Global Information Assurance Certification Paper

IDENTIFICATION WITH ZERO KNOWLEDGE PROTOCOLS

Annarita Giani

Security Essentials - Version 1.2e

Our time is characterized by the exchange of data, some of which is very sensitive, and knowledge of such information could have important consequences to our future. Let's think, for example, about the importance of knowing a credit card number or a password to access a secret database. Information exchange between two parties is essential in our world, and during this transfer, something bad could happen. For example, a third party could eavesdrop on the transmission, and then use the data in some malicious way for personal advantage. The so-called Mafia fraud consists of intercepting electronic payment messages, and then using that information to buy something very expensive.

Modern cryptography is based on the secrecy of the key. In Secret-Key Cryptosystems, two parties have to meet and agree on a common secret key before any communication can happen. In Public-key Cryptosystems, each party has a pair of keys, one published in a database available to everybody and the other kept secret. This scheme eliminates the need for a preliminary secure interaction between the two parties. The strength of this scheme rests on the limited computational resources available to each user, legitimate or malicious. The main idea in any public key cryptosystem is a difficult computational problem. The security is based on the fact that the private key can be computed from the public key only by solving this difficult problem. With the public key, a user could encrypt messages, and another could decrypt them with the private key. The owner of the private key would be the only one who could decrypt the messages, but anyone knowing the public key could send them in privacy.

The idea of proving knowledge of some assertion without revealing any information about the assertion itself is very attractive. Zero-Knowledge protocols allow this kind of scenario. They are cryptographic protocols that do not reveal the secret itself during operations, since the secret is not transferred to the other party, but the user still is able to prove to the other party that he knows the secret. This approach can be a good solution for proving mutual identity, or, for example, in the key-exchange step of a cryptographic application.

In Zero Knowledge protocols, a Prover tries to prove knowledge of a secret to a Verifier without revealing the secret itself. The Verifier can ask questions with the goal of finding out if the Prover really knows the secret, but it is impossible for him to discover information about the secret even if he doesn't follow the rules of the protocol. An Eavesdropper is a third party that listens to the conversation but, if the protocol is secure, he is not able to learn anything about the secret, or convince somebody else that he knows the secret. There is also a Malicious user able to send, modify or destroy messages. A good protocol should be resistant against this user. In the literature, these different parties are called Peggy, Victor, Eve and Maggie for obvious reasons. A protocol has to take into account that both Peggy and Victor may have malicious intentions as well. Peggy might try to cheat Victor into accepting a false statement, and Victor might try to get information to use in the future for personal advantage. A good scheme must be built in such a way that:

1. If Peggy does not know the secret information, she is not able to pretend to have such knowledge. Many rounds of the scheme should guarantee that (with probability close to 1) she couldn't cheat Victor.

2. Victor is able to convince himself that Peggy knows the secret, but he is not able to get any further information, which, for example, could allow him to convince somebody else that he knows the secret. In particular he cannot learn anything from the protocol that he could not learn without asking Peggy direct questions. From this concept comes the name of this approach.

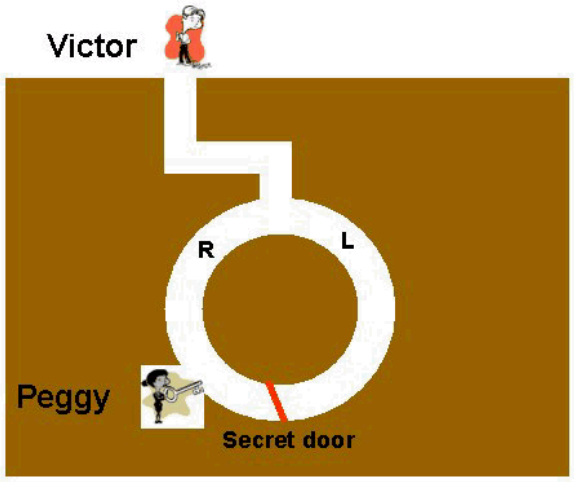
Let's see an example [5] of using a Zero-Knowledge protocol to prove an assertion. How can I convince you that I am able to count the leaves of a big maple tree in a few seconds without revealing to you my method to do it and without revealing to you the number of the leaves? Ok, let's go in front of the tree. A round of the protocol is:

- While I close my eyes, you can do either of the following actions, pull off a leaf or do nothing.

