Cryptology and Its Applications

Nordic Research Training course, Bergen

Zero knowledge and some applications

Helger Lipmaa

Helsinki University of Technology

http://www.tcs.hut.fi/~helger

Motivation behind zero-knowledge

- Take *any* reasonably complex protocol
- What happens if the participants misbehave?
  - ⋆ **Chaos and havoc!** ☹
  - ⋆ Think of an electronic payment protocol. . .
- Need to enforce correct behavior
- How?

Idea how to solve

- Participants prove that they behave correctly
  - ⋆ E.g., identification: prove that you know the secret
- After every message, verify the proof
- Privacy: the proof must not reveal any extra knowledge on the secrets of a participant to another one
  - ⋆ E.g., identification: secrets must stay secret
- How? Use *zero-knowledge proofs*

General problem statement

- Let L be some language (set of words), let x be an (encrypted) value
- How to prove that $x \in L$ without giving out any additional knowledge?
  - $\star$ x is positive? x is a full square? x is prime? x is a private key, corresponding to your public key g?
- Generally: How to prove that "I know an x such that $x \in L$"
- *Bad solution*: Send x to verifier. Verifier sees x and can test that $x \in L$; but this gives away more knowledge than is necessary

Usage example: Identification

- Private key: x, public key: g^x
- I want to prove you that I know the secret x
  - ⋆ I.e., that I know discrete logarithm of $g = g^x$
- Privacy: Without revealing x itself!
- Recall that computing discrete logarithms is assumed to be hard

What is knowledge?

- Hard to define—it is easier to define what is *gain of knowledge*
- I tell you $1 + 1 = 2$. Do you gain knowledge?
  - ⋆ Most of you don't ☺
- I tell you the factors of $2^{2^{41}} - 1$. Do you gain knowledge?
  - ⋆ Yes, if you cannot compute them yourselves

Proving with minimal disclosure

- I prove you that I know the factors of $2^{2^{41}} - 1$, without revealing them.
- I prove that two graphs G_1 and G_2 are isomorphic without revealing the isomorphism.
  - ⋆ Graph isomorphism is a well-known hard problem
- I prove that G_1 and G_2 are non-isomorphic, without revealing you why
- In general: I convince you that I know something, without you getting to know anything else but that I know this something
  - ⋆ $\approx$ zero-knowledge.

Knowledge$\neq$Information

Information: You are revealed an unknown object.

- Factors of $2^{2^{41}} - 1$: no new information
- Properties of information are studied in information theory

Knowledge: You are revealed results of calculations, that you cannot perform yourself, on a publicly-known object.

- Factors of $2^{2^{41}} - 1$: probably new knowledge
- Factors of a randomly generated 1024-bit integer: new knowledge, assuming that factoring is hard

Zero-knowledge: Intuition

- *ZK protocol*: protocol between verifier V and prover P
- **Big intuition**: Zero-knowledge is a property of prover P:
  - ⋆ Given a common input x with prover P, whatever any efficient machine V^* can calculate, based on the interaction with P, can be calculated based on x alone
- I.e., interaction with P can be *simulated*
- *Interactive proof system*: P convinces honest V that $x \in L$ iff $x \in L$

Notation

- If $\mathfrak{A}$ is an algorithm, then the notation

$$a \leftarrow \mathfrak{A}(b)$$

  refers to the computation of the output "a", on input bit string "b".

- For a set V, $v \leftarrow V$ denotes uniform and random selection of an element v from V.

- Blue variables are known only to P. Red variables are known only to V, green variables are known to both from the start of the protocol

IP system

- For formal definition of ZK, one must define an *interactive proof system* (IP system)
- IP system captures the completeness/soundness properties but not privacy properties
- IP system consists of two interactive machines that both have private
  - ⋆ (read-only) input, (read-only) random string, read-write working space, (write-only) output
- Machines communicate by sending messages

Preliminaries: Interactive Protocols

- A protocol takes several steps of communications, where in every step one participant sends a message to another one
- An interactive protocol IP is a pair (P, V), where at every step one participant decides, based on the previous communication, private and common inputs, and on the contents of the random tape, what would be the next message
- We assume that P is computationally unbounded
- V is computationally bounded

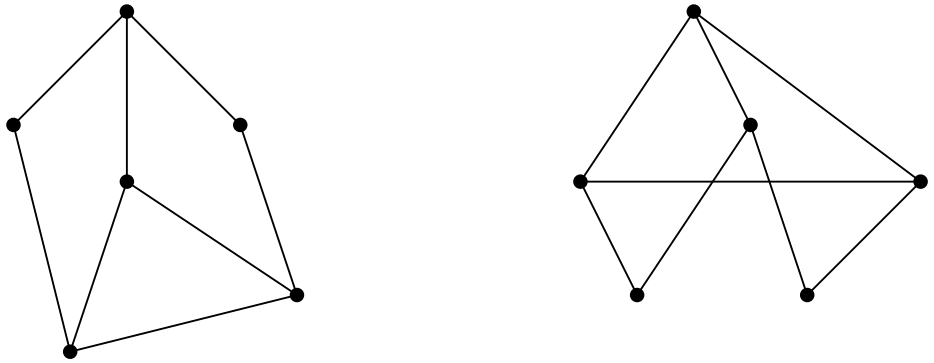
(In *interactive arguments*, P is bounded and V is not.)

Interactive proof system: definition

Language L has an *interactive proof system* if there is such an interactive machine V, so that

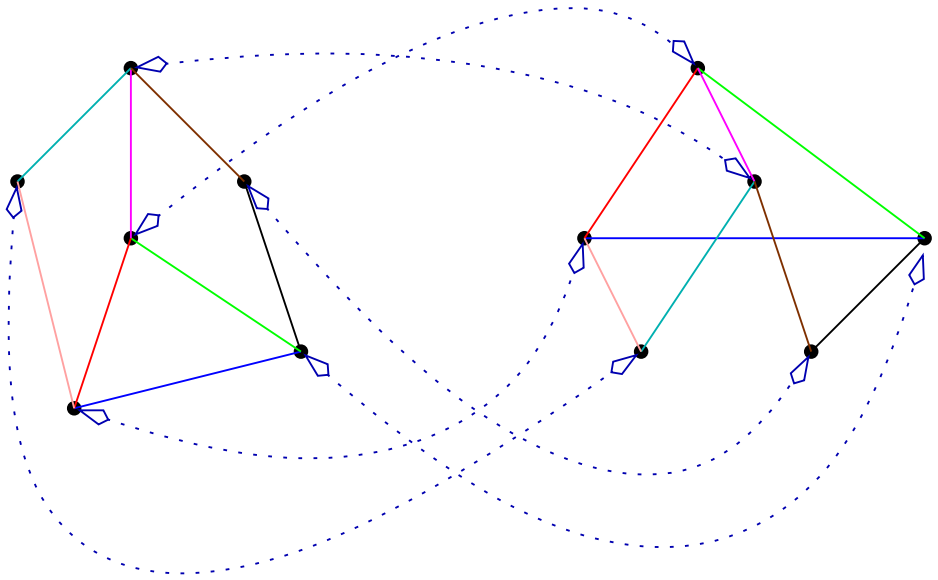
- $\exists P$, so that $\forall x \in L$, V "accepts" x, after a single run of (P, V), with probability $\geq 2/3$
- $\forall P^*$, where (P^*, V) is an IP: For all $x \notin L$, the probability that V "accepts", after a single run of (P^*, V), is $< 1/3$
  - $\star$ Probabilities are taken over the coin tosses of P, V
- Let $\mathcal{IP}$ be the set of languages that have IP proofs

Example: Graph Non-isomorphism



Are these two graphs non-isomorphic?

