Handout 6 (Zero-Knowledge Proofs)

Zero-knowledge proofs (short ZKP) solve a paradoxical puzzle: How to convince somebody else that one knows a secret, without revealing what the secret actually is? This sounds like a problem the Mad Hatter from Alice in Wonderland would occupy himself with, but actually there some serious and not so serious applications of it. For example, if you solve crosswords with your friend, say Bob, you might want to convince him that you found a solution for one question, but of course you do not want to reveal the solution, as this might give Bob an advantage somewhere else in the crossword. So how to convince Bob that you know the answer (or a secret)? One way would be to come up with the following protocol: Suppose the answer is *folio*. Then look up the definition of *folio* in a dictionary. Say you find:

> "an *individual* leaf of paper or parchment, either loose as one of a series or forming part of a bound volume, which is numbered on the recto or front side only."

Take the first non-article word in this definition, in this case *individual*, and look up the definition of this word, say

> "a single *human* being as distinct from a group"

In this definition take the second non-article word, that is *human*, and again look up the definition of this word. This will yield

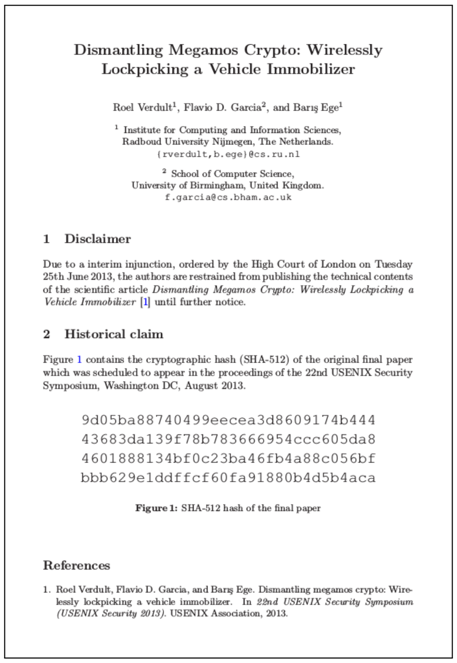
> "relating to *or* characteristic of humankind"

You could go on to look up the definition of the third non-article in this definition and so on. But let us assume you agreed with Bob to stop after three iterations with the third non-article word in the last definition, that is **or**. Now, instead of sending to Bob the solution *folio*, you send to him *or*.

How can Bob verify that you know the solution? Well, once he solved it himself, he can use the dictionary and follow the same "trail" as you did. If the final word agrees with what you send him, he must infer you knew the solution earlier than him. This protocol works like a one-way hash function because *or* does not give any hint as to what was the first word was. I leave you to think why this protocol avoids article words.

After Bob found his solution and verified that according to the protocol it "maps" also to *or*, can he be entirely sure no cheating is going on? Not with 100% certainty. It could have been entirely possible that he was given *or* as the word, but it derived from an entirely different word. This might seem very unlikely, but at least theoretical is a possibility. Protocols based on zero-knowledge proofs will produce a similar result—they give an answer that might be erroneous in a very small number of cases. The point is to iterate them long enough so that the theoretical possibility of cheating is negligibly small.

By the way, the authors of the paper "Dismantling Megamos Crypto: Wirelessly Lockpicking a Vehicle Immobilizer" who were barred from publishing



Dismantling Megamos Crypto: Wirelessly Lockpicking a Vehicle Immobilizer

Roel Verdult[1], Flavio D. Garcia[2], and Barış Ege[1]

[1] Institute for Computing and Information Sciences,
Radboud University Nijmegen, The Netherlands.
{rverdult,b.ege}@cs.ru.nl

[2] School of Computer Science,
University of Birmingham, United Kingdom.
f.garcia@cs.bham.ac.uk

1 Disclaimer

Due to a interim injunction, ordered by the High Court of London on Tuesday 25th June 2013, the authors are restrained from publishing the technical contents of the scientific article *Dismantling Megamos Crypto: Wirelessly Lockpicking a Vehicle Immobilizer* [1] until further notice.

2 Historical claim

Figure 1 contains the cryptographic hash (SHA-512) of the original final paper which was scheduled to appear in the proceedings of the 22nd USENIX Security Symposium, Washington DC, August 2013.

```
9d05ba88740499eecea3d8609174b444
43683da139f78b783666954ccc605da8
4601888134bf0c23ba46fb4a88c056bf
bbb629e1ddffcf60fa91880b4d5b4aca
```

Figure 1: SHA-512 hash of the final paper

References

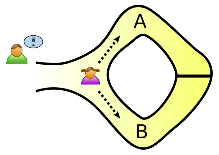
1. Roel Verdult, Flavio D. Garcia, and Barış Ege. Dismantling megamos crypto: Wirelessly lockpicking a vehicle immobilizer. In *22nd USENIX Security Symposium (USENIX Security 2013)*. USENIX Association, 2013.

Figure 1: The authors of this paper used a hash in order to prove later that they have managed to break into cars.

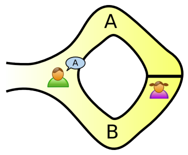
their results used also a hash to prove they did the work and (presumably) managed to get into cars without a key; see Figure 1. This is very similar to the method about crosswords: They like to prove that they did the work, but not giving out the "solution". But this also shows what the problem with such a method is: yes, we can hide the secret temporarily, but if somebody else wants to verify it, then the secret has to be made public. Bob needs to know that *folio* is the solution before he can verify the claim that somebody else had the solution first. Similarly with the paper: we need to wait until the authors are finally allowed to publish their findings in order to verify the hash. This might happen at some point, but equally it might never happen (what for example happens if the authors lose their copy of the paper because of a disk failure?). Zero-knowledge proofs, in contrast, can be immediately be checked, even if the secret is not public yet and never will be.

ZKP: An Illustrative Example

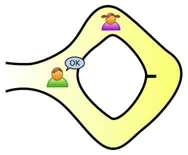
The idea behind zero-knowledge proofs is not very obvious and will surely take some time for you to digest. Therefore let us start with a simple illustrative example. The example will not be perfect, but hopefully explain the gist of the idea. The example is called Alibaba's cave, which graphically looks as follows:



Step 1



Step 2



Step 3

Let us have a look at the picture in Step 1: The cave has a tunnel which forks at some point. Both forks are connected in a loop. At the deep end of the loop is a magic wand. The point of the magic wand is that Alice knows the secret word for how to open it. She wants to keep the word secret, but wants to convince Bob that she knows it.

One way of course would be to let Bob follow her, but then he would also find out the secret. This does not work. So let us first fix the rules: At the beginning Alice and Bob are outside the cave. Alice goes in alone and takes either tunnel labelled *A* in the picture, or the other one labelled *B*. She waits at the magic wand for the instructions from Bob, who also goes into the gave and observes what happens at the fork. He has no knowledge which tunnel Alice took and calls out (Step 2) that she should emerge from tunnel *A*. If she knows the problem, this will not be a problem for Alice. If she was already in the A-segment of the tunnel, then she just comes back. If she was in the B-segment of the tunnel she will say the magic work and goes through the want to emerge from *A* as requested by Bob.

Let us have a look at the case where Alice cheats, that is not knows the secret. She would still go into the cave and use, for example the *B*-segment of the tunnel. If now Bob says she should emerge from *B*, she was lucky. But if he says she should emerge from *A* then Alice is in trouble and Bob will find out she does not know the secret. So in order to fool Bob she needs a protocol that anticipate his call, and already go into this tunnel.

Using an Graph-Isomorphism Problem for ZKPs