

COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

Spring 2020

Announcements

HW7 Due April 30th

Project 3/HW 8 due May 12th

Previously on COS 433...

Zero Knowledge Proofs

Interactive Proof

Statement x

Witness w



Properties of Interactive Proofs

Let (P, V) be a pair of probabilistic interactive algorithms for the proof system

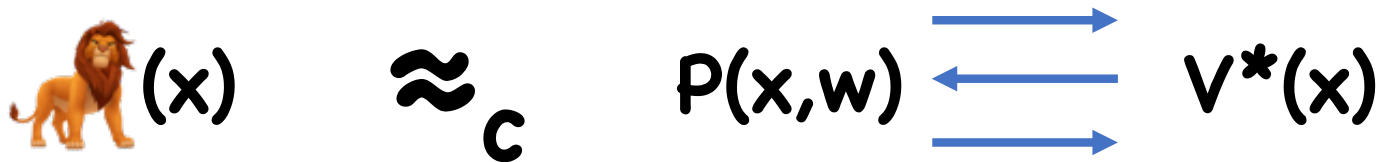
Completeness: If w is a valid witness for x , then V should always accept

Soundness: If x is false, then no cheating prover can cause V to accept

- Perfect: accept with probability 0
- Statistical: accept with negligible probability
- Computational: cheating prover is comp. bounded

Zero Knowledge

For every malicious verifier V^* , \exists “simulator” 
s.t. for every true statement x , valid witness w ,



QR Protocol

Statements: x is a Q.R. mod N

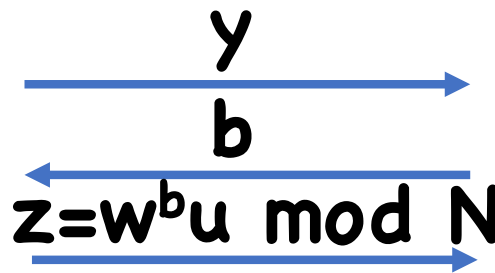
Witness: w s.t. $w^2 \bmod N = x$

Protocol:

$u \leftarrow \mathbb{Z}_N^*$
 $y \leftarrow u^2 \bmod N$



y
 b
 $z = w^b u \bmod N$

A diagram showing the interaction between Alice and the King. Alice sends y to the King. The King sends b back to Alice. Alice then sends $z = w^b u \bmod N$ to the King. The King then asks if $z^2 \equiv x^b \pmod N$.

$b \leftarrow \{0,1\}$

$z^2 \equiv x^b \pmod N?$

Zero Knowledge Proofs

Known:

- Proofs for any NP statement assuming statistically-binding commitments
- Non-interactive ZK proofs for any NP statement using trapdoor permutations

Today

Proofs of knowledge

Cryptocurrencies

Crypto from minimal assumptions (if time)

Proofs of Knowledge

Sometimes, not enough to prove that statement is true, also want to prove “knowledge” of witness

Ex:

- Identification protocols: prove knowledge of key
- Discrete log: always exists, but want to prove knowledge of exponent.

Proofs of Knowledge

We won't formally define, but here's the intuition:

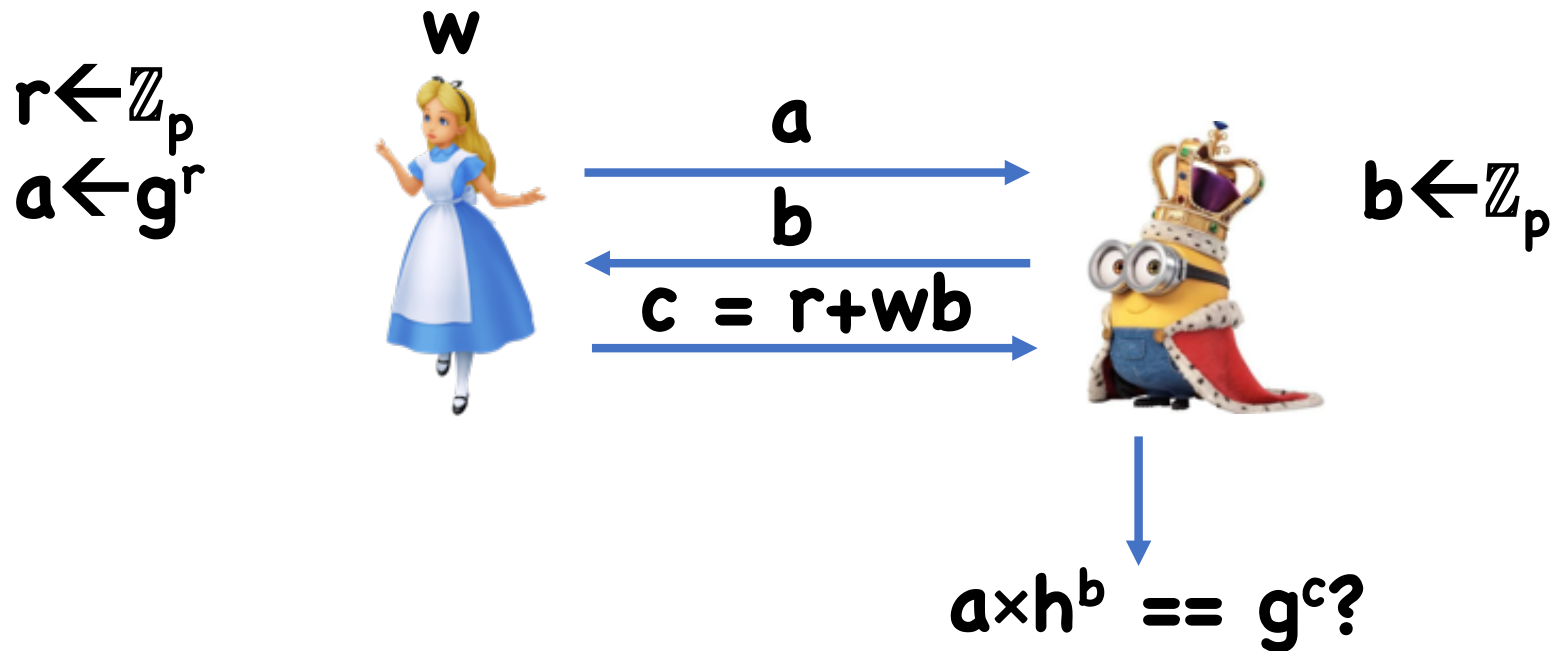
Given any (potentially malicious) PPT prover \mathbf{P}^* that causes \mathbf{V} to accept, it is possible to “extract” from \mathbf{P}^* a witness \mathbf{w}

Schnorr PoK for DLog

Statement: (g, h)

Witness: w s.t. $h = g^w$

Protocol:



Schnorr PoK for DLog

Completeness:

- $g^c = g^{r+wb} = a \times h^b$

Honest Verifier ZK:

- Transcript = (a, b, c) where $a = g^c / h^b$ and (b, c) random in \mathbb{Z}_p
- Can easily simulate. How?

