

COS433/Math 473: Cryptography

Mark Zhandry

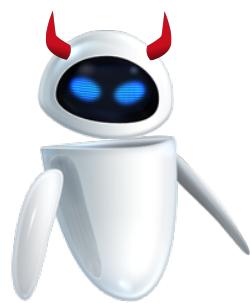
Princeton University

Spring 2018

Identification

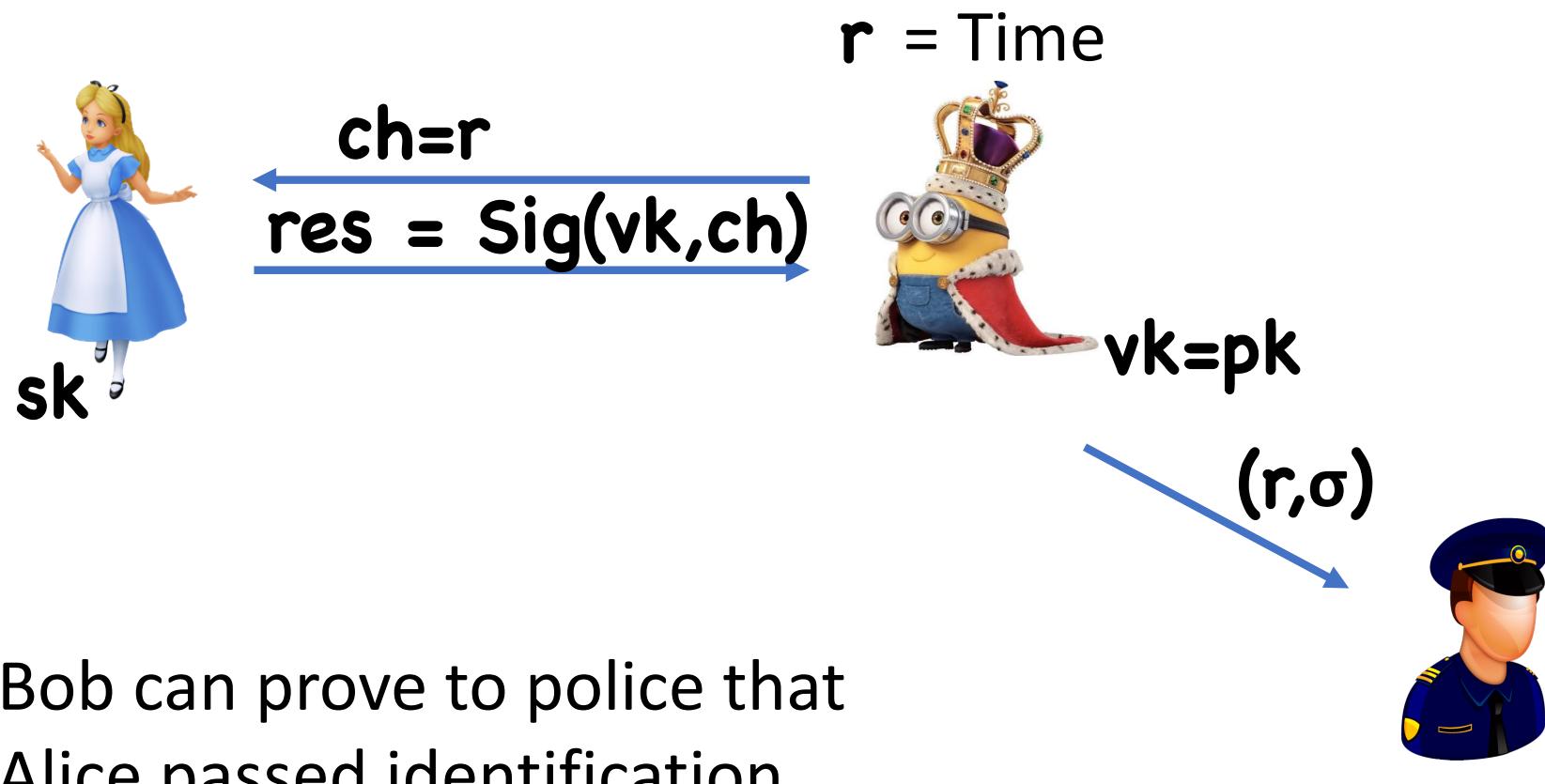


Identification



Non-Repudiation

Consider signature-based C-R



Zero Knowledge

What if Bob could have come up with a valid transcript, without ever interacting with Alice?

- Then Bob cannot prove to police that Alice authenticated

Seems impossible:

- If (public) **vk** is sufficient to come up with valid transcript, why can't an adversary do the same?

Zero Knowledge

Adversary CAN come up with valid transcripts, but Bob doesn't accept transcripts

- Instead, accepts *interactions*

Ex: public key Enc-based C-R

- Valid transcript: (\mathbf{c}, \mathbf{r}) where \mathbf{c} encrypts \mathbf{r}
- Anyone can come up with a valid transcript
- However, only Alice can generate the transcript for a given \mathbf{c} chosen by Bob

Takeaway: order matters

Today

Zero knowledge proofs

- Prove a theorem without revealing how to prove it

Mathematical Proof



π



π



$\text{Ver}(\pi)$

Mathematical Proof

Statement x

Witness w



$\text{Ver}(x,w)$

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

Intuition: prover doesn't learn anything by engaging in the protocol (other than the truthfulness of x)

How to characterize what adversary “knows”?

- Only outputs a bit
- May “know” witness, but hidden inside the programs state

Zero Knowledge

First Attempt:

\exists “simulator”  s.t. for every true statement x ,
valid witness w ,

$$\text{lion}(x) \approx_c P(x, w) \xrightarrow{\quad} V(x)$$

Zero Knowledge

First Attempt:

Assumes Bob obeys protocol

- “Honest Verifier”

But what if Bob deviates from specified prover algorithm to try and learn more about the witness?