Example: Graph Non-isomorphism



No! They are isomorphic: we can show an isomorphism (mapping between the nodes).

But how to show non-isomorphism? (How to convince verifier that graphs are non-isomorphic, without sending too much information?)

Example: Graph Non-isomorphism

- A problem is in $\mathcal{NP}$ if we know a short witness
  - ⋆ For graph isomorphism ($\mathbf{GI}$), we can exhibit a short π
  - ⋆ Thus $\mathbf{GI} \in \mathcal{NP}$
- It is not known whether $\mathbf{GNI} \in \mathcal{NP}$
- We will show that $\mathbf{GNI} \in \mathcal{IP}$

IP system for $\mathbf{GNI}$

Common input (G_1, G_2). Iterate the next step for $i = 1 \ldots k$:

- V chooses a random $\alpha_i \leftarrow_R \{1, 2\}$, and a random graph G'_i from the set of graphs that are isomorphic to G_{α_i}. She sends G'_i to P

- (Omnipotent) P finds a graph G_{β_i}, s.t. G_{β_i} and G'_i are isomorphic, and sends β_i to V

  - ⋆ Intuition: P can guess α_i iff graphs are non-isomorphic

V accepts iff $\beta_i = \alpha_i$, $\forall i$

Correctness of IP system for $\mathbf{GNI}$

- When $(G_1, G_2) \in \mathbf{GNI}$:
  - $\star$ P can distinguish isomorphic copies of graph G_1 from isomorphic copies of G_2; then V accepts with probability 1
- When $(G_1, G_2) \notin \mathbf{GNI}$:
  - $\star$ An isomorphic copy of G_1 is always an isomorphic copy of G_2. Thus the best strategy for P is to toss a coin, and hence the cheating probability is again $(1/2)^k$.

Back to ZK and formal definition

- Let us have an interactive proof system (P, V)
- When is (P, V) zero-knowledge?
- Important notion: $\text{view}_V^P(x)$ — *view* of V when interacting with P on common input x
  - ⋆ $\text{view}_V^P(x)$ is equal to the concatenation of all messages sent in this protocol, prefixed with all random coin tosses of V
- View of the previous protocol: $(\alpha_1, \ldots, \alpha_k)||(G'_1, \beta_1, \ldots, G'_k, \beta_k)$

Formal definition (First try)

Definition. Let (P, V) be an IP system for language L. (P, V) is (perfect) *zero-knowledge* if for every machine (probabilistic polynomial-time) machine V^* there exists a PPT *simulator* M^*, s.t. for every $x \in L$ the following two random variables are identically distributed:

- $\text{view}_{V^*}^{P}(x)$ — the *view* of V^* when interacting with P.
- $M^*(x)$ — the output of M^*.

Details

- Too strong a requirement! No non-trivial languages have such proofs.
- Modification: M^* can output $\perp$ with probability $\leq \frac{1}{2}$. If $M^*(x) \neq \perp$ then $\text{view}_{V^*}^P(x) = M^*(x)$. (*Perfect ZK*)
- Alternate modification: $\{\text{view}_{V^*}^P(x)\}_{x \in L}$ and $\{M^*(x)\}_{x \in L}$ are statistically close. (*Statistical ZK*)
  - $\star$ Statistical distance between two distributions is negligible
- Yet another: $\{\text{view}_{V^*}^P(x)\}_{x \in L}$ and $\{M^*(x)\}_{x \in L}$ cannot be distinguished in probabilistic polynomial time.

Intuition

Perfect ZK: The distributions $\text{view}_{V^*}^{P}(x)$ and $M^*(x)$ are equal

Statistical ZK: The distributions $\text{view}_{V^*}^{P}(x)$ and $M^*(x)$ are close

- Even an omnipotent adversary cannot distinguish, given that the protocol is executed (sequentially) not more than a polynomial number of times

Computational ZK: The distributions $\text{view}_{V^*}^{P}(x)$ and $M^*(x)$ cannot be distinguished by a PPT adversary

- Even after a polynomial number of executions

Complexity classification

The classes of languages that have computational/statistical/perfect zero-knowledge proofs:

$$\mathcal{BPP} \subseteq \text{Believed that } \neq \mathcal{PZK} \subseteq \mathcal{SZK} \subseteq \text{Believed that } \neq \mathcal{CZK} = \mathcal{IP} \ .$$

$\mathcal{BPP} \subseteq \mathcal{PZK}$: Trivial, uses no interaction: $\mathcal{PZK}$ can verify by himself whether $x \in L$.

Reminder: $\mathcal{BPP}$ — set of problems that can be decided by probabilistic polynomial-time Turing machines

Example: $\mathrm{GI} \in \mathcal{PZK}$

P knows an isomorphism $\phi : G_1 \to G_2$.

1. P generates a random permutation π of G_2-s vertices. She sends $G' \leftarrow \pi(G_2)$ to V.

2. V generates a random $\sigma \leftarrow \{0, 1\}$ and sends it to P.

3. If $\sigma = 1$, P sets $\tau \leftarrow \pi \circ \phi$, otherwise she sets $\tau \leftarrow \pi$. She sends τ to V.

4. V checks that $\tau(G_\sigma) = G'$.

Intuition: $\pi(\phi(G_1)) = \phi(G_2) = G'$.

$\mathcal{NP} \subseteq \mathcal{CZK}$

- To show that there are CZK proofs for every $\mathcal{NP}$-language, it is sufficient to show a proof for one concrete $\mathcal{NP}$-complete language
- A graph G can be colored with c colors when there exists an coloring of the vertices of G with c colors so that for no edge, the vertices connected to this edge are colored with the same color
- The *chromatic number* of G, $\chi(G)$: minimum c so that G can be colored with c colors
- $3COL$: the set of graphs with $\chi(G) \leq 3$. This language is $\mathcal{NP}$-complete. Say the colors are R, G, B.

