11. ZERO-KNOWLEDGE PROOF PROTOCOLS

One of the most important, and at the same time very counterintuitive, primitives for cryptographic protocols are so called zero-knowledge proof protocols (of knowledge).

Very informally, a zero-knowledge proof protocol allows one party, usually called PROVER, to convince another party, called VERIFIER, that PROVER knows some facts (a secret, a proof of a theorem,...) without revealing to the VERIFIER **ANY** information about his knowledge (secret, proof,...).

In this chapter we present and illustrate very basic ideas of zero-knowledge proof protocols and their importance for cryptography.

Zero-knowledge proof protocols are a special type of so-called interactive proof systems.

By a *theorem* we understand here a claim that a specific object has a specific property. For example, that a specific graph is 3-colorable.

INTERACTIVE PROOF PROTOCOLS

In an interactive proof system there are two parties

- An (all powerful) Prover, often called Peggy (a randomized algorithm using a private random number generator);
- A (little (polynomially) powerful) Verifier, often called Vic (a polynomial time randomized algorithm using a private random number generator).

Prover knows some secret, or a knowledge, or a fact about a specific object, and wishes to convince Vic, through a communication with him, that he has the above knowledge.

For example, both Prover and Verifier posses an input x and Prover wants to convince Verifier that x has a certain properties and that Prover knows how to proof that.

The interactive proof system consists of several rounds. In each round Prover and Verifier alternatively do the following.

1. Receive a message from the other party.
2. Perform a (private) computation.
3. Send a message to the other party.

Communication starts usually by a challenge of Verifier and a response by Prover.

At the end Verfifier either accepts or rejects Prover's attempts to convince Verifier.

Example – GRAPH NON-ISOMORPHISM

A simple interactive proof protocol exists for computationally very hard graph non-isomorphism problem.

Input: Two graphs G_1 and G_2, with the set of nodes $\{1, \ldots, n\}$

Protocol: Repeat n times the following steps:

1. Vic chooses randomly an integer $i \in \{1, 2\}$ and a permutation π of $\{1, \ldots, n\}$. Vic then computes the image H of G_i under permutation π and sends H to Peggy.
2. Peggy determines the value j such that G_j is isomorphic to H, and sends j to Vic.
3. Vic checks to see if $i = j$.

Vic accepts Peggy's proof if $i = j$ in each of n rounds.

Completeness: If G_1 is not isomorphic to G_2, then probability that Vic accepts is clearly 1.

Soundness: If G_1 is isomporphic to G_2, then Peggy can deceive Vic if and only if she correctly guesses n times the i Vic choosed randomly. Probability that this happens is 2^{-n}.

Observe that Vic's computations can be performed in polynomial time (with respect to the size of graphs).

INTERACTIVE PROOF SYSTEMS

An interactive proof protocol is said to be an interecative proof system for a secret/knowledge or a decision problem Π if the following properties are satisfied.

Assume that Prover and Verifier posses an input x (or Prover has secvret knowledge) and Prover wants to convince Verifier that x has a certain properties and that Prover knows how to proof that (or that Prover knows the secret).

(Knowledge) Completeness: If x is a yes-instance of Π, or Peggy knows the secret, then Vic always accepts Peggy's "proof" for sure.

(Knowledge) Soundness: If x is a no-instance of Π, or Peggy does not know the secret, then Vic accepts Peggy's "proof" only with very small probability.

CHEATING

- If the Prover and the Verifier of an interactive proof system fully follow the protocol they are called honest Prover and honest Verifier.
- A Prover who does not know secret or proof and tries to convince the Verifier is called cheating Prover.
- A Verifier who does not follow the behaviour specified in the protocol is called a cheating verifier.

Zero-knowledge proof protocols informally

Very informally An interactive "proof" protocol at which a Prover tries to convience a Verifier about the truth of a statement, or about possesion of a knowledge, is called "zero-kowledge" protocol if the Verifier does not learn from communication anything more except that the statement is true or that Prover has knowledge (secret) she claims to have.

Example The proof $n = 670592745 = 12345 \times 54321$ is not a zero-knowledge proof that n is not a prime.

Informally: A zero-knowledge proof is an interactive proof protocol that provides **highly convincing evidence** that a statement is true or that Prover has certain knowledge (of a secret) and that Prover knows a (standard) proof of it while providing **not a single bit of information** about the proof (knowledge or secret). (In particular, Verifier who got convinced about the correctnes of a statement cannot convince the third person about that.)

