verifying

$$\$ = (\text{pk}_A, \sigma_{M \rightarrow A}, |\text{sk}_A\rangle)$$

Run $\text{Ver}(\text{pk}^*, m = \text{pk}_A, \sigma_{M \rightarrow A})$  Verifies that pk_A signed by mint

Choose a random message m and run

$$\text{Ver}_{OSS}(\text{pk}_A, m, \text{Sign}_{OSS}(|\text{sk}_A\rangle, m))$$

 Verifies that $|\text{sk}_A\rangle$ is a valid secret key for pk_A

Security

Suppose adversary produces two banknotes that pass verification

$$\text{\$} = (\text{pk}_A, \sigma_{M \rightarrow A}, |\text{sk}_A\rangle) \quad \text{\$}' = (\text{pk}'_A, \sigma'_{M \rightarrow A}, |\text{sk}'_A\rangle)$$

Case 1: $\text{pk}_A \neq \text{pk}'_A$

One of them must be different from original
Alice's public key, which is impossible by ordinary
signature security

Security

Suppose adversary produces two banknotes that pass verification

$$\text{\$} = (\text{pk}_A, \sigma_{M \rightarrow A}, |\text{sk}_A\rangle) \quad \text{\$}' = (\text{pk}'_A, \sigma'_{M \rightarrow A}, |\text{sk}'_A\rangle)$$

Case 2: $\text{pk}_A = \text{pk}'_A$

Adversary then has two keys which can sign messages relative to pk_A . In particular, can sign two different messages, breaking OSS security

Alice sending to Bob

$$\text{pk}^* \text{ Alice } \$ = (\text{pk}_A, \sigma_{M \rightarrow A}, |\text{sk}_A\rangle)$$



$$\text{pk}^* \text{ Bob }$$



$$(|\text{sk}_B\rangle, \text{pk}_B) \leftarrow \text{Gen}_{OSS}(1^\lambda)$$

pk_B



$$\sigma_{A \rightarrow B} \leftarrow \text{Sign}(|\text{sk}_A\rangle, m = \text{pk}_B)$$

$\text{pk}_A, \sigma_{M \rightarrow A}, \sigma_{A \rightarrow B}$



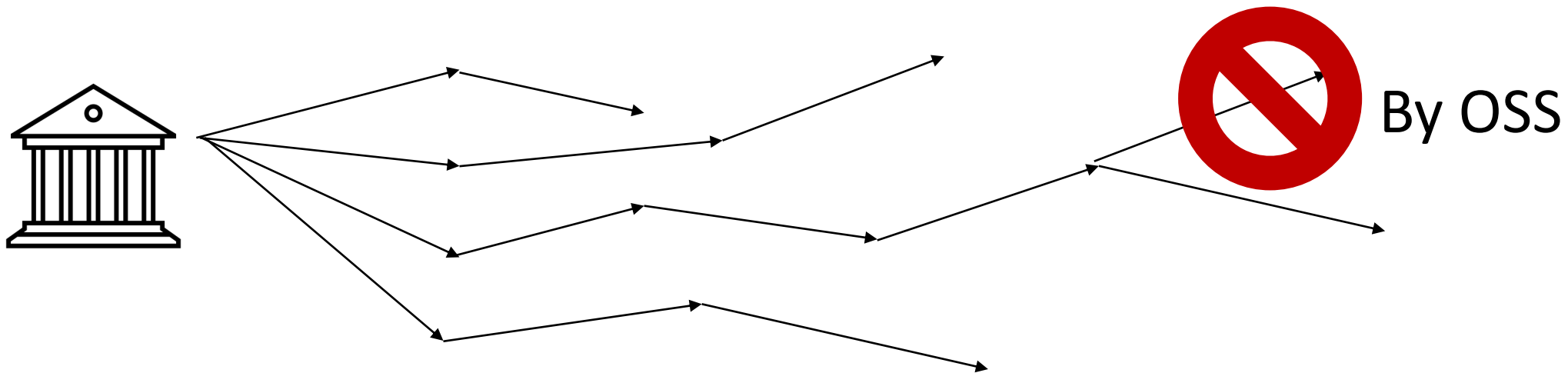
$$\$' = (\text{pk}_A, \sigma_{M \rightarrow A}, \text{pk}_B, \sigma_{A \rightarrow B}, |\text{sk}_B\rangle)$$

$$\mathcal{S}' = (\text{pk}_A, \sigma_{M \rightarrow A}, \text{pk}_B, \sigma_{A \rightarrow B}, |\text{sk}_B\rangle)$$

Verifies that pk_A
signed by mint

Verifies that pk_B
signed by Alice

By OSS security, Alice
can't sign anything else!