CZK protocol for $3COL$

Common input: G. P wants to prove that she knows a coloring $C : V(G) \to \{R, G, B\}$ in CZK. Iterate the next protocol $|E(G)|^2$ times:

- P chooses a random permutation π of colors. She encrypts the color $\pi(C(v))$ for every vertex v, using a probabilistic public-key cryptosystem, by using a different key for every vertex. P sends to V all ciphertexts together with the correspondence between them and the vertices
- V chooses a random edge $e = (v_1, v_2)$, and sends e to P
- P sends the decryption keys D_{v_1} and D_{v_2} to V
- V computes $\pi(C(v_1))$ and $\pi(C(v_2))$ and verifies that they are different

Correctness of the protocol for $3COL$

- If P knows the corresponding 3-coloring, V will never detect an incorrectly colored edge. Thus, V will accept with probability 1

- If $\chi(G) > 3$ then $\pi(C(v_1)) = \pi(C(v_2))$ in all steps with probability $\geq |E|^{-1}$. After $|E|^2$ steps the probability that V will accept is exponentially small

Is $\mathbf{GNI}$ in $\mathcal{CZK}$?

- Previously presented IP system for $\mathbf{GNI}$ is not zero-knowledge:
- V can submit an arbitrary graph H *not necessarily isomorphic to* G_1 *or to* G_2 and thus get to know additional information
- Modify the previous protocol by letting V to prove in $\mathcal{PZK}$ that G_i' is either isomorphic to G_1 or to G_2
  - $\star$ Doable, since $\mathbf{GI} \in \mathcal{PZK}$

Zero-Knowledge: Limitations

- ZK protocols require more than three moves unless the underlying language is trivial (in $\mathcal{BPP}$). Thus, in principle, none of the three-move protocols handled here can be ZK.

- Four-move ZK protocols exist.

- The very efficient procedure for turning identification schemes into signature schemes, presented later, cannot be used if the identification scheme is ZK (the simulation used for proving the ZK-ness can be used to forge the signature). Thus, a real ZK protocol cannot be used to construct such a signature scheme.

Honest Verifier ZK

- A party is *honest/nonmalicious/curious-but-honest* when he follows the protocol (though tries to deduce new information from it)
- (P, V) is *honest verifier ZK* if it is ZK with respect to honest V.
- General ZK protocols are far less efficient. HVZK is achievable in 3 rounds. HVZK is sufficient in many application.
- There exist efficient transformation methods for turning certain classes of HVZK protocols into ZK ones.

Non-Interactive ZK

- A ZK protocol is non-interactive, if it consists of only one step: prover sending some information to verifier
- NIZK protocols exist only if P and V have access to some common, publicly available source of random strings (beacon)
- NIZK honest-verifier protocols exist also in random-oracle model

Concept: random oracle

- Random oracle H = random function
  - ⋆ For every x, $H(x)$ is randomly drawn from the output domain
- Implementation:
  - ⋆ H is a subroutine with initially empty database (a, c). $H(a)$ returns c if (a, c) is in the database for some c. Otherwise H generates uniformly a new c, adds (a, c) to the database and returns newly generated c.
- In practice, a secure hash function (SHA1) is used

Motivation behind proofs of knowledge

- For an $\mathcal{NP}$-language L, L can also be seen as a relation, $L = \{(x, w)\}$, where w is an $\mathcal{NP}$-witness that $x \in L$.
  - ⋆ Definition of $\mathcal{NP}$: $x \in L$ iff $\exists w$, s.t. for some polynomial-time machine A, $A(x, w) =$ Accept.
- IP proof: shows that a property holds (non-constructive)
- Proof of knowledge: shows that the verifier knows the corresponding witness (constructive)
  - ⋆ In security proofs, we can assume that the prover knows a short witness, not that it is able to generate it on fly

ZK Proof of Knowledge: Definition

Π is a ZK proof of knowledge iff

- Π is an interactive proof system with zero-knowledge property
- Proof of knowledge:
  - ⋆ If P can make V accept, there is a *knowledge extractor* that, given oracle access to P, and for any $x \in L$, can extract a witness ω such that $(x, \omega) \in L$.
  - ⋆ Knowledge extractor M^* can rewind P: i.e., execute the protocol (P, M^*) several times, with the same common input x and the same random tape of P.

Knowledge extractor in $\mathcal{CZK}$ protocol for $3COL$

- M^* executes this protocol $|E(G)|$ times, in every time choosing a different edge $(v_1, v_2) \in E(G)$ during the first step of the protocol
- At the end, M^* has $\pi(C(v_1))$ for every vertex v of the graph G
- $\omega := \pi \circ C$ is a valid three-coloring of G
- Thus M^* has extracted a witness ω, s.t. $(x, \omega) \in 3COL$

Σ-protocols: idea

- "challenge-response" proof of knowledge:
  - $\star$ P sends a random-looking element to V,
  - $\star$ V challenges P with a uniformly random bit-string,
  - $\star$ P responds
- Such a three-round protocol is known as a Σ-*protocol*

Σ-Protocols: Notation

- The pair $(x, \omega) \in R$, where $R \subset \{0,1\}^* \times \{0,1\}^*$ is a publicly known, typically (but not necessarily) efficiently verifiable relation. Let

$$R_W(x) := \{\omega : (x, \omega) \in R\} \quad \text{and}$$
$$R_X := \{x : R_W(x) \neq \emptyset\} \ .$$

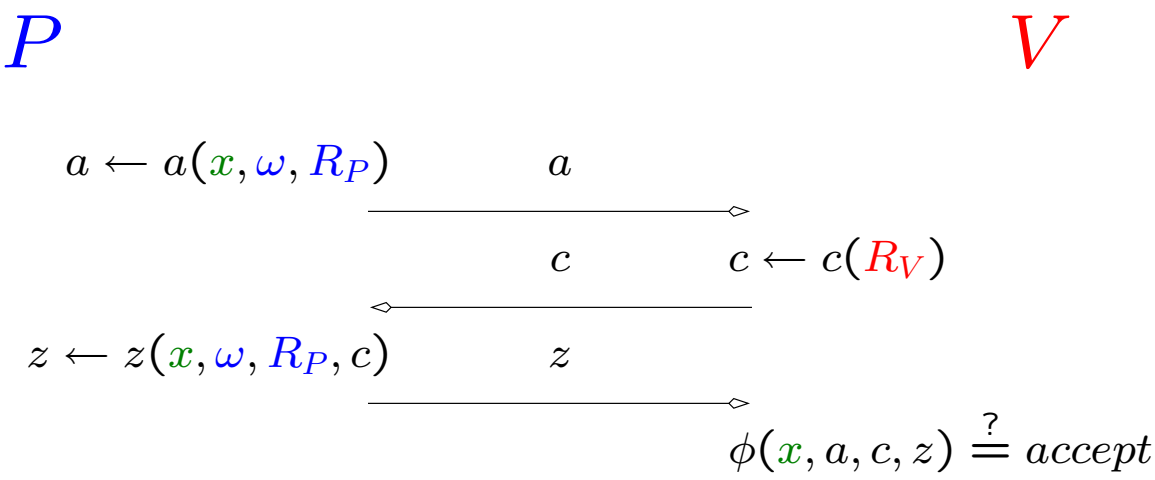
- Intuitively: $R_W(x)$ is the set of witnesses corresponding to common input x, and R_X is the set of secret keys that have a corresponding public key.