Schnorr PoK for DLog

Proof of Knowledge?

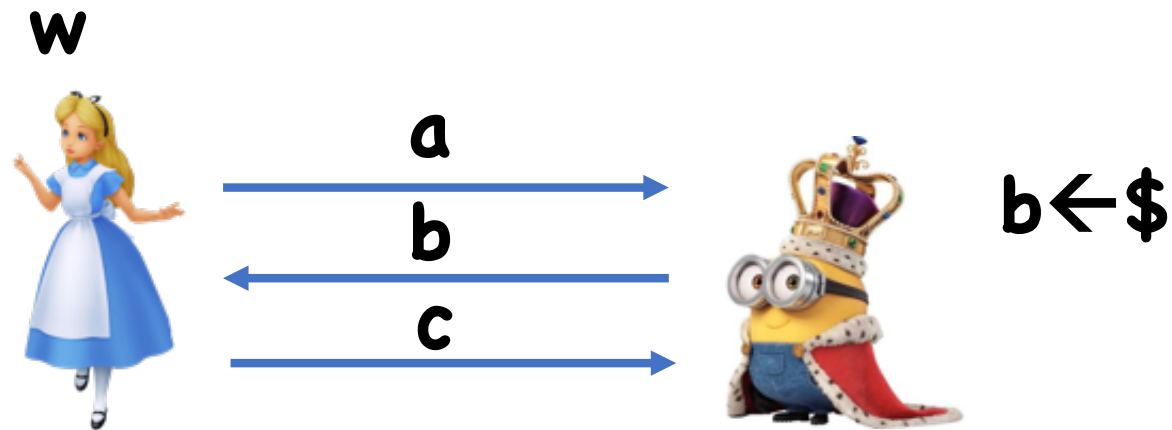
Idea: once Alice commits to $\mathbf{a}=\mathbf{g}^r$, show must be able to compute $\mathbf{c} = \mathbf{r}+\mathbf{b}\mathbf{w}$ for any \mathbf{b} of Bob's choosing

- Intuition: only way to do this is to know \mathbf{w}
- Run Alice on two challenges, obtain:

$$\mathbf{c}_0 = \mathbf{r}_0 + \mathbf{b}_0 \mathbf{w}, \mathbf{c}_1 = \mathbf{r}_1 + \mathbf{b}_1 \mathbf{w}$$

(Can solve linear equations to find \mathbf{w})

Σ Protocols



(fancy name for 3-round “public coin” protocols)

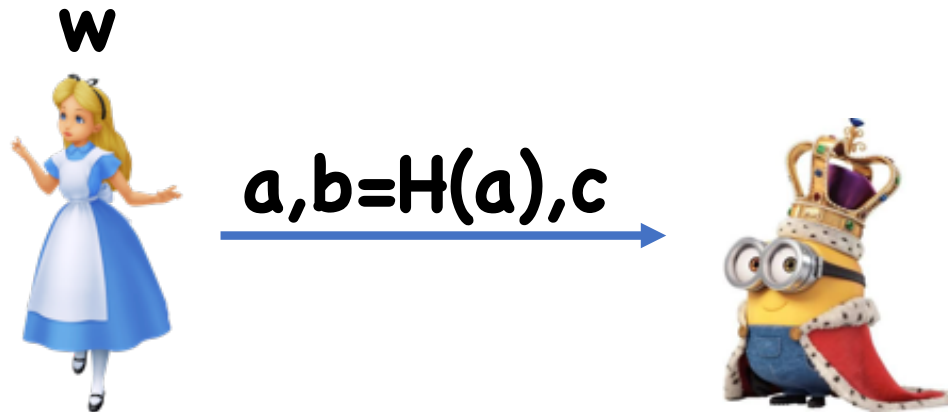
Fiat-Shamir Transform

Idea: set $\mathbf{b} = \mathbf{H}(\mathbf{a})$

- Since \mathbf{H} is a random oracle, \mathbf{a} is a random output

Notice: now prover can compute \mathbf{b} for themselves!

- No need to actually perform interaction



Theorem: If (P, V) was a secure ZKPoK for honest verifiers, and if H is a random oracle, then compiled protocol is a ZKPoK

Proof idea: second message is exactly what you'd expect in original protocol

Complication: adversary can query H to learn second message, and throw it out if she doesn't like it

Signatures from Σ Protocols

Idea: what if set $\mathbf{b} = \mathbf{H}(\mathbf{m}, \mathbf{a})$

- Challenge \mathbf{b} is message specific
- Intuition: proves that someone who knows \mathbf{sk} engaged in protocol depending on \mathbf{m}
- Can use resulting transcript as signature on \mathbf{m}

Schnorr PoK \rightarrow Schnorr Signatures

Applications of ZK (PoK)

Identification protocols: prove that you know the secret without revealing the secret

Signatures: prove that you know the secret in a “message dependent” way

Protocol Design:

- E.g. CCA secure PKE
 - To avoid mauling attacks, provide ZK proof that ciphertext is well formed
 - Problem: ZK proof might be malleable
 - With a bit more work, can be made CCA secure
- Example: multiparty computation
 - Prove that everyone behaved correctly

Cryptocurrency/Blockchain

Features of Physical Cash

Essentially anonymous

Hard to counterfeit

Easy to verify

Limitations of Physical Cash

Cannot be used online

- Instead, need to involve banks
- Banks see all transactions
- Merchants can also track you