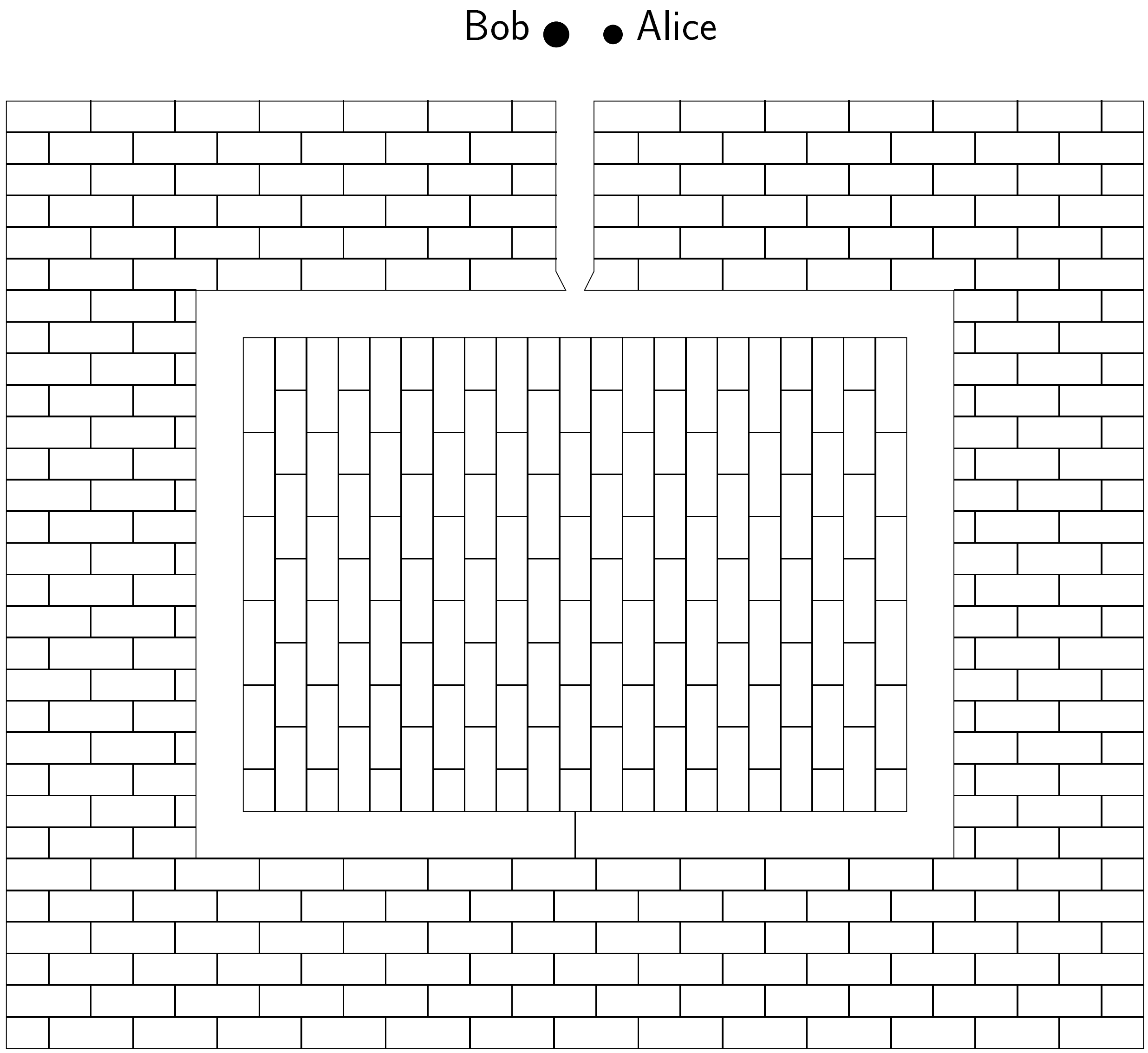
More formally: A zero-knowledge proof of a theorem T is an interactive two party protocol, in which Prover is able to convince Verifier who follows the same protocol, by the overhelming statistical evidence, that T is true, if T is indeed true, but no Prover is not able to convince Verifier that T is true, if this is not so. In additions, during interactions, Prover does not reveal to Verifier any other information, except whether T is true or not. Consequently, whatever Verifier can do after he gets convinced, he can do just believing that T is true.