- Simplification: assume that all witnesses ω correspond to some value x, s.t. $R(x, \omega) \in R$. I.e., R_X is the set of public keys. (For some well-known schemes like the Guillou-Quisquater, this is not the case!)

Proofs of knowledge: notation

- Denote a *proof of knowledge of* ω, *s.t.* $R(x, \omega) = 1$ by $PK(R(x, \omega) = 1)$
- Greek letters denote variables, knowledge of which is to be proved
- Latin letters denote variables that are either in public knowledge or secretly owned by some party
- $PK(y = g^{\omega})$: proof of knowledge of the discrete logarithm
- $PK(y = C_K(\mu; \rho) \wedge \mu \neq 0)$ (proof of knowledge of committed non-zero message μ)

Σ-Protocols: General Description



a: *initial message*. $t_P = |a|$ is the *authentication length*.
c: *challenge*, $c \leftarrow \{0,1\}^{t_V}$.
z: *reply* (may reuse a).

Finally, V invokes a polynomial time computable predicate ϕ to check whether the *conversation* (v, a, c, z) is *accepting*.

Σ-Protocol for knowledge of DL

- Let p, q be two large primes s.t. $q \mid (p-1)$. Let G_q be the unique subgroup of $\mathbb{Z}_p^*$ of order q. Let g be the generator of G_q.

- Let $\omega \leftarrow_R \mathbb{Z}_q$ and let $h := g^\omega$.

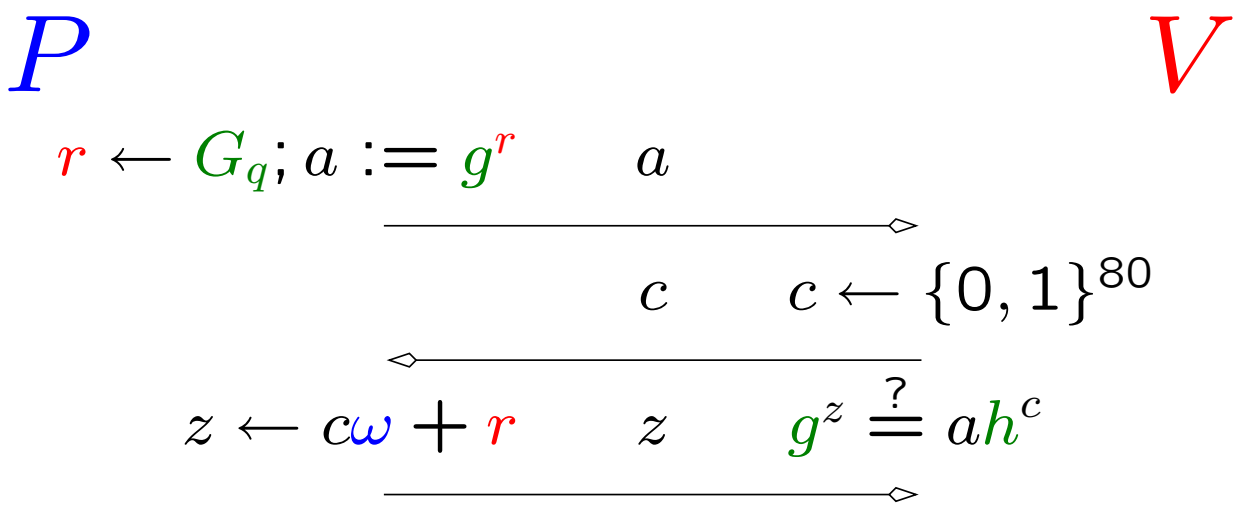
- Let $x = (G_q, g, h)$ be the common input, ω be the private input to P.

  - ⋆ The corresponding (unique) witness is $\omega \in \mathbb{Z}_q$ such that $g^\omega = h$. The relation R consists of all such pairs, $R = (g^\omega, \omega)$.

- Next, we show a Σ-protocol for $PK(h = g^w)$.

Schnorr's $PK(h = g^{\omega})$

Let $h = g^{\omega}$. Let $x = (G_q, g, h)$ be the common input, ω is the private input to P.



Completeness: $g^z = g^{c\omega + r} = g^r (g^{\omega})^c = a h^c$.

Schnorr's proof of knowledge: efficiency

- Communication: $\approx |p| + t + |q|$.
- On-line: one $|q| \times 80$ bit multiplication (and one t-bit addition). Random number generation and exponentiation can be done off-line, during the processor's idle time.
- If the scheme is used only for identification, where the prover has to reply to the challenge in a few seconds, the security parameter t_V can be lowered, say, to 48 bits.

Security Properties: Special Soundness (1/2)

- Let $x \in \{0,1\}^*$ be a string. A pair of accepting conversations (x, a, c, z) and (x, a, c', z') with $c \neq c'$ is called a *collision*.
  - ⋆ Collision occurs if the same person starts identification two times with the same first message, is answered by a different second message, and is accepted both times
- Σ-protocol (P, V) has *the special soundness* property if the following holds:
  - ⋆ Given a collision for a public key x, there exists an efficient algorithm that on input of a collision for x outputs a witness ω such that $(x, \omega) \in R$.

(Given security definitions are “simplified”)

Special Soundness (2/2)

- Intuitively, special soundness guarantees that P does not have an incentive to start the same protocol twice with the same message.
  - ⋆ She must include some randomness to not reveal her secret.
- Corresponds to the "proof of knowledge" property, but somewhat stronger.
  - ⋆ Knowledge extractor has to execute P only twice to extract the witness.

Schnorr's proof of knowledge: Special Soundness

- Given two accepting conversations (x, a, c, z) and (x, a, c', z')

  $\star$ $g^z = ah^c$ and $g^{z'} = ah^{c'}$

  with $c \neq c'$, ω is computed as

$$\omega \leftarrow \frac{z - z'}{c - c'} \ ,$$

  since

$$\frac{z - z'}{c - c'} = \frac{(c\omega + r) - (c'\omega + r)}{c - c'} = \frac{(c - c')\omega}{c - c'} = \omega \ .$$

- Thus, the Schnorr scheme satisfies special soundness.

Schnorr's proof of knowledge: Special HVZK

- Some simulator M^* must be able to generate an accepting conversation without communicating with Alice
  - ⋆ With the same distribution as "real" conversations
  - ⋆ "Special HVZK": this is achieved by first selecting randomly the second and the third message from corresponding domains, and then selecting the first message, s.t. the verification accepts
  - ⋆ Stronger than "non-special" HVZK
- Select $c, z \leftarrow \mathbb{Z}_q$, compute $a \leftarrow g^z \cdot h^{-c}$. Then (x, a, c, z) is an accepting conversation with the correct distribution.

Σ-protocol for $PK(A_1 \wedge A_2)$.