Requires central government to issue

- Ok for most people in US, but maybe you don't trust the government

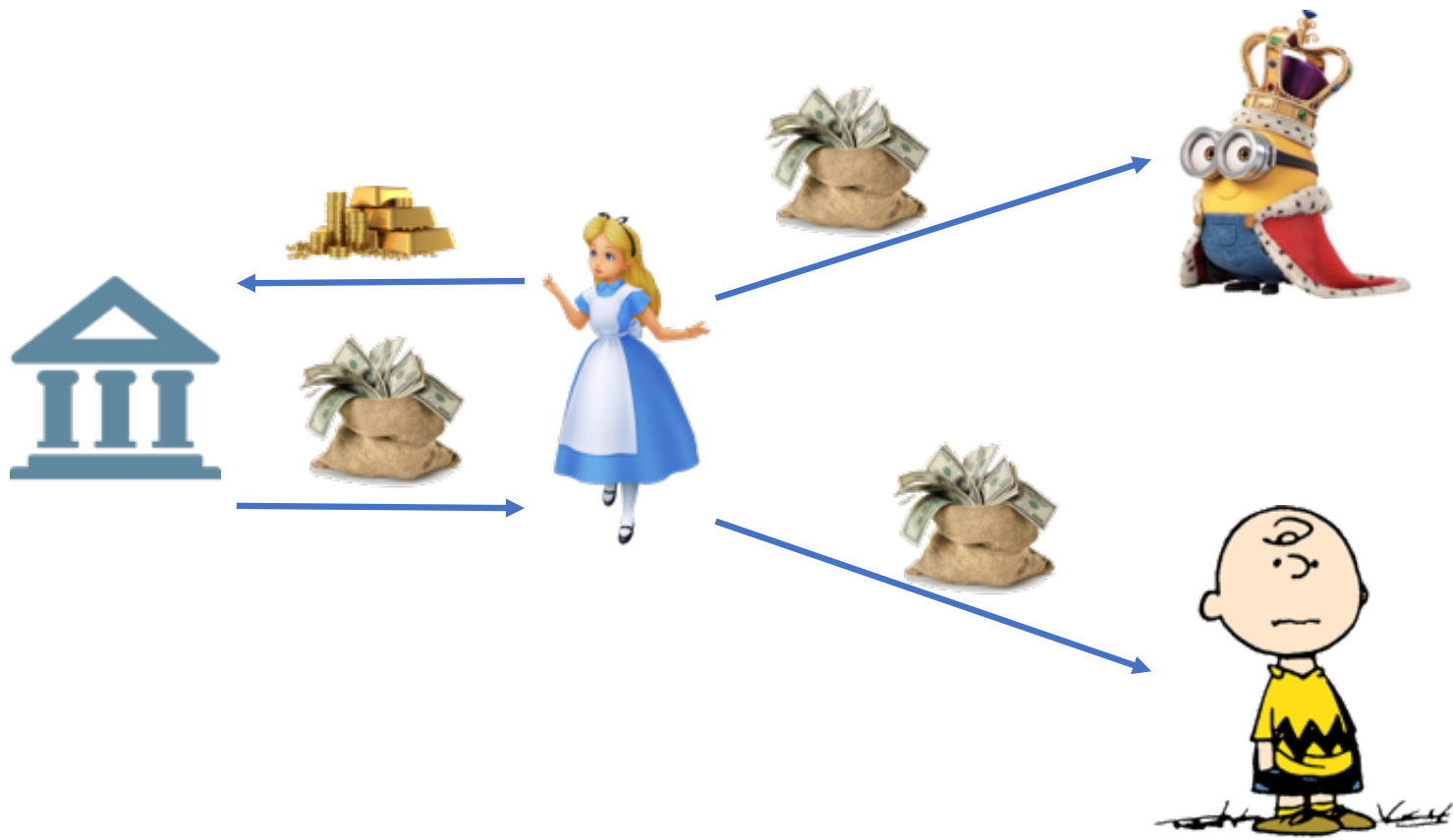
Digital Cash