Similar arguments hold for the case Prover posseses a secret.

Illustrative example

(A cave with a door opening on a secret word)

Alice knows a secret word opening the door in cave. How can she convince Bob about it without revealing this secret word?



Age difference finding protocol

Alice and Bob wants to find out who is older without disclosing any other information about their age.

The following protocol is based on a public-key cryptosystem, in which it is assumed that neither Bob nor Alice are older than 100 years.

Protocol Age of Bob: j, age of Alice: i.
1. Bob choose a random x, computes $k = e_A(x)$ and sends Alice $s = k - j$.
2. Alice first computes the numbers $y_u = d_A(s+u); 1 \leq u \leq 100$, then chooses a large random prime p and computes numbers

$$z_u = y_u \bmod p, \quad 1 \leq u \leq 100 \qquad (\star)$$

and verifies that for all $u \neq v$

$$|z_u - z_v| \geq 2 \text{ and } z_u \neq 0 \qquad .(\star\star)$$

(If this it not the case, Alice choose a new p, repeats computations in $(\star)$ and checks $(\star\star)$ again.)
Finally, Alice sends Bob the following sequence (order is important).

$$z_1, \dots, z_i, z_{i+1}+1, \dots, z_{100}+1, p$$

$$z'_1, \dots, z'_i, z'_{i+1}, \dots, z'_{100}$$

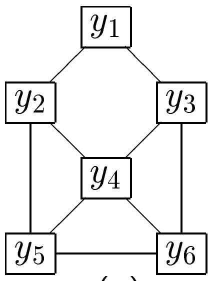
3. Bob checks whether j-th number in the above sequence is congruent to x modulo p. If yes, Bob knows that $i \geq j$, otherwise $i < j$.

$$i \geq j \Rightarrow z'_j = z_j \equiv y_j = d_A(k) \equiv x \pmod p$$

$$i < j \Rightarrow z'_j = z_j + 1 \not\equiv y_j = d_A(k) \equiv x \pmod p$$

3-COLORABILITY of GRAPHS

With the following protocol Peggy can convince Vic that a particular graph G, known to both of them, is 3-colorable and that Peggy knows such a coloring, without revealing to Vic any information how such coloring looks.



(a)

1	red	e_1	$e_1(red){=}y_1$
2	green	e_2	$e_1(green){=}y_2$
3	blue	e_3	$e_1(blue){=}y_3$
4	red	e_4	$e_1(red){=}y_4$
5	blue	e_5	$e_1(blue){=}y_5$
6	green	e_6	$e_1(green){=}y_6$

(b)

Protocol: Peggy colors the graph $G = \langle V, E \rangle$ with colors (red, blue, green) and she performs with Vic $|E|^2$-times the following interactions, where $v_1, \ldots, v_n$ are nodes of V.

1. Peggy choose a random permutation of colors, recolors G, and encrypts, for $i = 1, 2, \ldots, n$, the color c_i of node v_i by an encryption procedure e_i – for each i different.

   Peggy then removes colors from nodes, labels the i-th node of G with cryptotext $y_i = e_i(c_i)$, and designs Table (b).

   Peggy finally shows Vic the graph with nodes labeled by cryptotexts.

2. Vic chooses an edge and asks Peggy to show him coloring of the corresponding nodes.

3. Peggy shows Vic entries of the table corresponding to the nodes of the chosen edge.

4. Vic performs encryptions to verify that nodes really have colors as shown.

Zero-knowledge proofs and cryptographic protocols

The fact that for a big class of statements there are zero-knowledge proofs can be used to design secure cryptographic protocols. (All languages in $\mathbf{NP}$ have zero-knowledge.)

A cryptographic protocol can be seen as a set of interactive programs to be executed by non-trusting parties.

Each party keeps secret a local input.

The protocol specifies the actions parties should take, depending on their local secrets and previous messages exchanged.

The main problem in this setting is how can a party verify that the other parties have really followed the protocol?

The way out: a party A can convince a party B that the transmitted message was completed according to the protocol without revealing its secrets

An idea how to design a reliable protocol

1. Design a protocol under the assumption that all parties follow the protocol.
2. Transform protocol, using known methods how to make zero-knowledge proofs out of normal ones, into a protocol in which communication is based on zero-knowledge proofs, preserves both correctness and privacy and works even if some parties display an adversary behavior.

Zero-knowledge proof for quadratic residua

Input: An integer $n = pq$, where p, q are primes and $x \in QR(n)$.