- Assume you are given a Σ-protocol for $PK(A_1)$ and $PK(A_2)$, where A_1 and A_2 are some predicates

- To construct a Σ-protocol for $PK(A_1 \wedge A_2)$
  - $\star$ Run Σ-protocols for $PK(A_1)$ and $PK(A_2)$ in parallel
  - $\star$ But use the same challenge in both

Σ-protocol for $PK(A_1 \vee A_2)$.

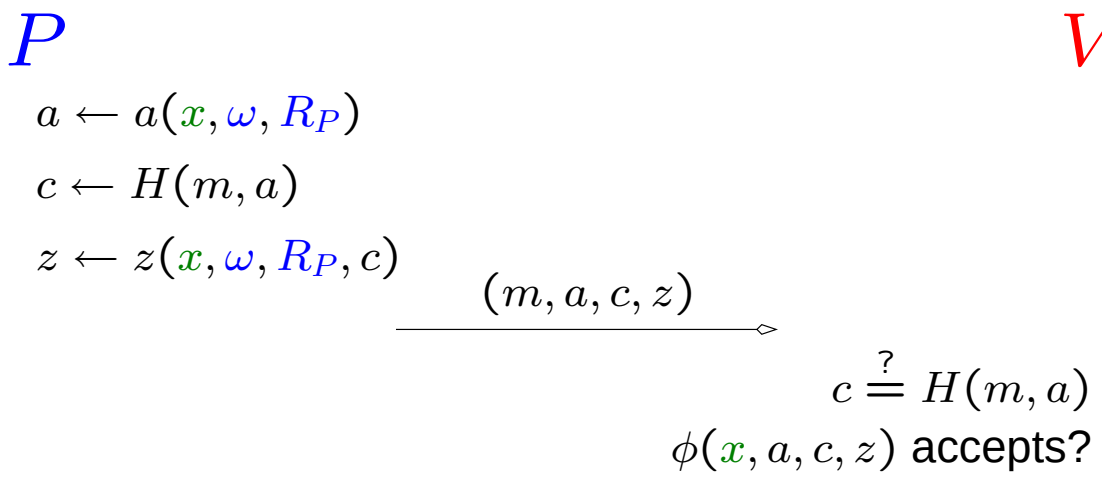
- Assume A_1 is true (other case is dual)
- P does: Generate a_1 as in $PK(A_1)$. Run the simulator to produce valid view (a_2, c_2, z_2) as in $PK(A_2)$. Send (a_1, a_2) to V.
- V generates a random $d \leftarrow \{0,1\}^{t_V}$ and sends it to P
- P computes $e_1 \leftarrow d - e_2 \mod 2^{t_V}$, and z_1 as it would be computed in $PK(A_1)$ after the first messages a_1, c_1. P sends (z_1, z_2) to V
- For $i \in \{1,2\}$, V performs the check, as done in $PK(A_i)$, on (a_i, c_i, z_i). He also checks that $e_1 + e_2 = d \mod 2^{t_V}$.

Application: identification schemes

- To get access, prove that you know your secret key
  - ⋆ Smart doors: use smart-card to get in
  - ⋆ ATM: identify yourself as a legal customer
- *Common problem*: must avoid re-execution of the protocol
- Use, e.g., Schnorr's Σ-protocol $PK(h = g^{\omega})$

Application: signature scheme

Assume H is a random oracle. Specially HVZK, specially sound, Σ-protocol (P, V) can be converted into the generic signature scheme $Sign(P)$ by using the next general method:



Signature of m is equal to (a, c, z)

H is a RO: c is "random", but "depends" on (m, a)

Security of resulting signature scheme

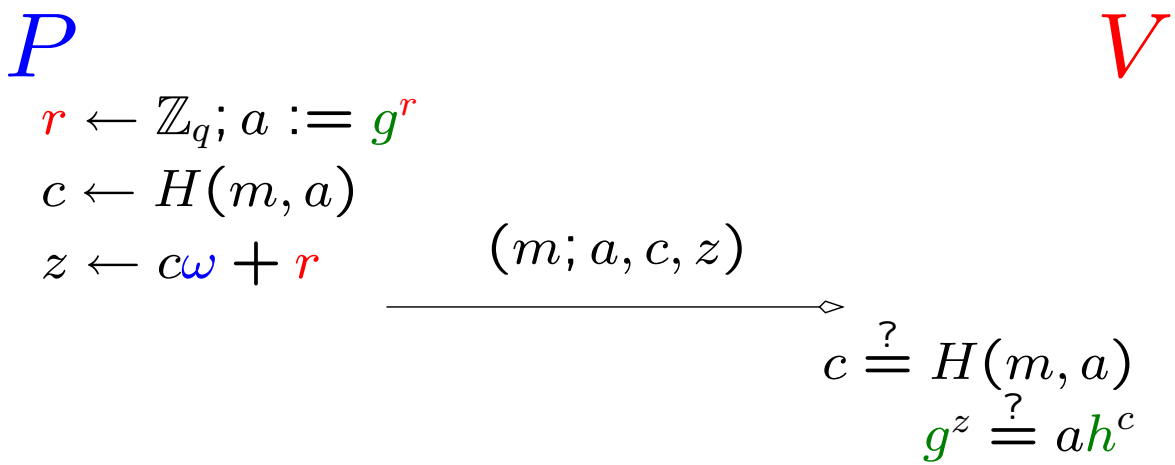
Fix a signature scheme $Sign(P)$. An adversary (q, t, ε)-*forges* P's signature if it works in time t and with success probability ε, can generate a tuple (m, a, c, z) such that $\phi(m, a, c, z)$ accepts. Adversary is allowed to ask P to sign up to q different messages (not equal to m).

Forking lemma (Pointcheval, Stern). Assume that some algorithm A (q, t, ε)-forges a signature (m, a, c, z), with $\varepsilon \geq 7q/2^k$, where k is the security parameter. Then there exists another machine, that with oracle access to A, can produce a collision (m, a, c, z), (m, a, c', z'), with $c \neq c'$, in expected time $t' \leq 84480tq/\varepsilon$.

E.g.: $\varepsilon = 2^{-50}$, $q = 2^{40}$, then $t' \leq 2^{77}t$.

Schnorr Signature Scheme

Let $h := g^{\omega}$. Let $x = (G_q, g, h)$ be the common input, ω is the private input to P.



Check: $g^z = g^{c\omega + r} = g^r (g^{\omega})^c = g^{\omega} h^c = ah^c$.

Schnorr's signature scheme: efficiency

- P has to perform on-line one H evaluation, one 160-bit multiplication and one addition.

- Communication can be reduced: P sends (m, c, z) and V verifies that $s = H(m, g^z h^{-c})$.