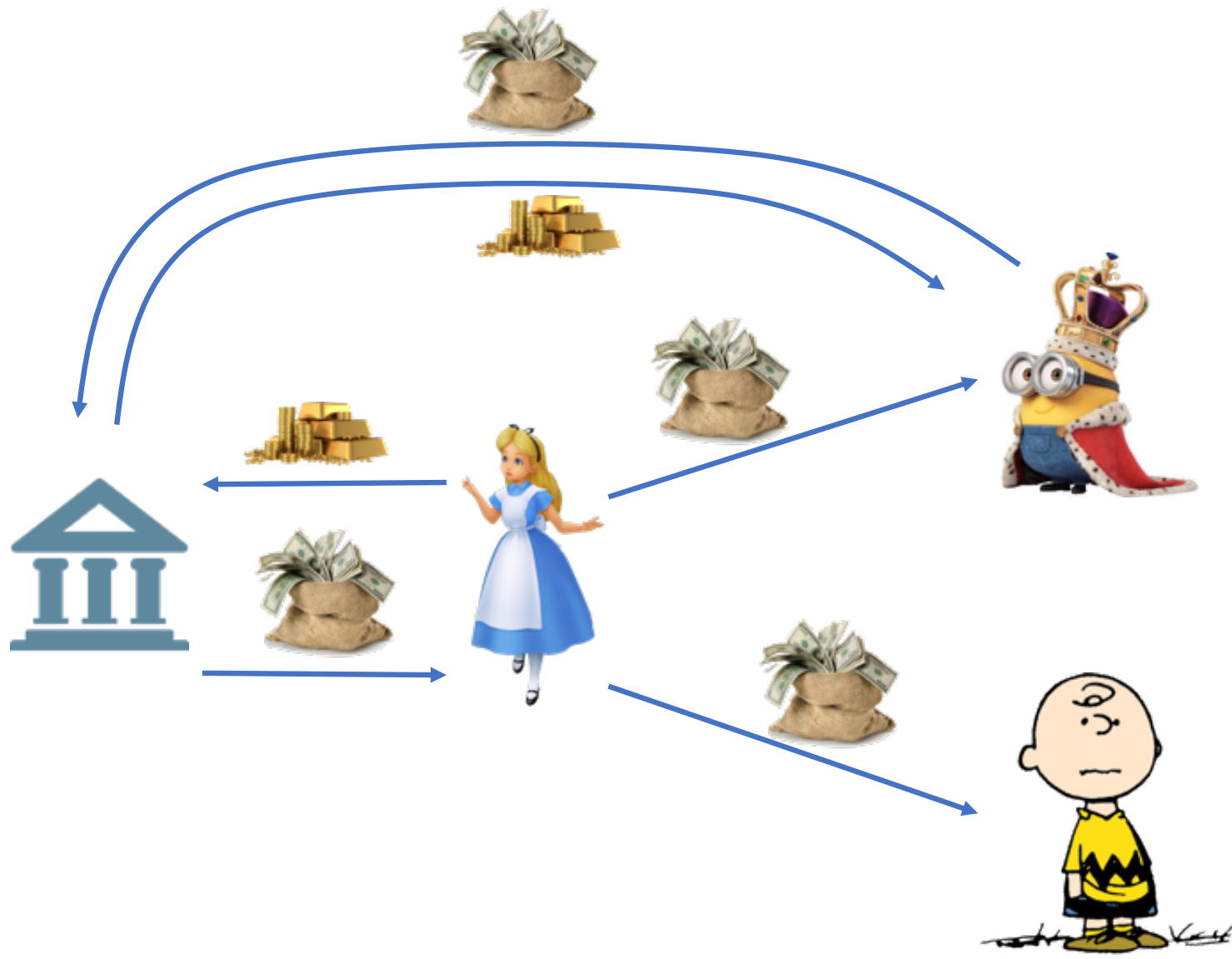
Currency is now 1s and 0s

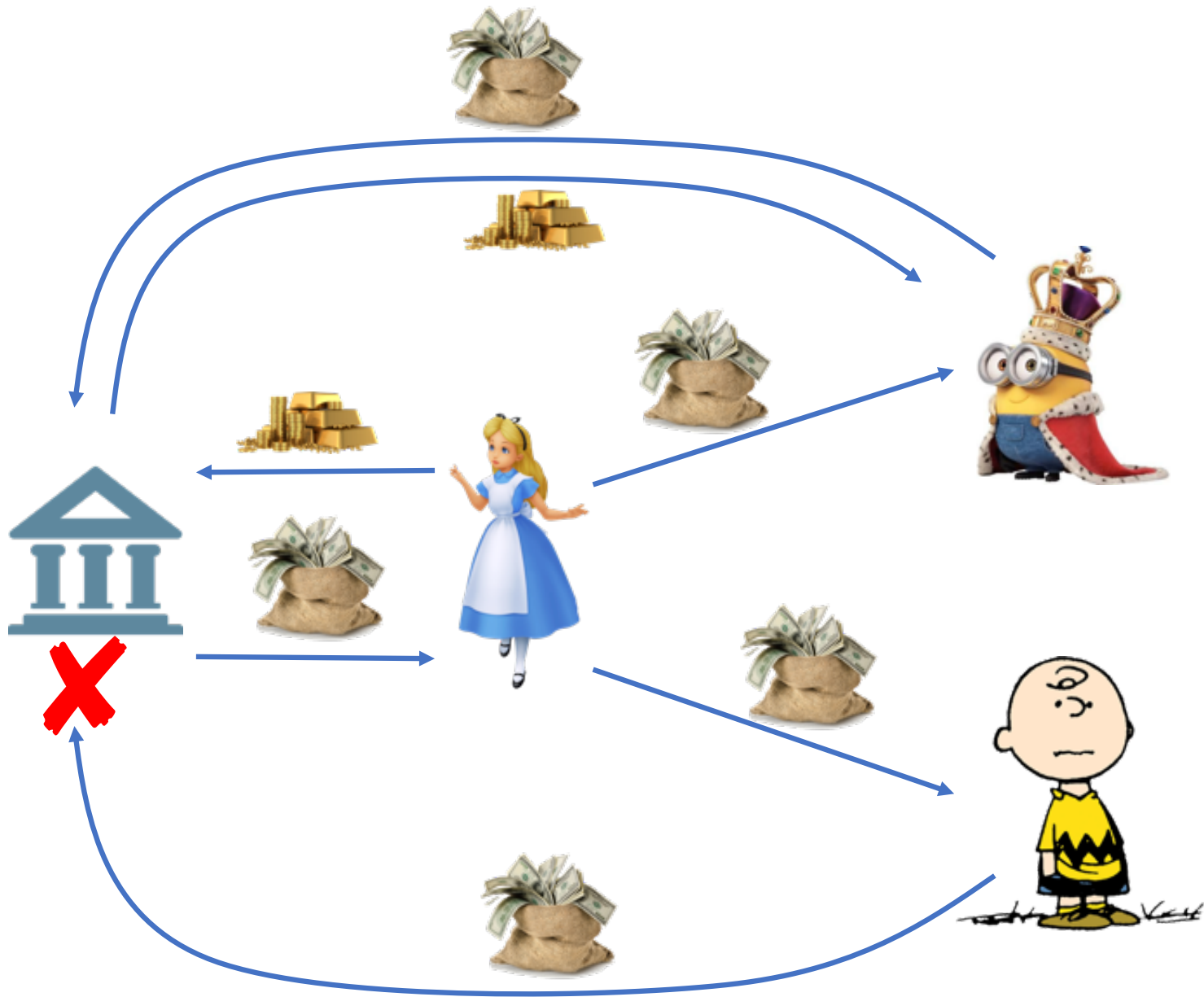
Crypto can make digital currency easy to verify, hard to mint