Protocol: Repeat $\lg n$ times the following steps:

1. Peggy chooses a random $v \in \mathbf{Z}_n^*$ and sends to Vic

$$y = v^2 \bmod n.$$

2. Vic sends to Peggy a random $i \in \{0, 1\}$.
3. Peggy computes a square root u of x and sends to Vic

$$z = u^i v \bmod n.$$

4. Vic checks whether

$$z^2 \equiv x^i y \pmod{n}.$$

Vic accepts Peggy's proof if he succeeds in Step 4 in each of $\lg n$ rounds.

Completeness is straightforward:

Soundness. If x is not a quadratic residue, then Peggy can answer only one of two possible challenges (only if $i = 0$), because in such a case y is a quadratic residue if and only if xy is not a quadratic residue. This means that Peggy will be caught in any given round of the protocol with probability $\frac{1}{2}$.

The overall probability that prover deceives Vic is therefore $2^{-\lg n} = \frac{1}{n}$.

Zero-knowledge proof for graph isomorphism

Input: Two graphs G_1 and G_2 with the set of nodes $\{1, \ldots, n\}$.
Repeat the following steps n times:

1. Peggy chooses a random permutation π of $\{1, \ldots, n\}$ and computes H to be the image of G_1 under the permutation π, and sends H to Vic.
2. Vic chooses randomly $i \in \{1, 2\}$ and sends it to Peggy. $\{$*This way Vic asks for isomorphism between* H *and* G_i.$\}$
3. Peggy creates a permutation ρ of $\{1, \ldots, n\}$ such that ρ specifies isomorphism between H and G_i and Peggy sends ρ to Vic.

   $\{$*If* $i = 1$ *Peggy takes* $\rho = \pi$*; if* $i = 2$ *Peggy takes* $\rho = \sigma \circ \pi$*, where* σ *is a fixed isomorphic mapping of nodes of* G_2 *to* G_1.$\}$
4. Vic checks whether H provides the isomorphism between G_i and H.

Vic accepts Peggy's "proof" if H is the image of G_i in each of the n rounds.

Completeness. It is obvious that if G_1 and G_2 are isomorphic then Vic accepts with probability 1.

Soundness: If graphs G_1 and G_2 are not isomorphic, then Peggy can deceive Vic only if she is able to guess in each round the i Vic chooses and then sends as H the graph G_i. However, the probability that this happens is 2^{-n}.

Observe that Vic can perform all computations in polynomial time.
However, why is this proof a zero-knowledge proof?

Why is last "proof" a "zero-knowledge proof"?

Because Vic gets convinced, by the overwhelming statistical evidence, that graphs G_1 and G_2 are isomorphic, but he does not get any information ("knowledge") that would help him to create isomorphism between G_1 and G_2.

In each round of the proof Vic see isomorphism between H (a random isomorphic copy of G_1) and G_1 or G_2, (but not between both of them)!

However, Vic can create such random copies H of the graphs by himself and therefore it seems very unlikely that this can help Vic to find an isomorphism between G_1 and G_2.

Information that Vic can receive during the protocol, called *transcript*, contains:

- The graphs G_1 and G_2.
- All messages transmitted during communications by Peggy and Vic.
- Random numbers used by Peggy and Vic to generate their outputs.

Transcript has therefore the form

$$T = ((G_1, G_2); (H_1, i_1, \rho_1), \ldots, (H_n, i_n, \rho_n)).$$

The essential point, which is the basis for the formal definition of zero-knowledge proof, is that Vic can forge transcript, without participating in the interactive proof, that look like "real transcripts", if graphs are isomorphic, by means of the following forging algorithm called **simulator**.

SIMULATOR

A simulator for the previous graph isomorphism protocol.

- $T = (G_1, G_2)$,
- **for** $j = 1$ **to** n **do**
  - Choose randomly $i_j \in \{1, 2\}$.
  - Choose ρ_j to be a random permutation of $\{1, \ldots, n\}$.
  - Compute H_j to be the image of G_{i_j} under ρ_j;
  - Concatenate (H_j, i_j, ρ_j) at the end of T.

CONSEQUENCES and FORMAL DEFINITION

The fact that a simulator can forge transcripts has several important consequences.