  - ⋆ Thanks to the special HVZK property

  - ⋆ Same trick works for any converted signature scheme

Signature conversion: Caveats (1/2)

In practice, H is a standard hash function

- In such a case, the conversion scheme looses provable security
- For some concrete identification schemes, the conversion works if H is the random oracle, but not for *any* instantiation of H by a real hash function. (Goldwasser, Tauman, 2003)

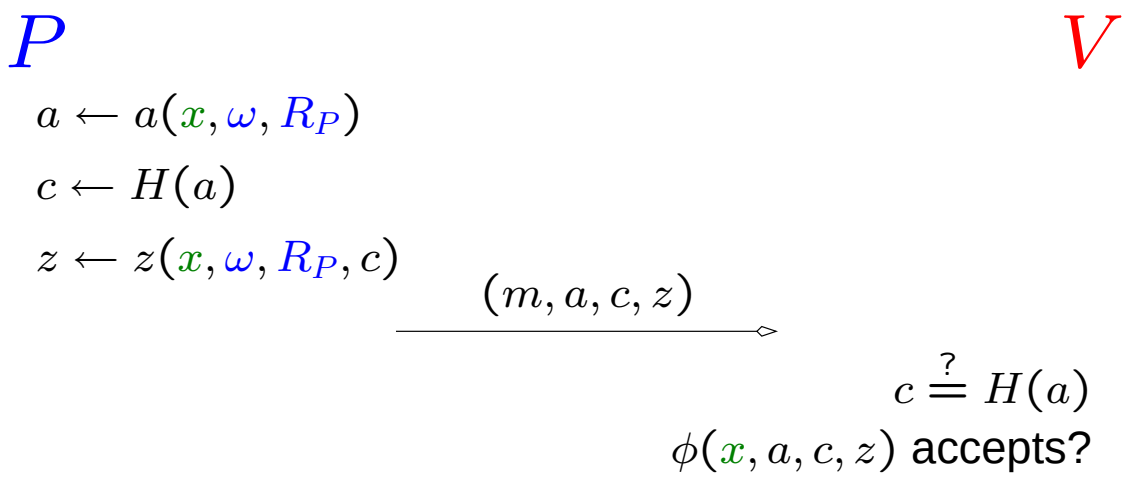
Signature conversion: Caveats (2/2)

- If both identification scheme and signature are used in the same smart-card, some care has to be taken. Namely, during the identification scheme V can output as the challenge $c = H(m, a)$ for m chosen by her. After receiving z from P, V will own a legitimate signature (a, c, z) of m.

- Solution (Schnorr's scheme): P sends the 80 least significant bits of a during the step 1. There is no known attack in this case.

Application: NIZK

Assume H is a random oracle. Specially HVZK, specially sound Σ-protocol (P, V) can be converted into a non-interactive zero-knowledge (NIZK) proof (a, c, z) by using the next general method:



c is random, but depends on a. It is NIZK only when H is a random oracle.

Thanks to special HVZK property, the NIZK proof can be usually shortened to (c, z).

Witness hiding (1/2)

- Let (P, V) be any Σ-protocol for some relation R
- *Witness hiding*: no matter how maliciously the enemy interrogates an honest prover, it gets at most a negligible advantage when trying to compute any ω in $R_W(x)$, compared to the situation before the start of the protocol
- ZK guarantees that no information whatsoever is revealed in case of any fixed common input v
- Difference: Witness hiding only guarantees that no *useful* information is given away in the average, but it does it also against a *dishonest* verifier

Witness hiding (2/2)

- It was not known if Schnorr's scheme is witness hiding. Very recently, Schnorr's scheme's security against impersonation has been finally proven.

  M. Bellare and A. Palacio, "GQ and Schnorr Identification Schemes: Proofs of Security against Impersonation under Active and Concurrent Attacks Authors", CRYPTO 2002 (august 2002)

More applications of Σ-protocols

Aside from identification and signing, Σ-protocols are also extensively used to prove the correctness of behavior in many protocols. For example:

- Blind signature/digital cash protocols
- Electronic voting
- Electronic auctions
- ...

Detour: Commitment Schemes

- P has private key K. Using this key and a random value r, she *commits* to x by sending $C_K(x; r)$ to V
- Later, P reveals x and V can verify that this is the value that was previously committed
- *Hiding property*: V is not able to compute x from $C_K(x; r)$
- *Binding property*: P cannot generate (x', r'), $x' \neq x$ in the plaintext set, s.t. $C_K(x; r) = C_K(x'; r')$

Application: Joint coin tossing

- Alice and Bob want to decide on something by tossing a coin over a phone. How to do this securely?
- Solution: Alice commits to a random bit $b_A \leftarrow_R \{0,1\}$, and sends $C_K(b_A; r)$ to Bob
- Bob selects a random bit $b_B \leftarrow_R \{0,1\}$ and sends it to Alice
- Alice decommits b_A
- Alice and Bob compute the coin toss as $b_A \oplus b_B$

Meta-application: Artificial Synchronicity

- In real life, protocols are asynchronous, no two messages are received simultaneously
- Easier to design secure protocols in synchronous setting
- Simulate synchronocity: Alice commits to her message, Bob sends his, Alice opens hers
- Alice's and Bob's messages are independent ☺
- Alice can refuse to open her message when she does not like Bob's ☹
  - ⋆ Partially solved with *fair exchange* protocols ☺

Pedersen commitment scheme

Assume that $p = 2q + 1$ is a safe prime (i.e., q is also prime)

Set-up Let h be a generator of G_q, a subgroup of $\mathbb{Z}_p^*$ of prime order q. Let $g \leftarrow_R G_q$

Commitment $C_K(m; r) = g^m h^r \mod p$ where $r \leftarrow_R \mathbb{Z}_q$

Opening Reveal m and r

Pedersen: Proof of security

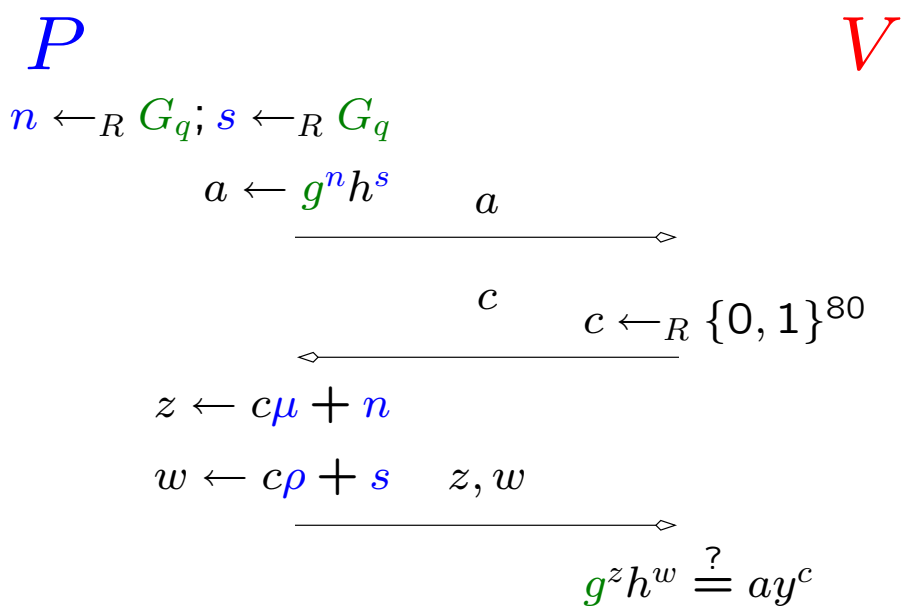
Unconditional hiding : Since r is a random element of $\mathbb{Z}_q$ then $g^m h^r$ is a random element of G, independently of the choice of m