Zero Knowledge

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



$$\text{lion}(x) \approx_c P(x, w) \xrightarrow{\quad} V^*(x)$$

$\xleftarrow{\quad}$ $\xleftarrow{\quad}$

QR Protocol

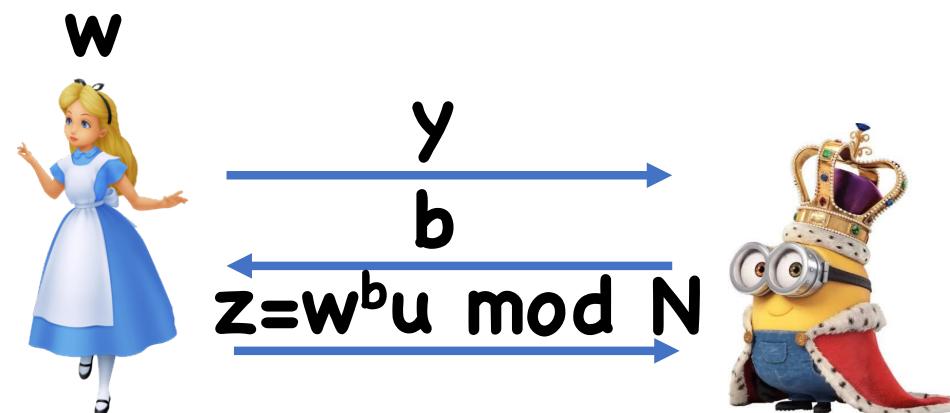
Statements: x is a Q.R. mod N

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

Protocol:

$$u \leftarrow Z_N^*$$

$$y \leftarrow u^2 \bmod N$$



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

$$z^2 == x^b y \bmod N?$$

QR Protocol

Completeness:

- $z^2 = (w^b u)^2 = (w^2)^b u^2 = x^b y$

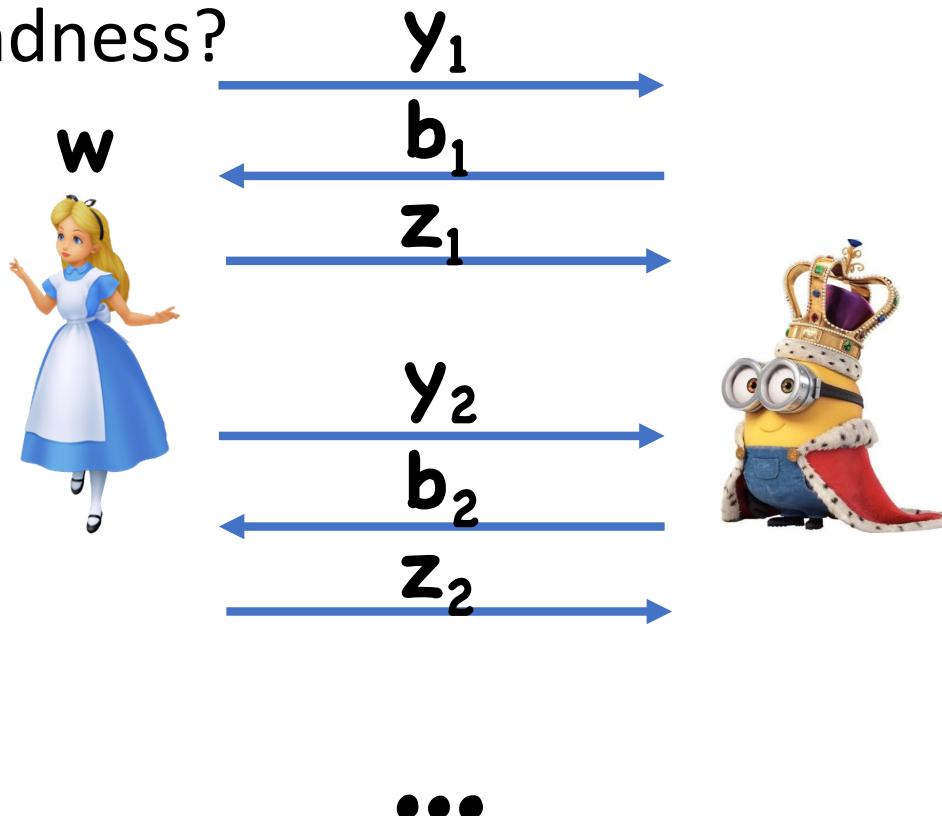
Soundness:

- Suppose x is not a QR
- Consider malicious prover P^*
- No matter what y is, either
 - y is not a QR, or
 - xy is not a QR
- With prob. $1/2$, P^* will have to find a non-existent root

QR Protocol

Boosting Soundness?

Repetition:

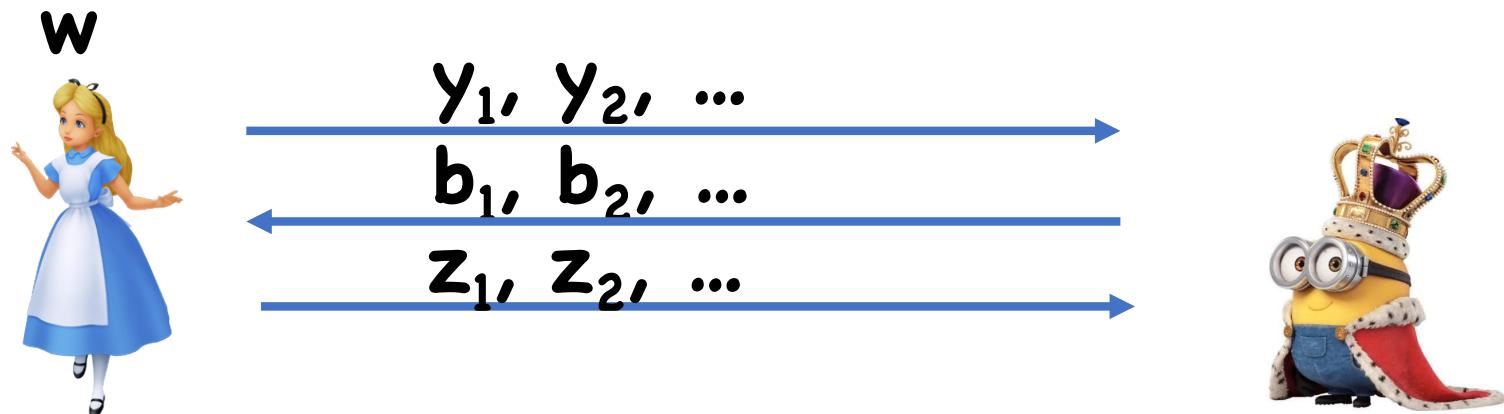


Theorem: If (P, V) has soundness error $\frac{1}{2}$, then repeating t times gives soundness error 2^{-t}

QR Protocol

Boosting Soundness?

Parallel Repetition:



Theorem: If (P, V) has soundness error $\frac{1}{2}$, then repeating t times in parallel gives soundness error 2^{-t}

QR Protocol

Zero Knowledge:

What does Bob see?