**Major challenge: prevent double spending
(Also decentralizing minting process)**









Solution: Public Ledger

Bank transfers \$\$ to Alice

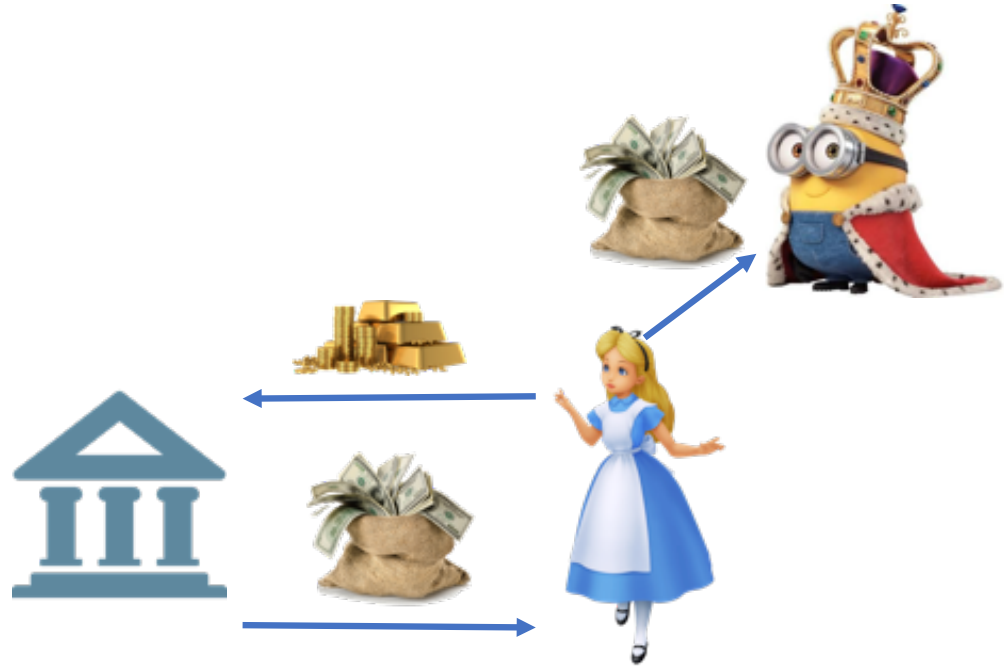
Each bill has unique serial number



Solution: Public Ledger

Bank transfers \$\$ to Alice

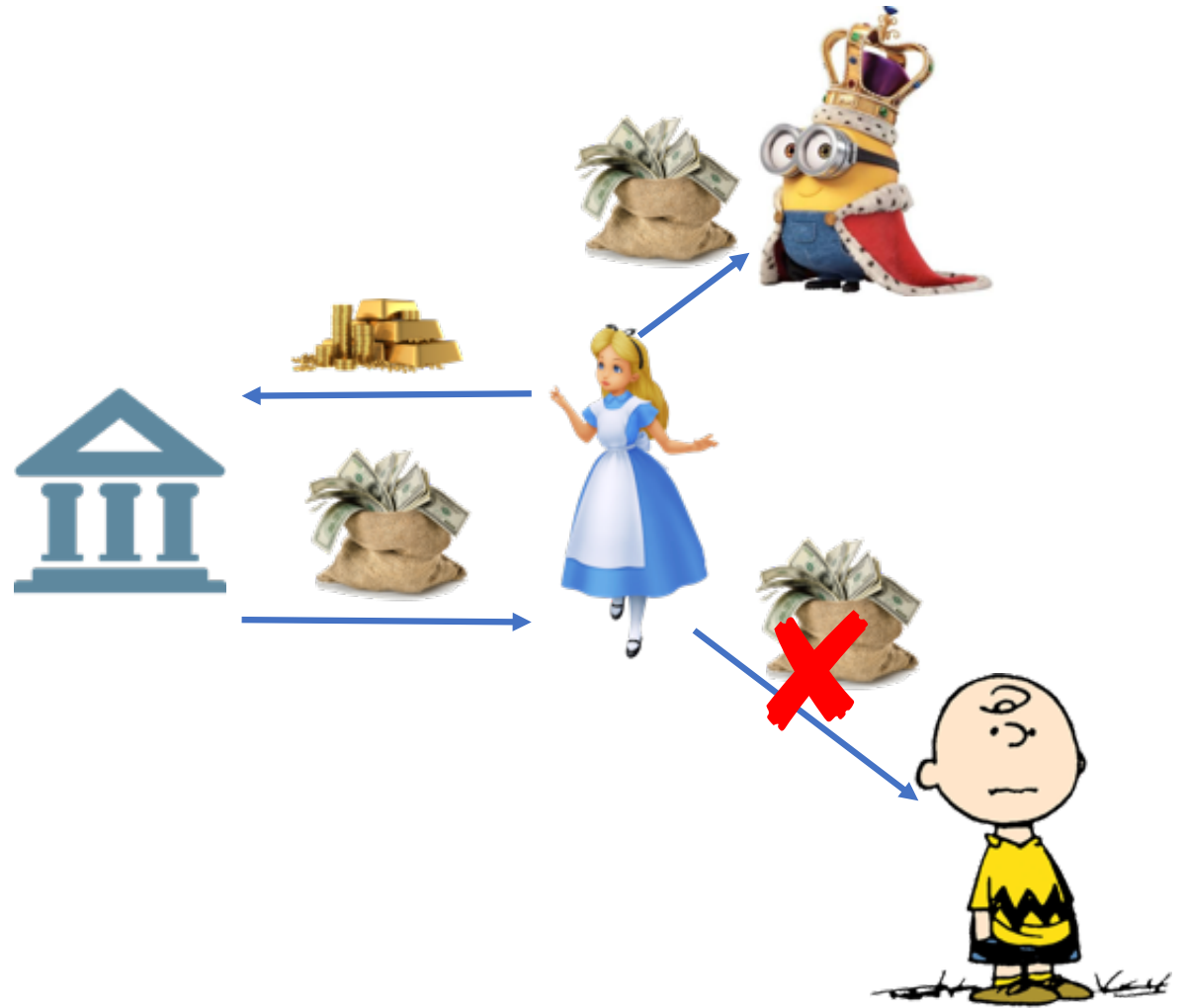
Alice transfers \$\$ to Bob



Solution: Public Ledger

Bank transfers \$\$ to Alice

Alice transfers \$\$ to Bob



Solution: Public Ledger

Bank maintain ledger?