- Anything Vic can compute using the information obtained from the transcript can be computed using only a forged transcript and therefore participation in such a communication does not increase Vic capability to perform any computation.
- Participation in such a proof does not allow Vic to prove isomorphism of G_1 and G_2.
- Vic cannot convince someone else that G_1 and G_2 are isomorphic by showing the transcript because it is indistinguishable from a forged one.

Formal definition what does it mean that a forged transcript "looks like" a real one:

Definition Suppose that we have an interactive proof system for a decision problem Π and a polynomial time simulator $\mathcal{S}$.

Denote by $\Gamma(x)$ the set of all possible transcripts that could be produced during the interactive proof communication for a yes-instance x.

Denote $\mathcal{F}(x)$ the set of all possible forged transcripts produced by the simulator $\mathcal{S}$.

For any transcript $T \in \Gamma(x)$, let $p_\Gamma(T)$ denote the probability that T is the transcript produced during the interactive proof. Similarly, for $T \in \mathcal{F}(x)$, let $p_\mathcal{F}(T)$ denote the probability that T is the transcript produced by S.

If $\Gamma(x) = \mathcal{F}(x)$ and, for any $T \in \Gamma(x)$, $p_\Gamma(T) = p_\mathcal{F}(T)$, then we say that the interactive proof system is a zero-knowledge proof system.

Proof for graph isomorphism protocol

Theorem The interactive proof system for *Graph isomorphism* is a perfect zero-knowledge proof if Vic follows protocol.

Proof Let G_1 and G_2 be isomorphic. A transcript (real or forged) contains triplets (H_j, i_j, ρ_j).

The set R of such triplets contains $2n!$ elements (because each pair i, ρ uniquely determines H and there are $n!$ permutation ρ.

In each round of the simulator each triplet occurs with the same probability, that is all triplets have probability $\frac{1}{(2n!)^n}$.

Let us now try to determine probability that a triplet (H, i, ρ) occurs at a jth round of the interactive proof.

i is clearly chosen with the same probability. Concerning ρ this is either randomly chosen permutation π or a composition π with a fixed permutation. Hence all triplets (H, i, ρ) have the same probability $\frac{1}{(2n!)^n}$.

The next question is whether the above graph isomorphism protocol is zero-knowledge also if Vic does not follow fully the protocol.

The case Vic does not follow protocol

It is usually much more difficult to show that an interactive proof system is zero-knowledge even if Vic does not follow the protocol.

In the case of graph isomorphism protocol the only way Vic can deviate from the protocol is that i he does not choose in a completely random way.

The way around this difficulty is to prove that, no matter how a "cheating" Vic deviates from the protocol, there exists a polynomial-time simulator that will produce forged transcripts that "look like" the transcript T of the communication produced by Peggy and (the cheating) Vic during the interactive proof.

As before, the term "looks like" is formalized by requiring that two probability distributions are the same.

Definition Suppose that we have an interactive proof system for a decision problem Π.

Let V^* be any polynomial time probabilistic algorithm that a (possibly cheating) Verifier uses to generate his challenges.

Denote by $\Gamma(V^*, x)$ the set of all possible transcripts that could be produced as a result of Peggy and V^* carrying out the interactive proof with a yes-instance x of Π.

Suppose that for every such V^* there exists an expected polynomial time probabilistic algorithm $S^* = S^*(V^*)$ (the simulator) which will produce a forged transcript.

Denote by $\mathcal{F}(V^*, x)$ the set of possible forged transcripts.

For any transcript $T \in \Gamma(V^*, x)$, let $p_{\Gamma,V^*}(T)$ denote the probability that T is the transcript produced by V^* taking part in the interactive proof.

Similarly, for $T \in \mathcal{F}(x)$, let $p_{\mathcal{F},V^*}(T)$ denote the probability that T is the (forged) transcript produced by S^*.

If $\Gamma(V^*, x) = \mathcal{F}(V^*, x)$ and for any $T \in \Gamma(V^*, x)$, $p_{\mathcal{F},V^*}(T) = p_{\Gamma,V^*}(T)$, then the interactive proof system is said to be a perfect zero-knowledge protocol.

ADDITIONS

- It can be proved that the graph isomorphism protocol is zero-knowledge even in the case Vic cheats.
- If, in an interactive proof system, the probability distributions specified by the protocols with Vic and with simulator are the same, then we speak about **perfect zero-knowledge proof system**.
- If, in an interactive proof system, the probability distributions specified by the protocols with Vic and with simulator are computationally indistinguishable in polynomial time , then we speak about **computationally zero-knowledge proof system**.