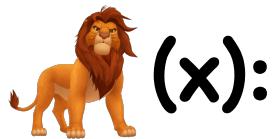
- A random QR y ,
- A random bit b ,
- A random root of $x^b y$

Idea: simulator knows b when generating y ,

- Can choose y s.t. it always knows a square root of $x^b y$

QR Protocol

Honest Verifier Zero Knowledge:



(x):

- Choose a random bit b
- Choose a random string z
- Let $y = x^{-b} z^2$
- Output (y, b, z)

- If x is a QR, then y is a random QR, no matter what b is
- z is a square root of $x^b y$



(y, b, z) is distributed identically to $(P, V)(x)$

QR Protocol

(Malicious Verifier) Zero Knowledge?

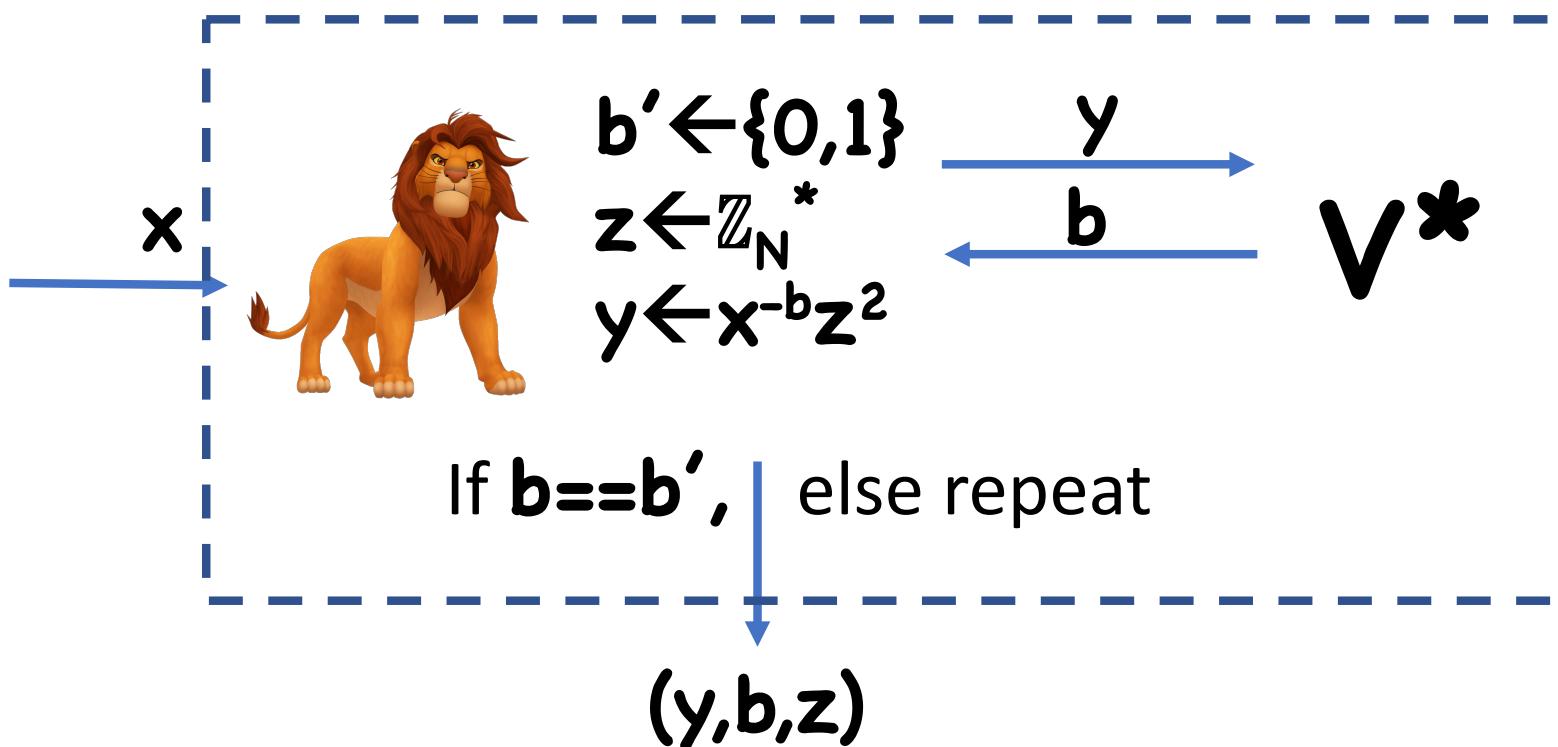
Problem: **b** might not be random

Worse yet, **b** may depend on **y**

- But simulator generates **b** first, then **y**

QR Protocol

(Malicious Verifier) Zero Knowledge:



QR Protocol

(Malicious Verifier) Zero Knowledge:

Proof:

- If x is a QR, then y is a random QR, independent of b'
- Conditioned on $b' = b$, then (y, b, z) is identical to random transcript seen by V^*
- $b' = b$ with probability $1/2$

Repetition and Zero Knowledge

(sequential) repetition also preserves ZK

Unfortunately, parallel repetition might not:

-  makes guesses $\mathbf{b}_1', \mathbf{b}_2', \dots$
- Generates valid transcript only if all guesses were correct
- Probability of correct guess: 2^{-t}

Maybe other simulators will work?

- Known to be impossible in general, but nothing known for QR

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 P^* that causes V to accept, it is possible to “extract” from P^* a witness w

Schnorr PoK for DLog

Statement: (g, h)

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

Protocol:

$$\begin{aligned} r &\leftarrow \mathbb{Z}_p \\ a &\leftarrow g^r \end{aligned}$$



w

$$\begin{array}{c} a \\ \xrightarrow{\hspace{2cm}} \\ b \\ \xleftarrow{\hspace{2cm}} \\ c = r + wb \end{array}$$



$$b \leftarrow \mathbb{Z}_p$$

$$a \times h^b == g^c?$$

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?