- Then I take a look at the tree, and you ask me to tell you what you did.

If I give you the wrong answer, you immediately say that I lied about my ability to count the leaves in a few seconds, but what if I give you the right answer? You can think that I was just lucky! Yes, but you can repeat the steps as many times as you like. What about if I give you the right answer for, let's say, 1,000 times. Aren't you sure yet? You can repeat the operation of the protocol 10,000 or 30,000 times. Maybe now you start to think that in some way, even if you don't know how, I am able to count the leaves in a tree. After 100,000 rounds, you should be quite sure of my knowledge.

In another classic example [5] Peggy wants to prove to Victor that she knows the password to open a secret door inside a cave without revealing the password itself. The scenario is shown in the picture below.



Victor stands outside while Peggy goes into a branch of the cave (Victor doesn't know which one). Then Victor goes inside the cave and asks Peggy to come out from a particular branch that he chooses at random. It is clear that if Peggy knows the secret key to open the door, she is able to come out from the branch that Victor called out, but if she doesn't know the password, she has 50% probability of coming out from the wrong branch. Victor can easily tell that she does not have the knowledge she claims to have. Victor could be very difficult to convince, so he could require many repetitions of these steps but, if, after 100 times, Peggy comes out the right way and doesn't fail once, he could be "sure" that she knows the password to open the door. In fact, she has a probability of 0.5^100 to cheat Victor, and this number is close to zero. Victor can repeat the round as many times as he desires until he is certain that Peggy knows the secret word. An important point here is that even if Victor is completely convinced that Peggy knows the secret word, with the information he receives, he is not able to convince anybody else. Let's say that he videotapes everything. A third party could think he agreed with Peggy about which branch to choose each time. This attests that Victor cannot use any information he got for his own purposes.

If Peggy wants to prove to Victor that she knows how to solve the Rubik's cube, but she does not want to reveal how she solves it, she could use Zero Knowledge theory. Victor shows Peggy a messed up cube A and wants her to solve it. Peggy shows Victor a new scrambled cube B. Victor can ask Peggy

- To move from the new cube B to the old one A, or
- To move from the new cube B to the solved position.

If Peggy knows how to obtain B from A, she can answer the first question or the second one. Peggy can solve the cube if she knows both parts of the solution. She must show a different messed up cube each round to prevent Victor from learning how to solve the problem [5].

Proof of identity

Zero Knowledge proofs can be used for identification. First we discuss identification schemes in general, then "traditional" secret-key and public-key schemes, and finally zero-knowledge schemes. Identification schemes are methods by which a user may prove his or her identity without revealing knowledge that may be used by an eavesdropper to impersonate the user. The traditional form of identification is by use of a secret key, password or pin, but this scheme is extremely insecure since they are easy to guess, for example, through an exhaustive search. Recently, biometric parameters like fingerprints, retinal scans or facial recognition are used, but they are not comfortable, and they give value to body parts, which can have many disadvantages. Another common scheme is using digital signatures and public-key cryptography. An identification scheme consists essentially of two separate stages

- Initialization
- Operation

In public-key identification schemes, during the first stage each user generates a private key and a public key. A Trusted Authority is required to ensure the link between each user and his public key. At the end of the operation stage the verifier can accept or reject the identification.

The problem of studying and building identification schemes comes from the need of proving the identity of somebody without any doubt, and from preventing any user but the Prover to identify thimself as the Prover. Any identification scheme must be:

> *Complete* in the sense that if the user who tries to identify himself follows the protocol, then the identification is surely successful.
>
> *Soundness* in the sense that nobody can identify himself as somebody else.