Computational binding : Given $(m; r)$, $(m'; r')$, s.t. $g^m h^r = g^{m'} h^{r'}$, $m \neq m' \mod q$, one can compute $g \leftarrow h^{(r-r')/(m'-m)}$. (This is valid since $m \neq m'$, q is prime and therefore $(m' - m)^{-1}$ exists.) Therefore, the adversary has computed the DL of g in base h

The security proofs are similar to the security proofs of Schnorr's identification scheme.

HVZK: protocols about commitments

Pedersen commitment scheme. Proof that P knows how to open $y = C_K(\mu; \rho)$:



Commitment + Σ-protocol $\Rightarrow$ ZK

- Design a 3-round Σ-protocol between P and V:
  - $\star$ P sends the first and the third steps, V sends a random string on the second step.
- In practice, hard to guarantee that V does not cheat
- Solution:
  - $\star$ V selects his challenge c and commits to it before seeing P's first messages
  - $\star$ P sends then her first message, V opens his commitment, and P sends her second message

Advanced example: Vickrey Auctions

- You have a limited number of options: bidding $\mu \in [0, H]$
- You bid by encrypting your bid and sending it to some center
- Goal: seller S should not be able to decrypt your bid; but she should get to know the second highest bid X_2
- Solution: Encrypt by using the public key of another center A but send encryption to S

Lipmaa, Asokan, Niemi. Secure Vickrey Auctions without Threshold Trust. Financial Cryptography 2002. Bermuda.

http://www.tcs.hut.fi/~helger/papers/

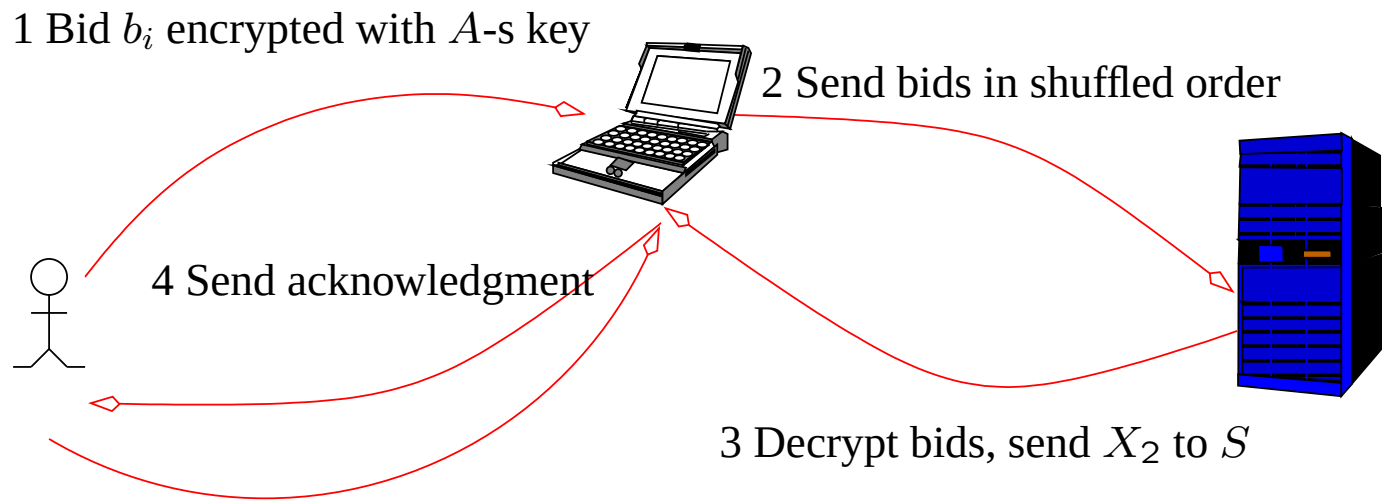
Advanced example: Auctions, 2

- Assume E is *homomorphic*: $E_K(m)E_K(m') = E_K(m + m')$
- Instead of bid μ, encrypt B^μ, where B is the maximum number of bidders
- S multiplies all ciphertexts, obtaining $c \leftarrow E_K(\sum_i B^{\mu_i})$. Due to the choice of B, this is equal to $E_K(\sum_j \alpha_j B^j)$, where α_j is the number of bidders who bid j
- S sends c to A, who decrypts c, and obtains all values α_j. A calculates the second highest bid X_2, and sends X_2 to S
- S announces X_2 to bidders

Advanced example: Auctions, 3

- Protocol, works only when different parties are honest
- Standard solution: Add NIZK proofs of knowledge that every step was correct
  - ⋆ Same methodology used in almost all cryptographic protocols!
- Every bidder NIZK-proves that it encrypted a valid bid B^{μ}, $\mu \in [0, H]$
- And: A NIZK-proves that A computed X_2 correctly