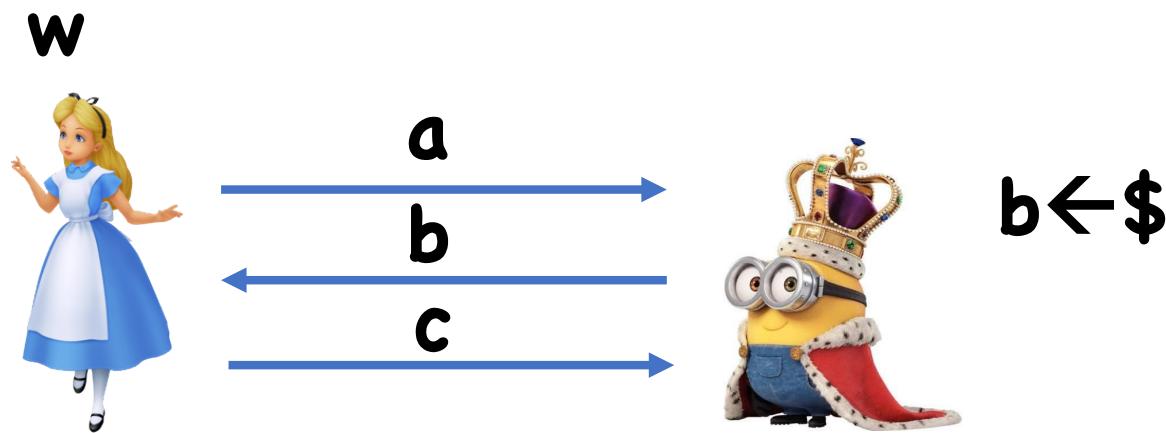
Idea: once Alice commits to $a=g^r$, show must be able to compute $c = r+bw$ for any b of Bob's choosing

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

$$c_0 = r_0 + b_0 w, c_1 = r_1 + b_1 w$$

(Can solve linear equations to find w)

Σ Protocols



Identification from Σ Protocols

pk = some hard statement (e.g. (g, h))

sk = witness (e.g. Dlog)

To identify, just engage in ZKPoK that you know witness

- Zero knowledge means prover learns nothing from interaction
- PoK means you'll only be let in if you indeed know witness

If ZKPoK is only ZK for honest verifiers, more work needed to get active security

Deniability

Zero Knowledge proofs provide deniability:

- Alice proves statement **x** is true to Bob
- Bob goes to Charlie, and tries to prove **x** by providing transcript
- Charlie not convinced, as Bob could have generated transcript himself
- Alice can later deny that she knows proof of **x**

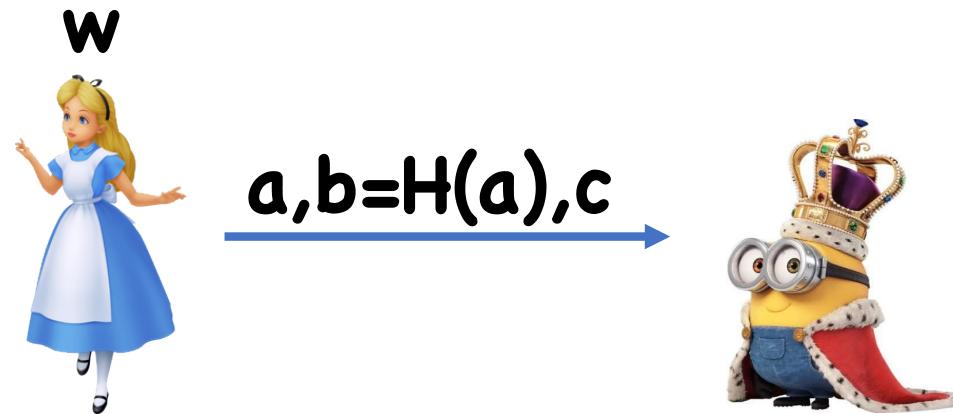
Fiat-Shamir Transform

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

- Since 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, then the random oracle protocol is a ZKPoK in the random oracle model

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

Non-Interactive Zero Knowledge (NIZK)

Claim: ZK proof consisting of just one message from prover to verifier is impossible (Why?)

Why doesn't this contradict statement on previous slide?

Other variation: NIZK with common reference string

Observation: NIZKs loose deniability

Signatures from Σ Protocols

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

- 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 Signatures

$\text{sk} = w$

$\text{pk} = h := g^w$

$\text{Sign}(\text{sk}, m)$:

- $r \leftarrow \mathbb{Z}_p$
- $a \leftarrow g^r$
- $b \leftarrow H(m, a)$
- $c \leftarrow r + wb$
- Output (a, c)

$\text{Ver}(h, m, (a, c))$:

- $b \leftarrow H(m, a)$
- $a \times h^b == g^c ?$

Zero Knowledge Proofs

Known:

- Proofs for any NP statement assuming just one-way functions
- Non-interactive ZK proofs for any NP statement using trapdoor permutations

Applications

Identification protocols

Signatures

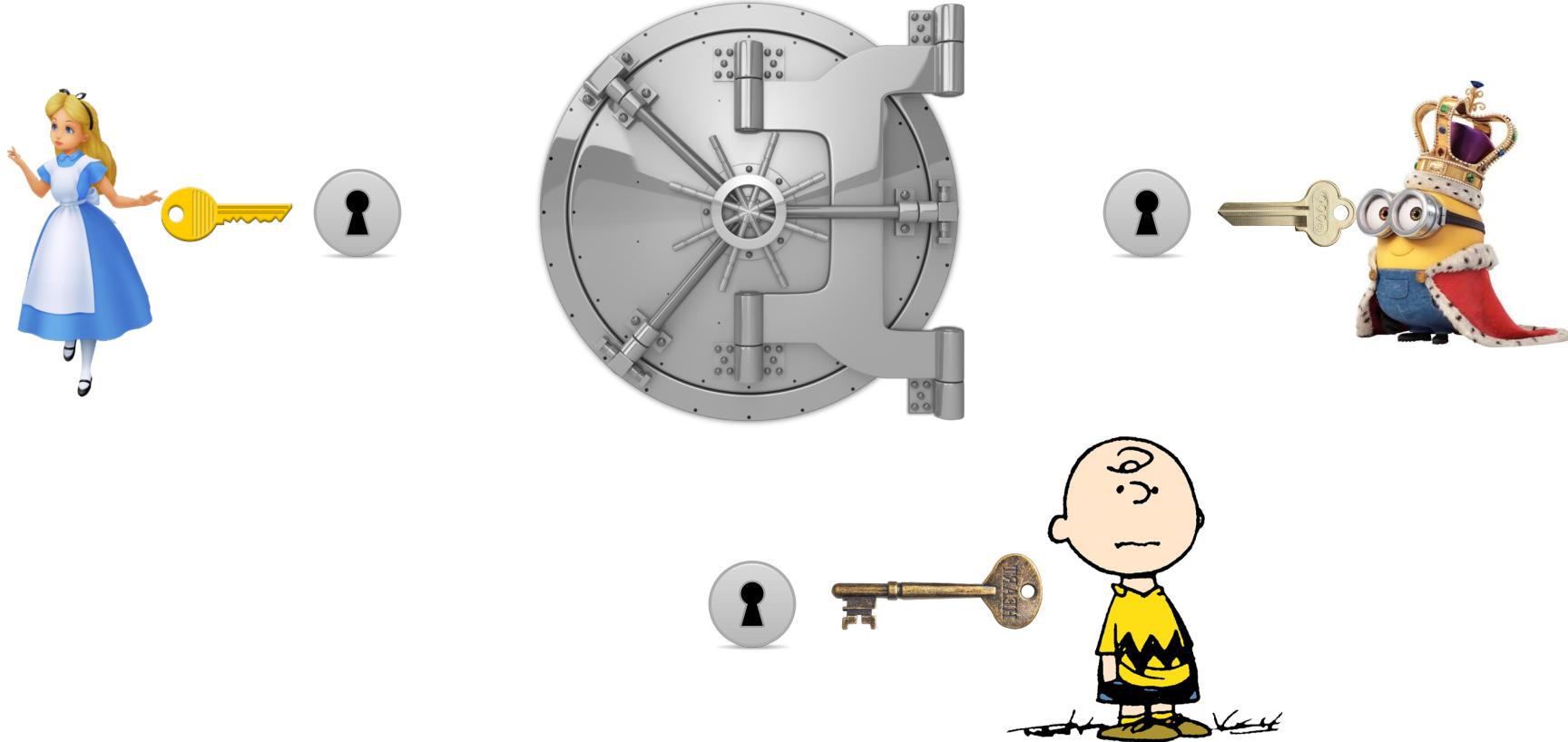
Protocol Design:

- 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

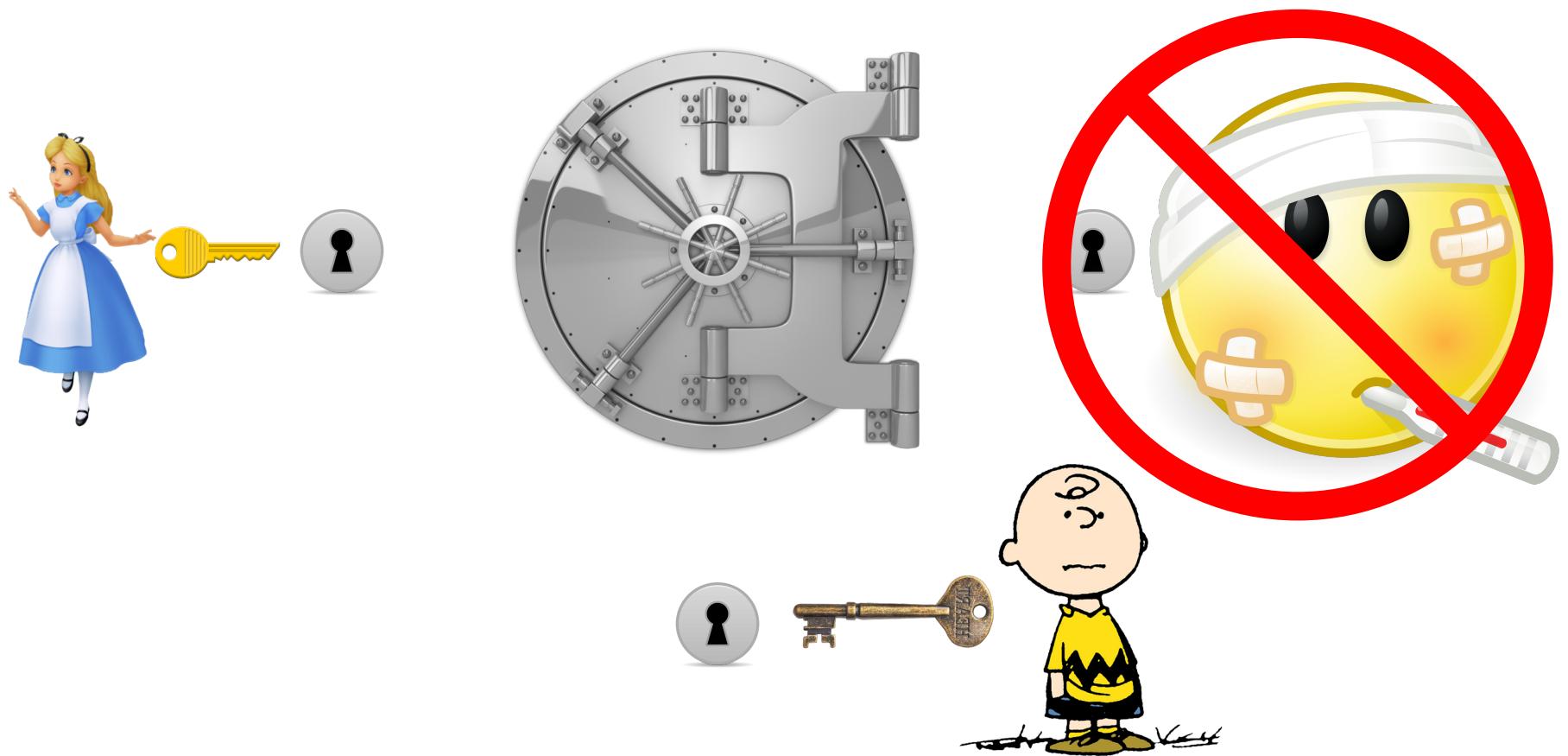
Secret Sharing



Vault should only open if both Alice and Bob are present



Vault should only open if Alice, Bob, and Charlie are all present



Vault should only open if any two of
Alice, Bob, and Charlie are present

(Threshold) Secret Sharing

Syntax:

Share(k, t, n) outputs (sh_1, \dots, sh_n)
Recon($(sh_i)_{i \in S}$) outputs k'

Correctness: $\forall S$ s.t. $|S| \geq t$

If $(sh_i)_{i=1, \dots, n} \leftarrow Share(k, t, n)$, then

$\Pr[\text{Recon}((sh_i)_{i \in S}) = k] = 1$

(Threshold) Secret Sharing

Security:

For any S , $|S| < t$, given $(sh_i)_{i \in S}$, should be impossible to recover k

$(sh_i)_{i \in S} : (sh_i)_{i=1, \dots, n} \leftarrow \text{Share}(k_0, t, n)$

\approx

$(sh_i)_{i \in S} : (sh_i)_{i=1, \dots, n} \leftarrow \text{Share}(k_1, t, n)$

n-out-of-**n** Secret Sharing

Share secret **k** so that only can only reconstruct **k** if all **n** users get together

Ideas?