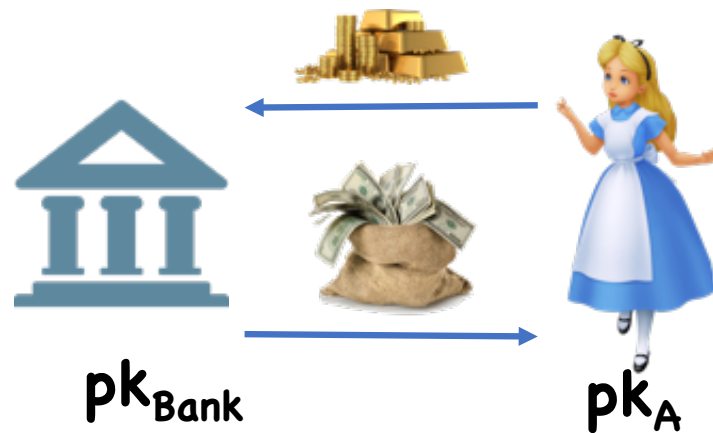
- But then bank must be involved in every transaction
- How does bank prevent malicious Bob from claiming Alice transferred money to him?

Anonymity also lost, since all transactions public

Solution: Use Signatures

pk_{Bank} transfers \$\$ to pk_A , σ_1

$\sigma_1 = \text{Sign}(sk_{\text{Bank}}, \text{"}pk_{\text{Bank}} \text{ transfers } \$\$ \text{ to } pk_A\text{"})$

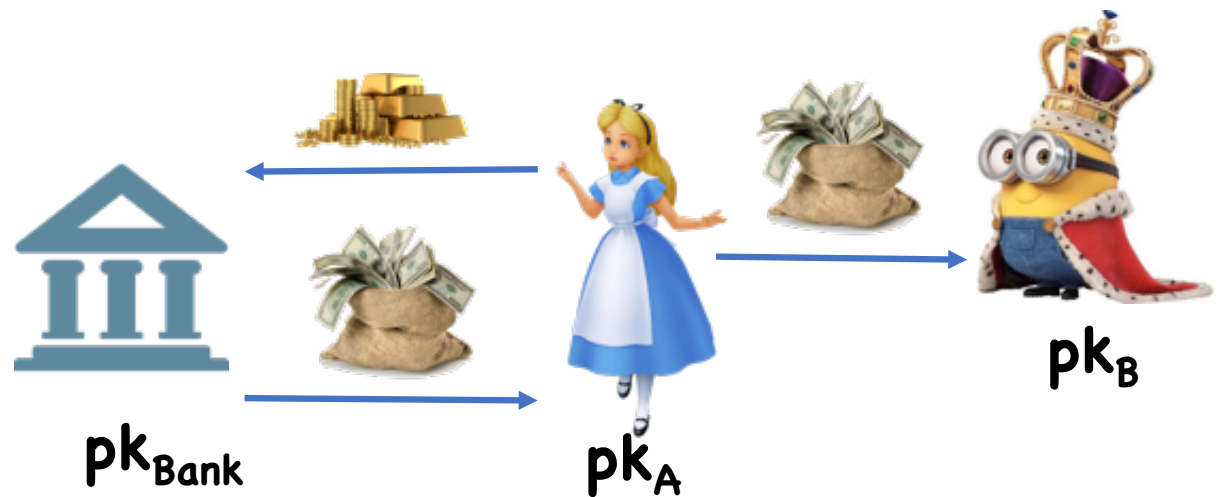


Solution: Use Signatures

pk_{Bank} transfers \$\$ to pk_A , σ_1

pk_A transfers \$\$ to pk_B , σ_2

$\sigma_2 = \text{Sign}(sk_A, \text{"pk}_A \text{ transfers \$\$ to pk}_B \text{"})$



Solution: Use Signatures

By using public key as identity, transactions not immediately traced to individual

- Though can still trace sequences of transactions

By signing, prevents Bob from claiming Alice gave him money when she didn't

Decentralized Currency

Removing the bank is hard:

- How is ledger maintained?
- How to prevent ledger from being tampered with
- Who mints new currency?
- How do we limit supply?

Proofs of Work

Prove that some amount of computation has been performed

Ex:

- Let H be a hash function (modeled as a RO)
- An input x such that $H(x) = 0^{t*****}$ is a “proof” that you computed approximately 2^t hashes

Proofs of Work and Cryptocurrency

Idea: currency is a proof of work

- Limits supply of money, so keeps inflation in check
- Now, anyone can mint new money

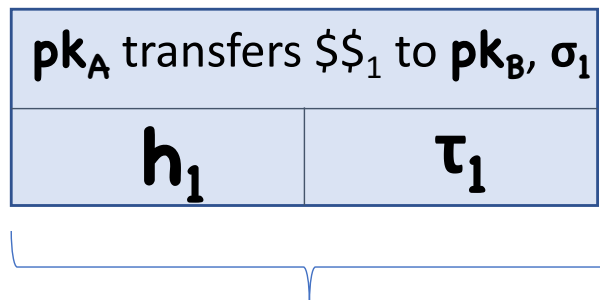
Proofs of work not the only option

- Proofs of stake
- Proofs of space

Blockchain

Immutable public ledger

Block:

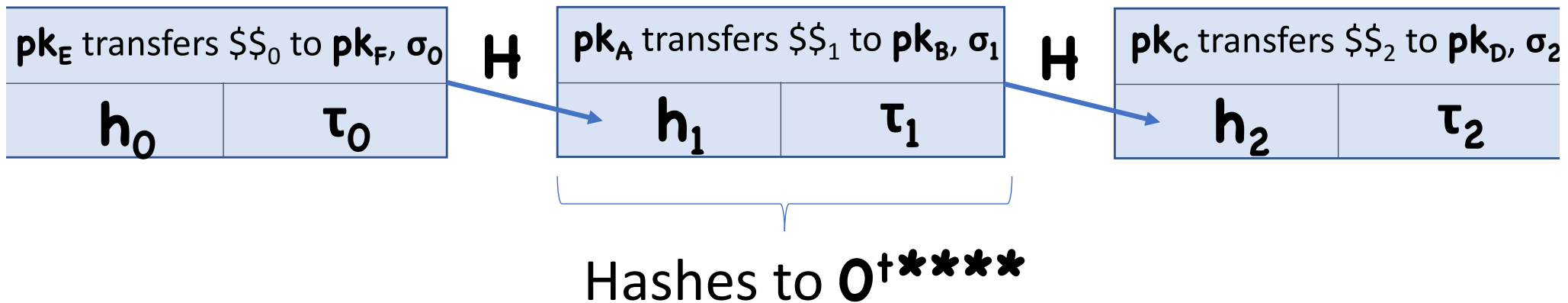


Hashes to 0^{t****}

Blockchain

Immutable public ledger

Block:



Blockchain

By making each block a proof of work, hard to modify blockchain

So proofs of work used to:

- Mint new money
- Add transactions to blockchain

Why would anyone go through the effort of adding transactions to the blockchain?