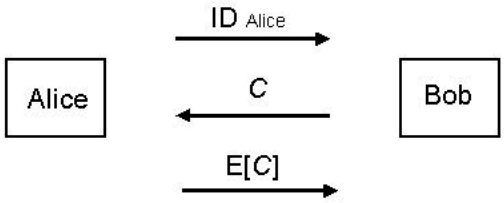
Moreover, it is important to highlight that the protocol should work with low memory, since the person who wishes to be identified could have low memory, and can perform calculations of only limited complexity. The presence of a Trusted Authority that certifies the identity of the Prover is essential. It uses information inside some identity documents like Passport or SSN to generate a string, an ID, which it signs to create a certificate.

The request of a certificate helps to prevent some types of attacks, for example, a man-in-the-middle attack. Let's give an example of this kind of attack. If a malicious user (Michael) wants to persuade somebody that he is a very good chess player, he could set up chess games with the two most famous chess players in the world, Ann and Bob, to play against him. He could place each player in a different room, let them play, and repeat each move to the other player. Ann and Bob think they are playing against Michael, but they are actually playing against each other. A countermeasure to this attack could be imposing time limits for the replies. Using only a zero-knowledge proof of identity over a communications channel is not sufficient to provide a secure communication channel. A malicious user with access to the Internet may wait until the authentication protocol completes and then hijack your connection.

Secret Key identification Scheme

A and B share a secret key, and Peggy wants to identify herself to Victor.

1. A sends to B her ID_{Alice},
2. B sends to A a challenge, a number "*c*",
3. A encrypts *c* and sends back to B E[*c*],
4. B decrypts E[c], and if he obtains *c*, he is sure to be talking with A.



This protocol is complete, because if both A and B are sincere, A succeeds in identifying herself to B. The soundness depends on the algorithms used during the phases of encryption and decryption. If they are not strong and allow some kind of attack, somebody else could pretend to be A and even convince B. The problem with this secret-key is that Peggy is able to identify herself to the people who share a secret key with her, and if the number of people becomes high, she will have a complex key management task.

Public Key identification Scheme

In a public key identification scheme, A has a pair of keys, one kept secret and the other available to everybody. The Schnoor identification scheme requires a Trusted Authority to choose the parameters to use during the protocol between A and B. I don't want to go in the mathematical details of the different steps, but it is important to say that it is based on how difficult it is to solve some mathematical problems. The power of zero-knowledge protocols is due to some very hard mathematical problems like the discrete logarithm for large numbers, the factorization of numbers, products of large primes, or the determination if a number is a square mod *n* without knowing the factors of *n*. A computationally light protocol is not secure because A could easily calculate the right answer. Moreover a third party might be able to impersonate A and convince B. Replacing B with a hash function we can use the Schnoor identification Scheme for digital signatures. In this case there is no interaction between A and B. A computes a Hash function and then sends it to B, and B can verify it.

The Guillou-Quisquater identification scheme is based on the RSA. Similar to the above scheme, it requires a Trusted Authority to choose the parameters to use.

Identification Proof based on Zero Knowledge theory.

The idea behind this kind of authentication is to associate with each person something unique, so that if the person can prove to have such information, he can identify himself. Let's

consider two parties, Peggy and Victor. If Peggy is able to convince Victor that something is true without giving any information about the proof, we say that Peggy is using a Zero Knowledge proof. These kinds of proofs use computationally complex problems. In the case of identification, Peggy, who wants to identify herself to Victor, chooses a computationally difficult problem and using Zero-Knowledge, proves to Victor that she knows the solution to the problem without revealing to him any further information about the problem.

A Zero-knowledge protocol is the "**Fiat-Shamir Proof of Identity**" [2] based on the difficulty of factoring. Users can prove identity or authenticity of messages without shared or public keys. These kinds of schemes are simple, secure and speedy, and for this particular reason, they are suited very well, for example, for smart cards. There are different levels:

> *Authentication Scheme*: Alice proves to Bob that she is Alice, but someone else cannot prove to Bob that they are Alice.
>
> *Identification Schemes*: Alice proves to Bob that she is Alice, but Bob cannot prove to someone else that he is Alice.
>
> *Signature Schemes*: Alice proves to Bob that she is Alice, but Bob cannot prove to anybody (himself included) that he is Alice.