Proofs of Quantumness

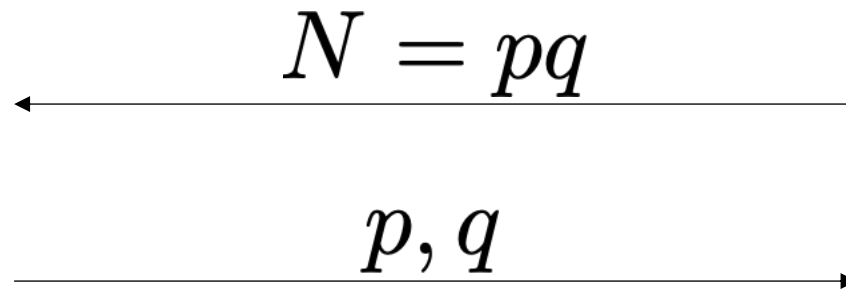


I have a quantum
computer

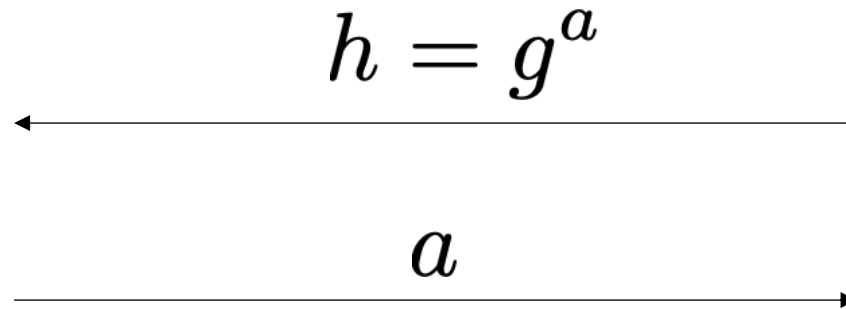


Prove it! (BTW, I'm
classical)

Proofs of quantumness from quantum advantage



Proofs of quantumness from quantum advantage

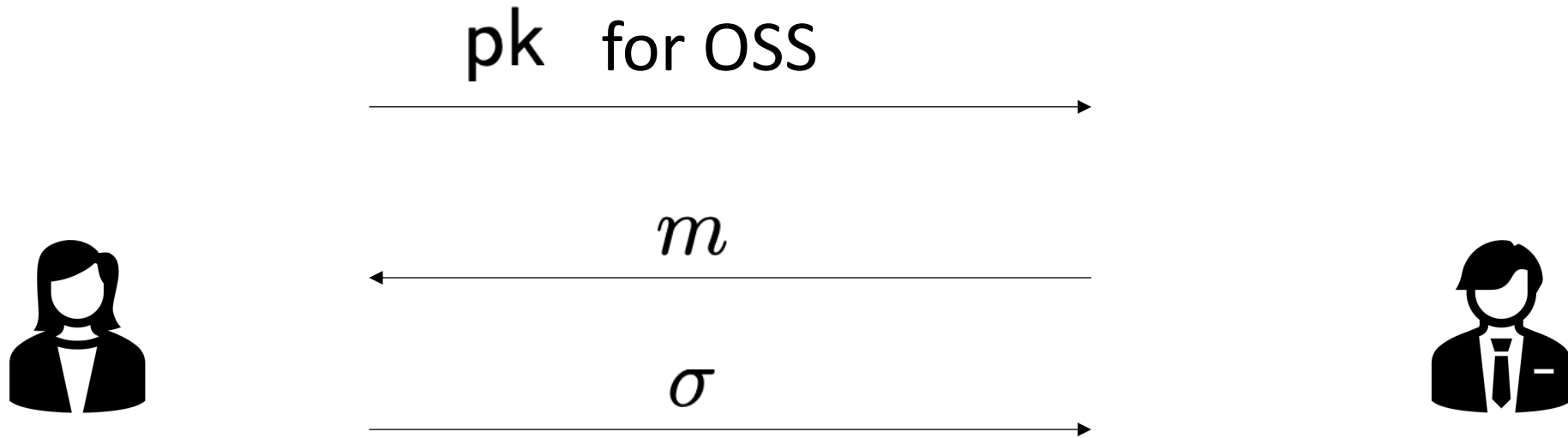


Efficient quantum computation

Can you prove quantum advantage if $BPP = BQP$?

Efficient classical computation
(with randomness)

Proofs of quantumness from hardness of rewinding



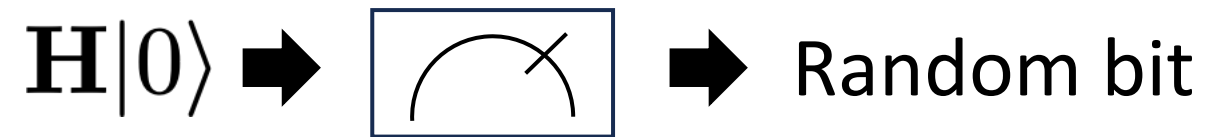
If Alice was classical, then could rewind her to sign multiple messages

Observe that hardness follows from an assumption about hardness for quantum algorithms. Could exist even if $BPP = BQP$

Certified Randomness

Classical computers are not very good at
generating quality randomness

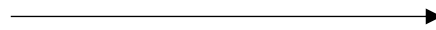
But quantum computers are fantastic!