Blockchain

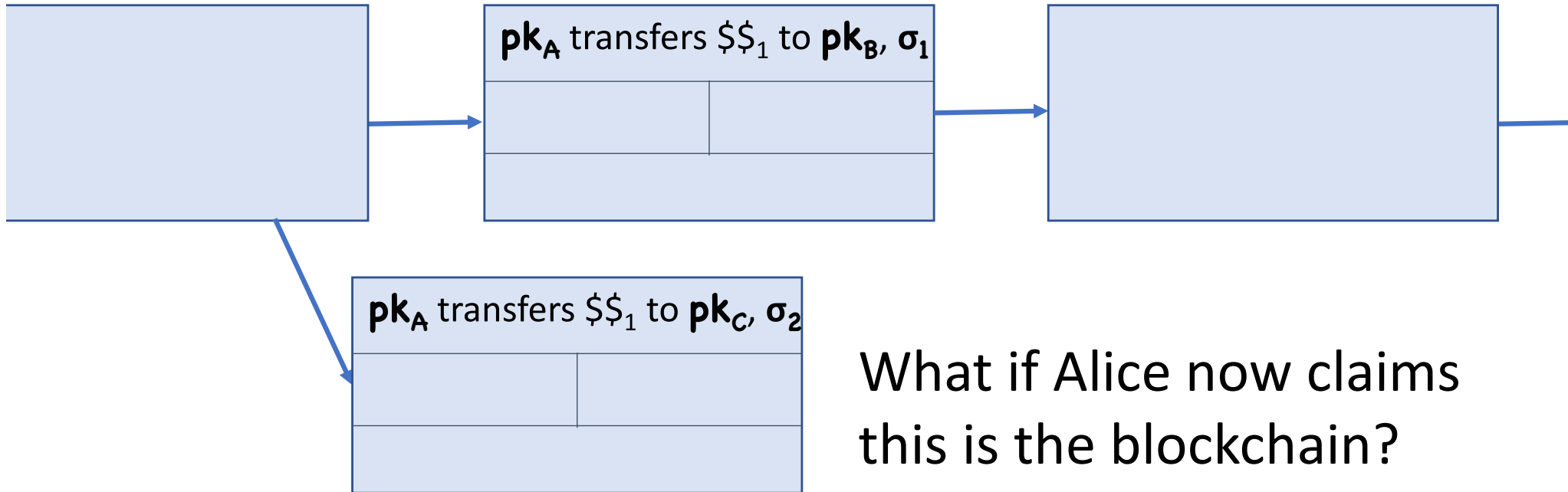
Idea: combine minting and adding blocks

Block:

pk_A transfers $\$ \$_1$ to pk_B, σ_1	
h_1	τ_1
pk_M mined $\$ \$_M$	

Hashes to 0^{*****}

Double Spending



Double Spending

To prevent double spending, everyone always uses longest chain as the blockchain

If Alice tries to double spend, she will need to create a separate chain that is as long as the main chain

- As long as she has $\ll 50\%$ of computing power of mining power, will not be possible

Crypto from Minimal Assumptions

Many ways to build crypto

We've seen many ways to build crypto

- SPN networks
- LFSR's
- Discrete Log
- Factoring

Questions:

- Can common techniques be abstracted out as theorem statements?
- Can every technique be used to build every application?

One-way Functions


The minimal assumption for crypto

Syntax:

- Domain \mathbf{D}
- Range \mathbf{R}
- Function $\mathbf{F: D \rightarrow R}$

No correctness properties other than deterministic

Security?

Definition: F is One-Way if, for all polynomial time  \exists negligible ϵ such that:


$$\Pr[x \leftarrow \text{ wizard } (F(x)), x \leftarrow D] < \epsilon$$

Trivial example:

$F(x)$ = parity of x

Given $F(x)$, impossible to predict x

Security

Definition: F is One-Way if, for all polynomial time  \exists negligible ϵ such that:

$$\Pr[F(x)=F(y):y\leftarrow_{\text{img alt="wizard icon" data-bbox="476 466 501 552"} (F(x)),x\leftarrow D] < \epsilon$$