Shamir Secret Sharing

Let p be a prime $> n, \geq \#(k)$

Share(k, t, n):

- Choose a random polynomial P of degree $t-1$ where $P(0) = k$
- $sh_i = P(i)$

Recon($(sh_i)_{i \in S}$): use shares to interpolate P , then evaluate on 0

Shamir Secret Sharing

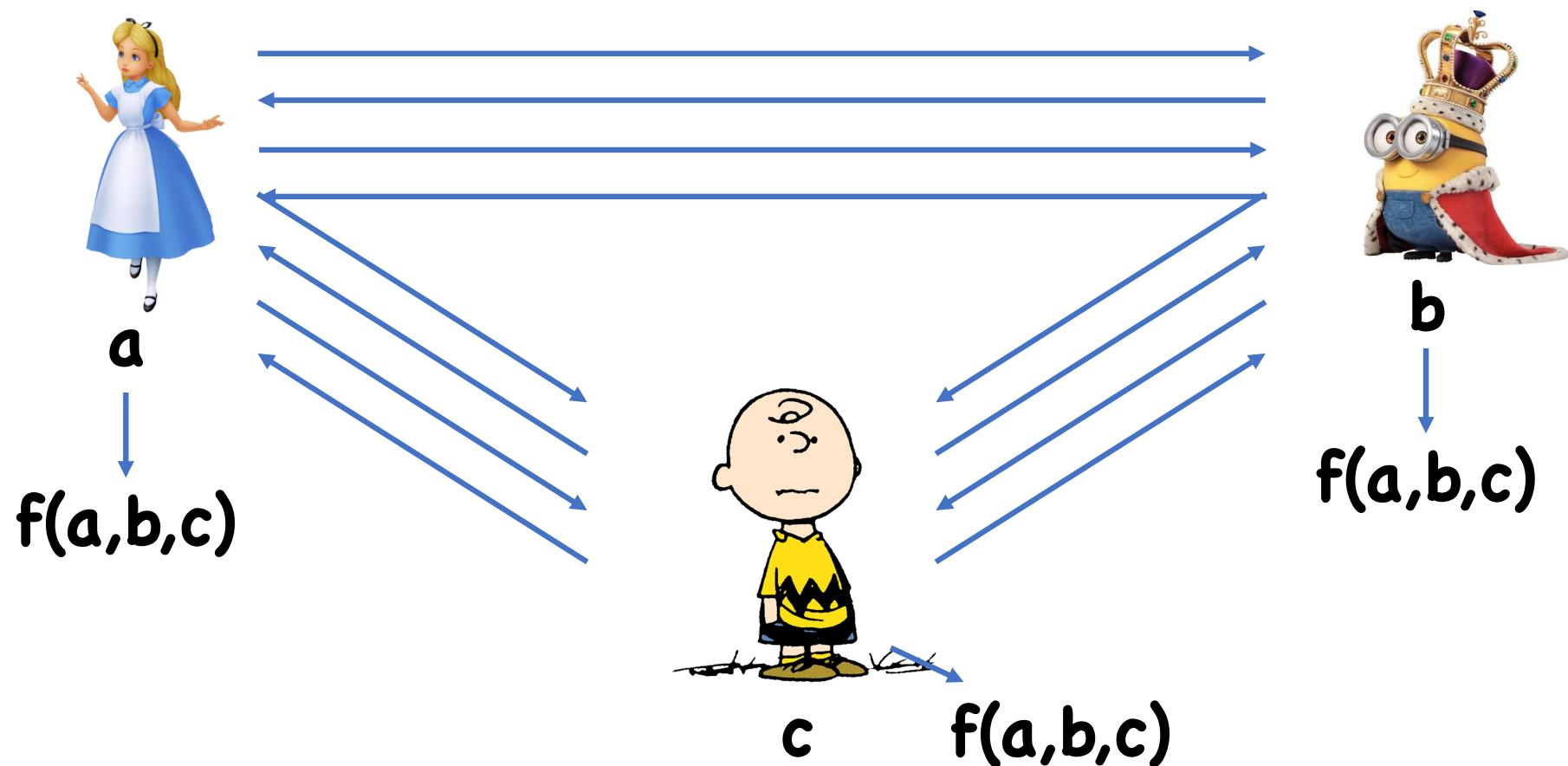
Correctness:

- t input/outputs (shares) are enough to interpolate a degree $t-1$ polynomial

Security:

- Given just $t-1$ inputs/outputs, $P(0)$ is equally likely to be any value

Multiparty Computation



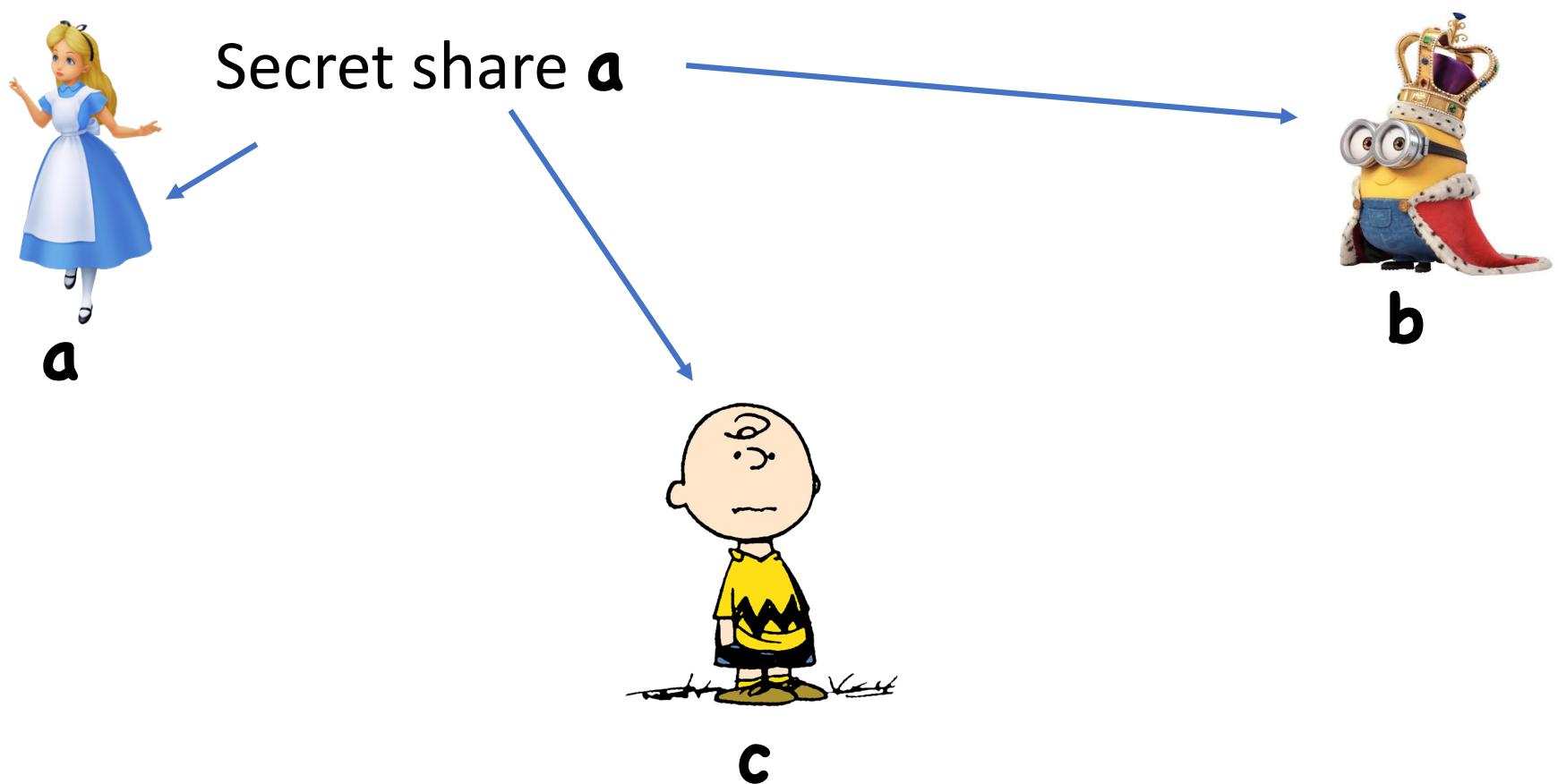
Multiparty Computation

Observation 1: Shamir secret sharing is additively homomorphic:

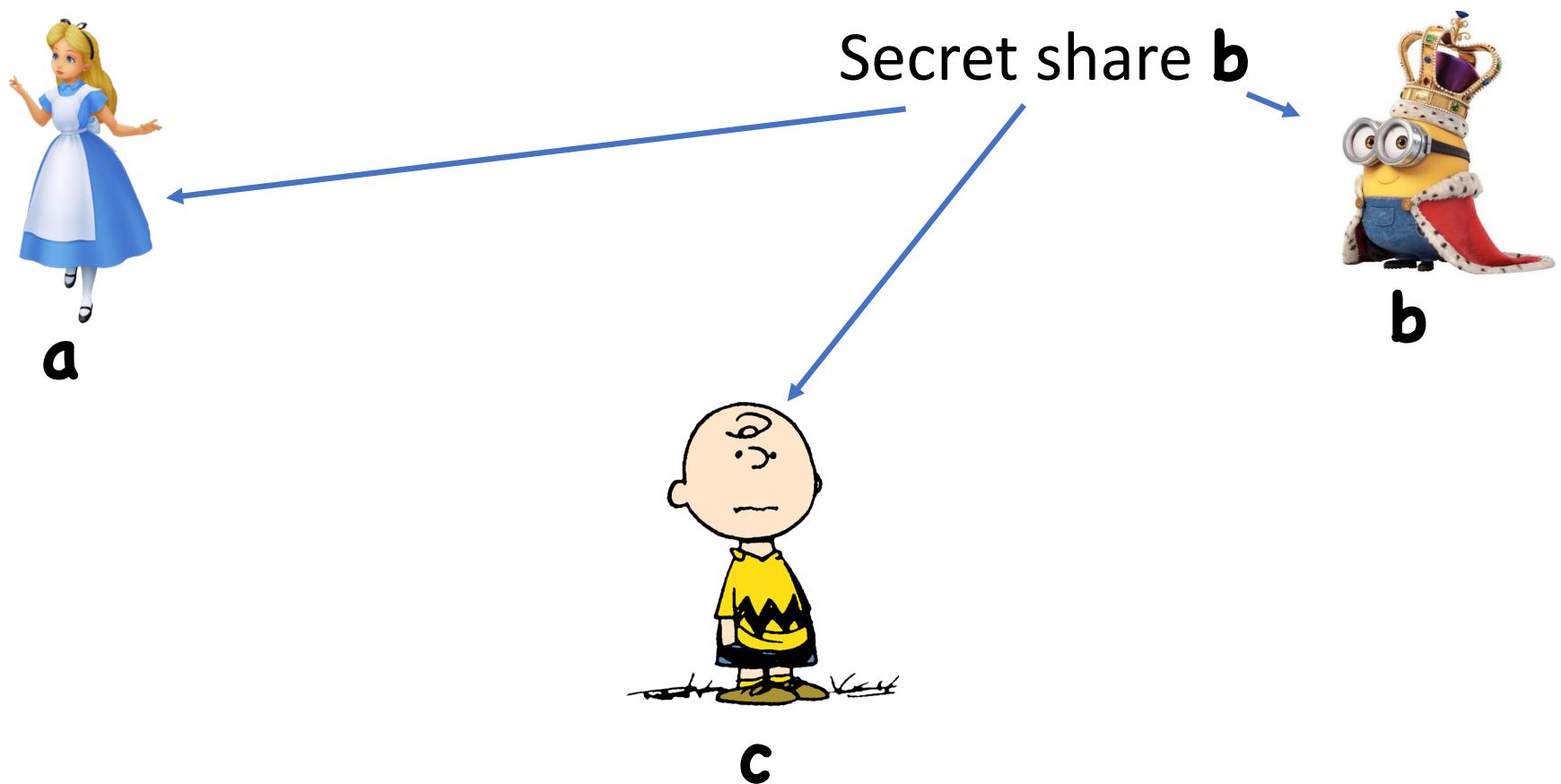
Given shares \mathbf{sh}_1 of x_1 and \mathbf{sh}_2 of x_2 , $r \times \mathbf{sh}_1 + s \times \mathbf{sh}_2$ is a share of $r \times x_1 + s \times x_2$

- $\mathbf{sh}_1 = P_1(i)$, $\mathbf{sh}_2 = P_2(i)$, so
$$r \times \mathbf{sh}_1 + s \times \mathbf{sh}_2 = (r \times P_1 + s \times P_2)(i)$$
- $r \times P_1 + s \times P_2$ has same degree