Problem: what if you don't trust the quantum computer?

Supposedly
quantum device

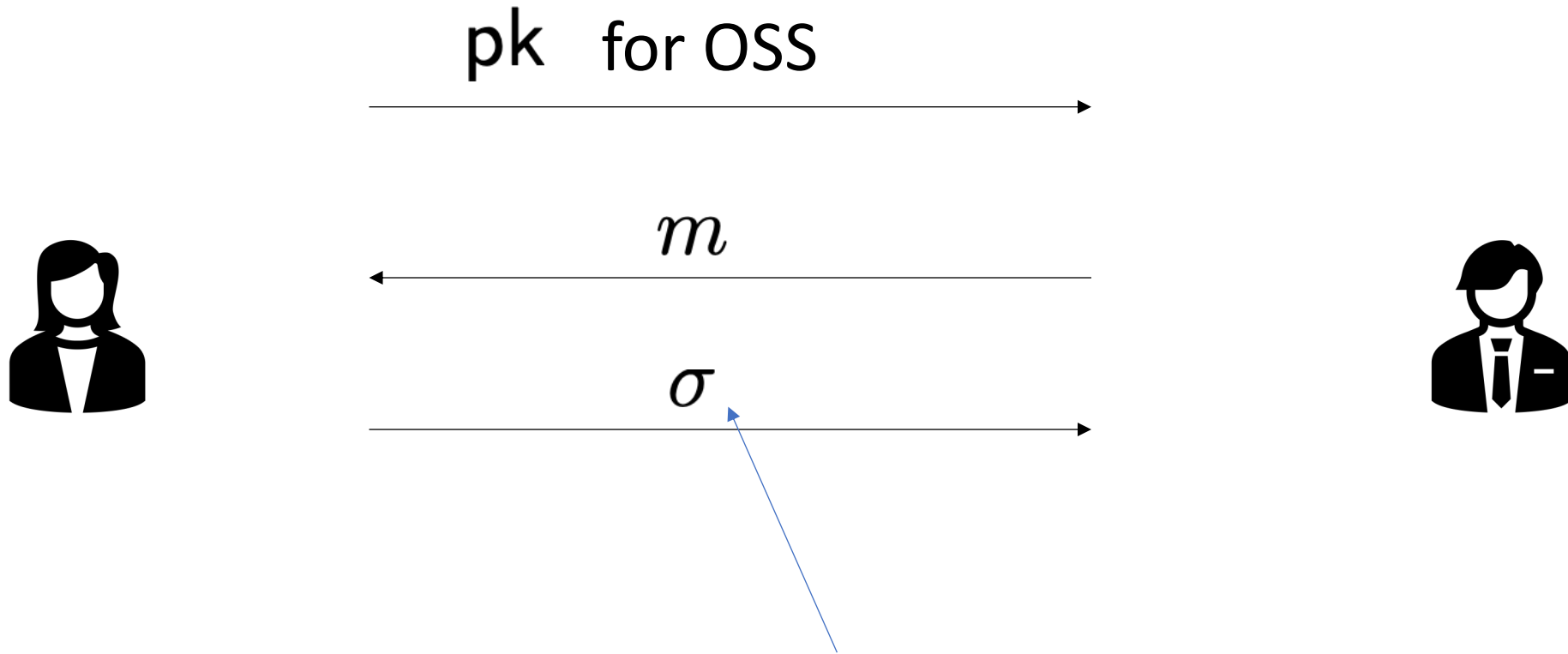
1101101000111011111110001110



Is this actually random?

Maybe box is deterministic or very low entropy

Maybe string was generated randomly by device
manufacturer and hardcoded into device



Must have entropy, or else can
rewind by gentle measurements

The issues:

- Bob already needs entropy to generate m
- Potentially only a small amount of entropy gained through σ

Solution: run protocol many times

- Bob generates his messages *pseudorandomly* from small seed
- Since Alice cannot distinguish from actual random messages, must give entropic signatures
- By running protocol many times, Bob gets a lot of entropy, but his seed length stays small

OSS definitely overkill for certified randomness/ proofs of quantumness

Constructions from:

- LWE
- Group actions
- Hash functions modeled as random oracles
- Even random quantum circuits

Many other topics

Connection to quantum gravity/black holes

Quantum query complexity

Encryption/authentication of quantum messages

Superposition attacks

Other post-quantum crypto assumptions (LPN, Code-based crypto)

NISQ applications of quantum computers

Position verification

Numerous quantum analogs of fundamental classical results

...

Thanks for a great quarter!