Examples

Any PRG

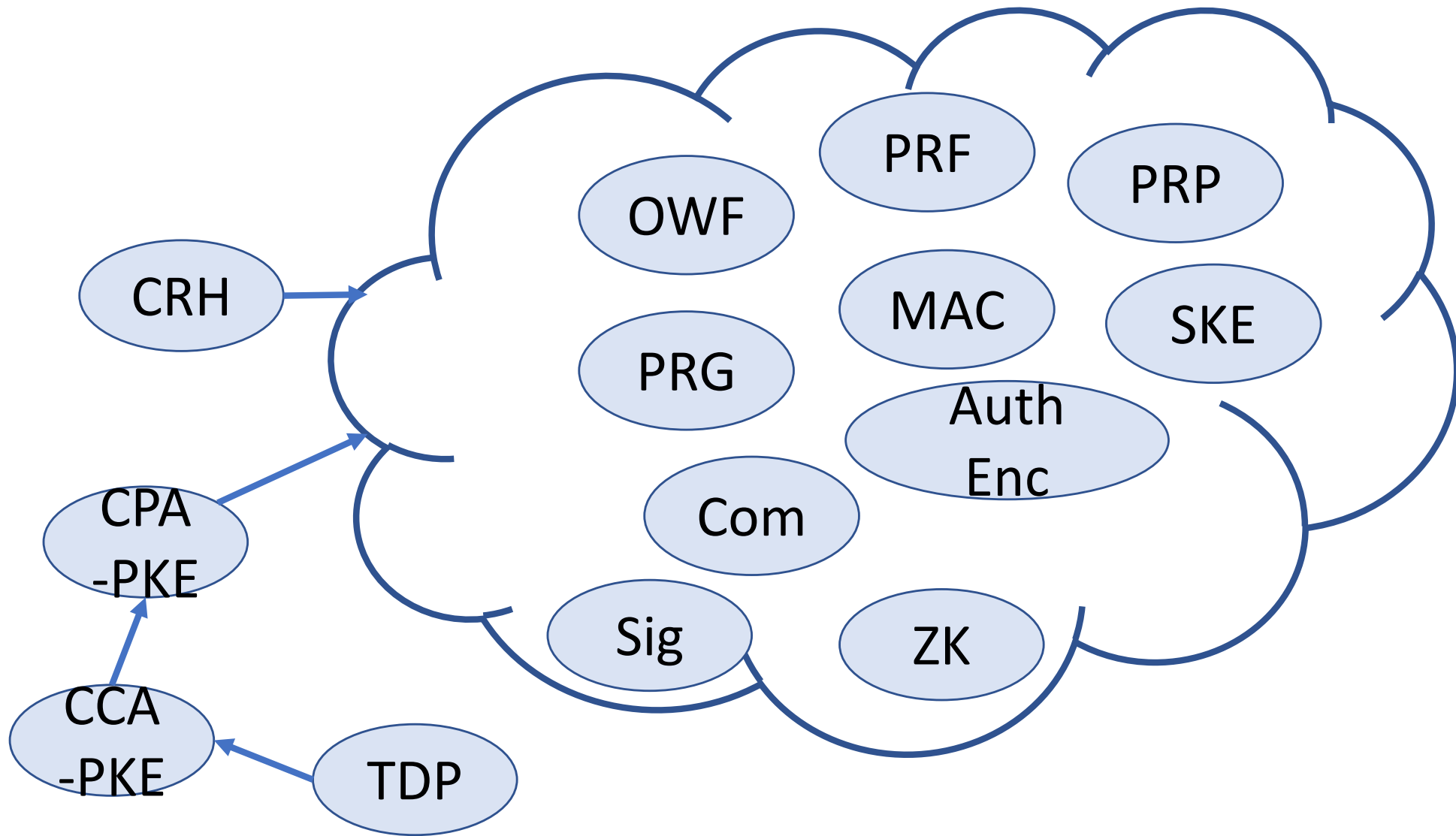
Any Collision Resistant Hash Function (with sufficient compression)

$$F(p,q) = pq$$

$$F(g,a) = (g, g^a)$$

$$F(N,x) = (N, x^3 \bmod N) \text{ or } F(N,x) = (N, x^2 \bmod N)$$

What's Known



Generally Believed That...

Cannot construct PKE from OWF

Cannot construct CRH from OWF

Cannot construct PKE from CRH

Cannot construct CRH from PKE

Black Box Separations

How do we argue that you cannot build PKE from one-way functions?

- We generally believe both exist!

Observation: most natural constructions treat underlying objects as black boxes (don't look at code, just input/output)

Maybe we can rule out such natural constructions

Black Box Separations

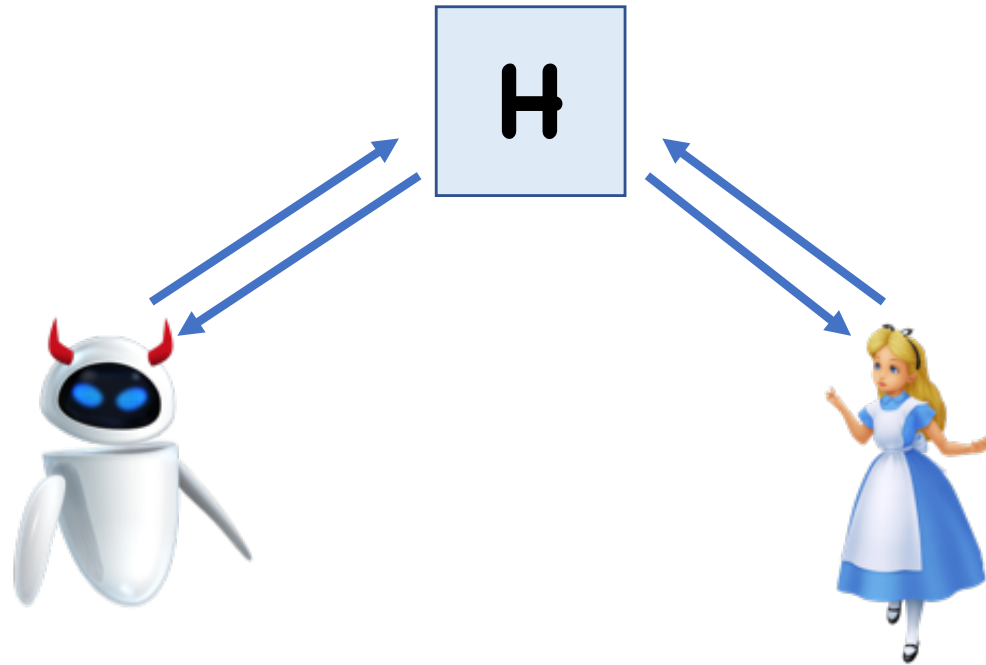
Present a world where one-way functions exist, but PKE does not

Hopefully, natural (black box) constructions make sense in this world

- Can construct PRGs, PRFs, PRPs, Auth-Enc, etc

Separating PKE from OWF

Random oracle model:



Computation power is unlimited, but number of calls to random oracle is polynomial

Separating PKE from OWF

In ROM, despite unlimited computational power, one-way functions exist

- **$F(x) = H(x)$**
- Can only invert oracle by making exponentially-many calls

Possible to show PKE does NOT exist in ROM

- In fact, not even public key distribution exists
- Idea: adversary can use unlimited computational power to narrow down search to just a few secret keys without making any oracle queries

Black Box Separations

Of course, our pretend world isn't real

However, it shows a barrier for commonly used proof techniques

- Similar to “relativization” for complexity theory

Non-black box techniques are known and used, but relatively rare

Announcements

HW7 Due April 30th

Project 3/HW 8 due May 12th