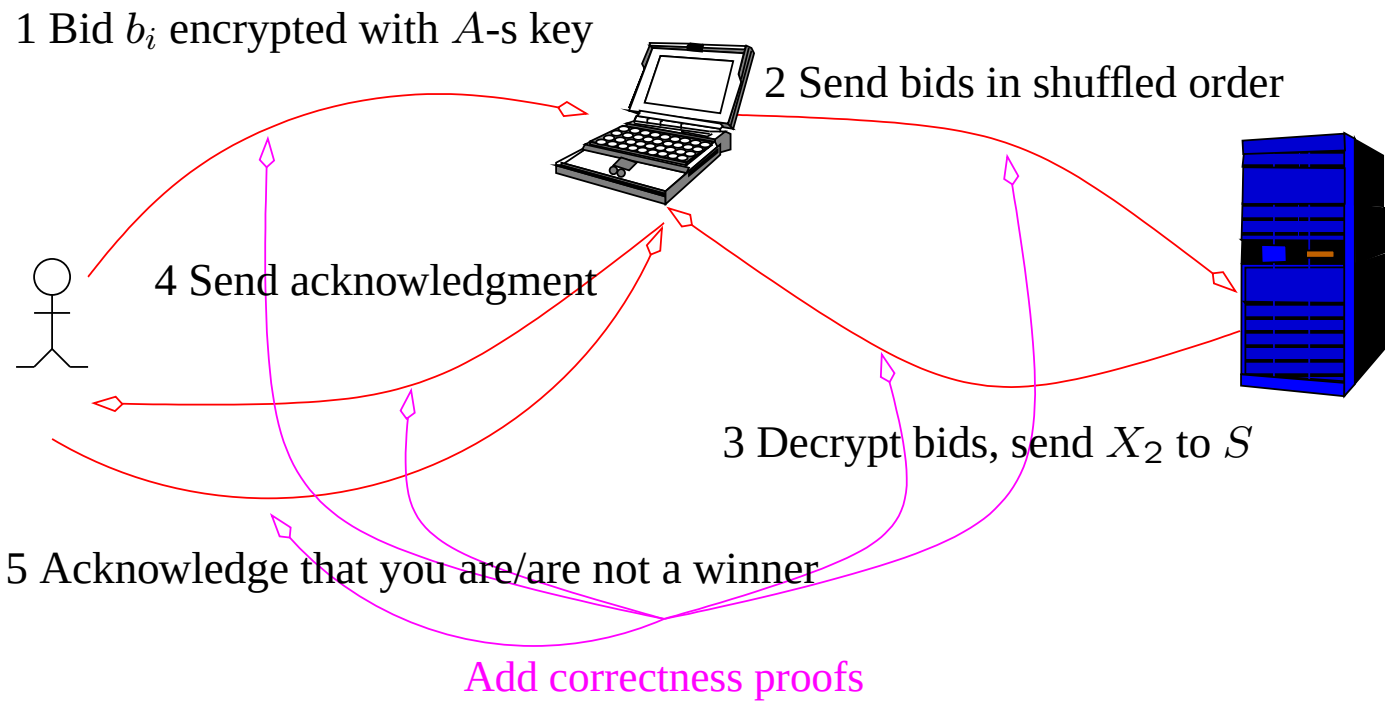
Simple scheme



S will not get any extra information, but S can increase X_2

$P \rightarrow S$ interaction is quite large

Simple scheme $\longrightarrow$ secure scheme



Just add NIZK proofs of knowledge ☺

$PK(y = E_K(B^\mu; \rho) \wedge (\mu \in [0, H]))$

- Denote $H_j := \lfloor (H + 2^j)/2^{j+1} \rfloor$, $j = 0 \ldots \lfloor \log_2 H \rfloor$. Then

$$\mu \in [0, H] \iff \mu = \sum_{j=0}^{\lfloor \log_2 H \rfloor} \mu_j H_j \quad \text{for some } \mu_j \in \{0, 1\} \ . \quad (1)$$

- For example, $\mu \in [0, 10] \iff \mu = 5\mu_0 + 3\mu_1 + \mu_2 + \mu_3$ and $\mu \in [0, 9] \iff \mu = 5\mu_0 + 2\mu_1 + \mu_2 + \mu_3$.

- NIZK proof of knowledge of μ_j for which the right side (1) holds

How to prove that X_2 is correct?

- You have

$$y = E_K(\sum_j \alpha_j B^j) \ .$$

  You must NIZK-prove that

$$PK(y = E_K(\mu;\rho) \wedge \mu = B^{\mu_1} + B^{X_2} + \mu_2 \wedge \mu_1 > X_2 \wedge \mu_2 < B^{x_2}) \ .$$

- Doable in honest-verifier computational zero-knowledge, but inefficient

Security properties

If A and S do not cooperate:

- A will not be able to change the highest bid or bidder
- S will not get to know anything about the bids
- A will know the statistics (how many bid j) but no individual bids
- System can be strengthened: even cooperating A and S will not be able to change the highest bid or bidder

E-voting

- E-voting: can do analogously. Bidder = voter, bid = vote
- S must get to know α_j, so instead of X_2 a NIZK proof of its correctness A will send to her the sum $\sum_j \alpha_j B^j$ with a proof of correct decryption (simpler!)
- Problem: Can we trust that S and A do not to cooperate?
- If not, another possibility is to share the trust among a larger number of authorities

Detour: Integer commitment schemes

- *Integer commitment scheme*: like a commitment scheme but for given (m_1, r_1), hard to find (m_2, r_2), $m_1 \neq m_2$ *over integers*, such that $C_K(m_1; r_1) = C_K(m_2; r_2)$

- [Fujisaki, Okamoto], [Damgård, Fujisaki] — *statistically hiding, computationally binding*

- The known ICS-s are homomorphic, $C_K(m_1; r_1)C_K(m_2; r_2) = C_K(m_1 + m_2; r_1 + r_2)$

- There exist efficient honest-verifier *statistical* zero-knowledge (HVSZK) proofs of knowledge that y_3 commits to a product/sum of integers, commited to by y_1 and y_2

Detour: Integer commitment schemes

- [Lipmaa, 2003]: given an ICS C, can construct efficient *HVSZK arguments of knowledge* for all languages in bounded arithmetic

- $PK(h = C_K(\mu; \rho) \wedge \mu \geq 0)$:
  - ⋆ Use Lagrange's theorem that every nonnegative integer is a sum of four squares
  - ⋆ Prove that $h = C_K(\mu; \rho) \wedge h_1 = C_K(\omega_1; \rho_1) \wedge \cdots \wedge h_4 = C_K(\omega_4; \rho_4) \wedge \mu = \omega_1^2 + \cdots + \omega_4^2$
  - ⋆ [Groth, 2004] Prove that $4\mu + 1$ is a sum of three squares

- $PK(h = C_K(B^\mu; \rho) \wedge \mu \in [0, H]$:
  - ⋆ [Lipmaa]: Assume B is a prime. Prove that $\mu \mid B^\mu \wedge \mu \geq 0$
  - ⋆ [Damgård, Groth, Salomonsen]: Assume $B = p^2$, p is a prime. Prove that $h_1 = C_K(\omega; \rho_1)$, $\mu = \omega^2$, $\omega \mid p^H$

Secure encryption

- Goal: design a public-key cryptosystem that is secure against chosen-ciphertext attacks
- Step 1: design a cryptosystem that is semantically secure
  - ⋆ Easier
- Step 2: Add a ZK proof of knowledge that the encrypter knows the plaintext

Big caveat: random oracle model

- ☹ *All previous examples that use NIZK work in the random oracle model*
- ☹ *Random oracles do not exist*
- ☹ *Several protocols are known that are secure in the ROM, but when the RO is instantiated with any existing function f, the protocols become insecure*
- ☹ *If a protocol would be instantiable, ROM would not be needed in the first place*

Major research issue: eliminate ROM

- Construct similar non-interactive protocols without random oracles. Cannot use zero-knowledge. May be something weaker suffices?

- See [Barak, Ong, Vadhan, 2002], [Barak, Pass, 2004]

Conclusions

- ZK is a very useful concept that enables to design secure protocols for almost anything
- Constructing efficient ZK proofs is not easy, and must often be accompanied with profound knowledge of the subject area
- This lecture gave some overview of ZK and its applications
  - ⋆ Not very formal, not very complete, not very extensive
- Some universities offer whole courses on ZK!