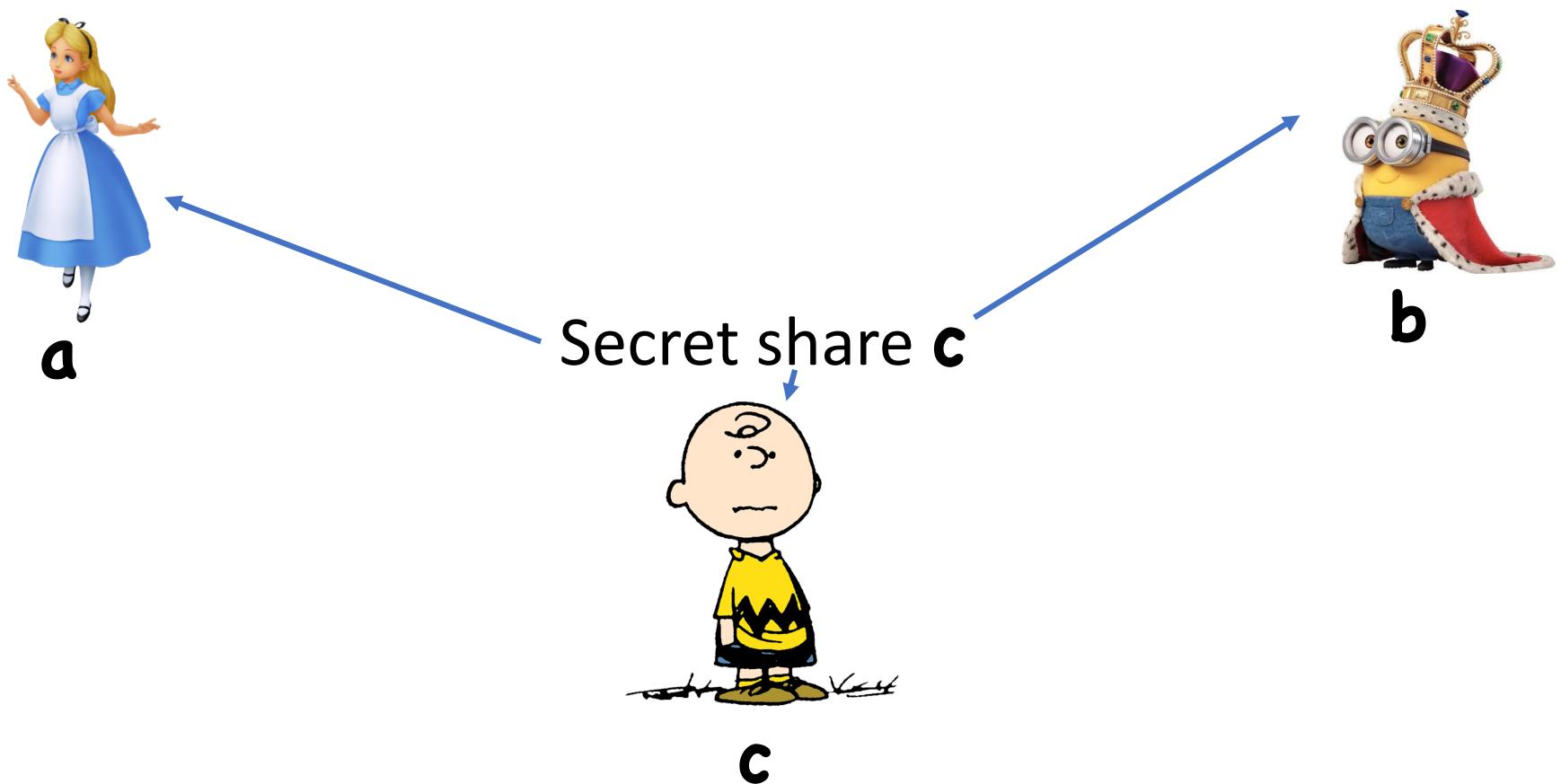
MPC for linear f



MPC for linear f



MPC for linear f



MPC for linear \mathbf{f}



a

Locally compute
shares of $\mathbf{f(a,b,c)}$

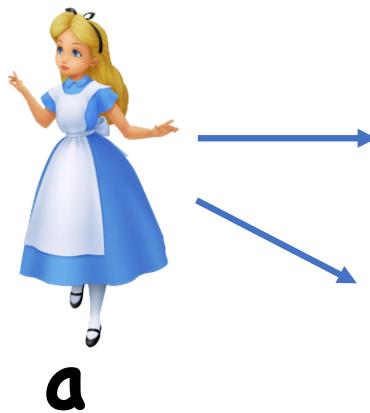


b



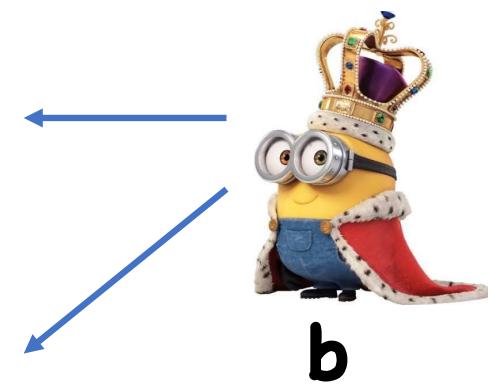
c

MPC for linear f

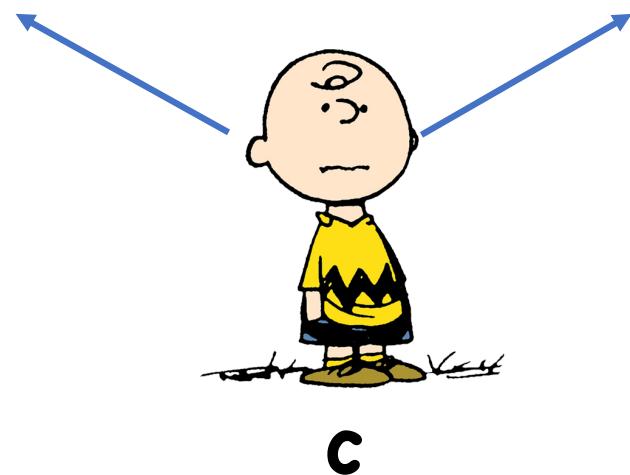


a

Broadcast shares,
then reconstruct



b



c

MPC for General f

Observation 2: Shamir Secret Sharing is sort of multiplicatively homomorphic

Given shares sh_1 of x_1 and sh_2 of x_2 , $sh_1 \times sh_2$ is a share of $x_1 \times x_2$, but with a different threshold

- $sh_1 = P_1(i)$, $sh_2 = P_2(i)$, so
 $sh_1 \times sh_2 = (P_1 \times P_2)(i)$
- $P_1 \times P_2$ has degree $2d$

Idea: can do multiplications locally, and then some additional interaction to get degree back to d

MPC for Malicious Adversaries

So far, everything assumes players act honestly, and just want to learn each other's inputs

But what if honest players deviate from protocol?

Idea: use ZK proofs to prove that you followed protocol without revealing your inputs

Next Time

Wrap up:

- More random topics
- Beyond COS 433

Reminders

HW 7 Due tomorrow

HW 8 Due May 8

Project 3 Due Dean's date