Hand-out Zero-Knowledge proofs

Joost van Amersfoort

1 What are zero-knowledge proofs?

Zero-knowledge proofs are proofs that yield nothing beyond the validity of the assertion.

An analogy [4] of such a proof can be given with help of the special cave of figure 1. Imagine you know a secret password that opens a special door that connects the two passages of the cave. Chris will give a bonus to anyone that can convince him that they know the password. However, once he knows the password, he won't give out the bonus anymore. You take Chris into the special cave, leave him at the splitting point and run into either one of the passages without him looking. Chris will scream "left" or "right" at random and that's the exit you need to come out of. The first time the probability that this happens by chance is $\frac{1}{2}$. You repeat this n times and convince Chris that the probability you are lying is only $\frac{1}{2^n}$. Since Chris learns nothing, this method could be performed any number of times, to have as many classmates as possible convincing Chris.

Figure 1: Cave

Now imagine a cave that instead of two passages has 2^n passages. Again you run into a passage at random and Chris screams a passage number at random. If you indeed come out of the right exit, then Chris is immediately convinced.

2 Interactive Proofs

In order to give the formal definition of zero-knowledge proofs, it is first necessary to introduce the notion of an interactive proof. The Prover (P) has infinite computing power, while the verifier (V) is polynomial time bounded. Both are ordinary probabilistic Turing machines that are in addition equipped with communication tapes allowing a machine to send and receive messages from the other one. L is some binary language. The prover claims that a certain statement of a certain language $x \in L$ is true. If the pair (P,V) rejects $x \in L$ with negligible probability (completeness) and accepts $x \notin L$ with negligible probability (soundness), then it is an interactive proof system.

3 Formal Definition of Zero-Knowledge Proofs

Now we have all the pieces to formally define a zero-knowledge proof [2]: fixing an interactive machine (for example the prover), we look at what can be computed by an arbitrary adversary (for example the verifier) that interacts with the fixed machine on a common input from a predetermined set S.

Now an interactive strategy A is zero-knowledge on the set S, if, for every feasible (interactive) strategy B^*, there exists a feasible (non-interactive) computation C^* s.t. the following two probability ensembles are computationally indistinguishable:

- $\{(A, B^*)(x)\}_{x \in S} \stackrel{def}{=}$ the output of B^* after interacting with A on common input $x \in S$; and
- $\{(C^*)(x)\}_{x \in S} \stackrel{def}{=}$ the output of C^* on input $x \in S$

Here the first ensemble is the execution of an interactive protocol, the second represents a stand-alone procedure ("the simulator"). This means that anything that could be extracted from A was also already in C. So nothing was gained from the interaction. This notion is called computational zero-knowledge and the

one used in practice in cryptography. Another notion is perfect zero-knowledge, where the two ensembles are exactly equal.

4 Commitment Schemes

A commitment scheme means that a player in a protocol is able to choose a value from some set and commit to his choice such that he can no longer change his mind. An informal example of such a scheme, is a game with two players P and V, where P wants to commit to a bit b. He writes b down on a piece of paper, puts it in a box and locks it using a padlock. He then passes the box to V. Now when P wants to he can pass the key to V to open the padlock. In this way P is bound to his original choice and he hides his choice until he decided to give the key.

5 Formal example

Imagine a scheme where a prover (P) wants to prove to be the owner of a public/private key pair to a verifier (V). Now V can choose a random message M, encrypt it using the public key and send the resulting ciphertext to P. P decrypts this message and sends the result M' back. If M equals M' then V accepts P's proof. The problem with this example is that it assumes V follows the protocol, while V could be asking the decryption of messages that it eavesdropped before. This can be solved by changing the protocol. Instead of sending back M', P sends a commitment message with M'. He then receives the original message M (forcing the verifier to know M). If M = M', he opens the commitment. Now the verifier accepts the identity of the prover iff the commitment is correctly opened and M' = M. [1]

This scheme forces the Verifier to behave in the correct way. In fact, it has been shown that using zero-knowledge protocols as sub-protocols it is possible to transform any protocol that is secure assuming players follow the rules into one that is secure even if players deviate from the protocol. For more information refer to [3].

6 Applications of Zero-Knowledge

The biggest impact of zero-knowledge is in the design of efficient protocols for particular problems. By for example giving the user the solution to a hard problem and the user identifies himself by providing a zero-knowledge proof that he knows this solution. An example of this is in [5], where the computation is done on a smart card and thus severely restricted.

References

[1] Ivan Damgård, *Commitment schemes and zero-knowledge protocols*, Lectures on Data Security, Springer, 1999, pp. 63–86.

[2] Oded Goldreich, *Zero-knowledge twenty years after its invention.*, IACR Cryptology ePrint Archive **2002** (2002), 186.

[3] Oded Goldreich, Silvio Micali, and Avi Wigderson, *Proofs that yield nothing but their validity and a methodology of cryptographic protocol design*, FOCS, vol. 86, 1986, pp. 174–187.

[4] Jean-Jacques Quisquater, Myriam Quisquater, Muriel Quisquater, Michaël Quisquater, Louis Guillou, Marie Annick Guillou, Gaïd Guillou, Anna Guillou, Gwenolé Guillou, and Soazig Guillou, *How to explain zero-knowledge protocols to your children*, Advances in Cryptology CRYPTO89 Proceedings, Springer, 1990, pp. 628–631.

[5] Claus-Peter Schnorr, *Efficient signature generation by smart cards*, Journal of cryptology **4** (1991), no. 3, 161–174.