There is just a subtle difference between Identification Schemes and Signature Schemes: in the first case, Bob could generate a credible transcript of an imaginary communication; in the second, only a real dialog with Alice would allow Bob to create a credible transcript. If transcripts are irrelevant, as in cases when the main problem is to detect forgery in real time, Identification Schemes and Signature Schemes are basically the same.

The Scheme that Fiat and Shamir proposed is based on the difficulty of extracting modular square roots when the factorization of a number n is unknown. They suppose that there is a trusted organization that checks the identity of someone and in the positive case, issues for, example, smart cards. In this type of identification, there is no need for a user database storing, for example, public keys. Another advantage is that there is no limit on the number of users that can participate in the systems.

Smart cards or other mini-processor based devices, can be used an unlimited number of times since interaction with these devices does not enable verifiers to reproduce or modify identities.

The organization chooses a number n and makes it public, together with a function f, which maps a string to the interval $[0, n)$. Everybody can use the same n because only the center knows how it can be factored. Unlike RSA, it is not the product of two primes, p and q. When a user applies for a smart card, the center creates a string with important information about the user and the card. At the end of the procedure, it issues a smart card. Note that the only information stored in each verification device is the universal number n and the function f.

A variation of the above scheme is the "Fiege-Fiat-Shamir Proof of Identity" [1] based on thedifficulty of extracting square roots modulo large composite integers n of unknown

factorization, which is equivalent to factoring *n*.

Another improvement to the Fiat and Shamir protocol is the "Guillou and Quisquater protocol" [3] that reduces cheating probability. The main problems in the Fiat and Shamir protocol is in the number of iterations between Alice and Bob, the Prover and the Verifier, and in the memory Alice needs. Guillou and Quisquater, with longer computations, optimizes the Fiat and Shamir protocol.

As we have seen a zero knowledge proof is a way to prove that you know a secret without revealing the secret. A practical scenario is proving that a public key is good without revealing the private key. This technique also can be used for identification. A user proves to somebody else who he is without giving him any information that he can use later to pretend to be the first person. We have seen some examples of classical identification scheme and some new techniques using zero knowledge theory.

References:

[1] U. Fiege, A Fiat, A Shamir, Zero-knowledge proofs of Identity, Journal of Cryptology, vol. 1, n. 2, 1988, pp. 77-94.

[2] A. Fiat, A. Shamir, How to prove yourself: Practical solutions to identification and signature problems, proceeding of CRYPTO '86, Lecture Notes in Computer Science, vol. 263, Springer-Verlang, Berlin, 1987, pp. 186-194.

[3] L. C. Guillou, J.J. Quisquater, A practical zero-knowledge protocol fitted to security microprocessor minimizing both trasmission and memory, Lecture Notes in Computer Science, vol. 330, Advances in Cryptology: Proceeding Eurocrypt '88, C. G. Gunthet, Ed., Davos, Switzerland, May 25-27, 1988, pp. 123-128, Springer-Verlang, Berlin, 1988.

[4] Schneier, Bruce. Secrets and Lies: Digital Security in a Networked World John Wiley, 2000

[5] Zero Knowledge Protocols and Small Systems
http://www.tml.hut.fi/Opinnot/Tik-110.501/1995/zeroknowledge.html

[6] Interactive Proofs & Zero Knowledge
http://www.msri.org/publications/ln/msri/2000/crypto/dwork/3/index.html

[7] What are Interactive Proofs and Zero-Knowledge Proofs?
http://www.x5.net/faqs/crypto/q107.html

[8] Resettable Zero Knowledge with Applications to Identification
http://www.lix.polytechnique.fr/Labo/Bernadette.Charron/Slidescirm/Goldwasser/index.htm

[9] Identification and entity authentication
http://www.cs.huji.ac.il/~security/Slides/ident/index.htm