

Zero Knowledge Protocols

Ian Stewart

30/09/2008

**Mathematics Institute
University of Warwick
Coventry CV4 7AL
UK**

In the age of the Internet it has become important to be able to send messages that convey certain facts to their intended recipient without inadvertently revealing other facts, to them or to anybody else. For instance, suppose that you want to pay for a purchase by credit card. Transmitting your credit card number on its own is not a wise method. In order for this bare message to work, the recipient must transfer money whenever a valid credit card number is received. But then somebody could intercept your number, or even set up an illicit computer program that collects credit card numbers, and buy other goods using your account.

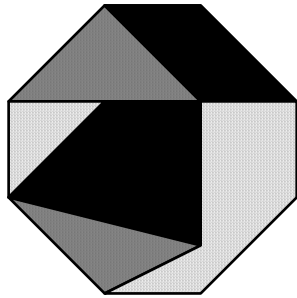
Using a simple PIN number doesn't add much security, because that too must be transmitted over the net. Most security systems use some kind of encryption method to confirm that the message is from a legitimate source. Such systems work if the code is a secure one, and nowadays there are plenty of good ideas for secure codes. In fact, some codes are so secure that law-enforcement agencies want them banned, because they would let criminals send messages that could not be understood even if they were intercepted.

An alternative approach is to use a 'zero knowledge protocol'. This is a way of convincing the recipient that you are in possession of some key item of information (such as a PIN) without revealing what that item actually is. The surprise is that such protocols can exist at all, but in recent years cryptographers have devised them in large numbers.

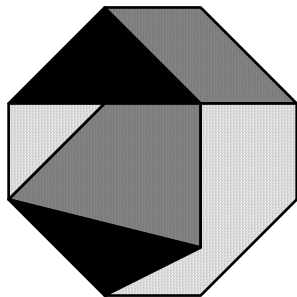
The kind of principle that is involved can be illustrated most simply in the context of map-coloring rather than PINs. The celebrated Four Color Theorem was first conjectured in 1852 by Francis Guthrie, a graduate student at University College London. It was proved in 1976 by Kenneth Appel and Wolfgang Haken of the University of Illinois, and it states that every map in the plane can be colored so that no two adjacent countries have the same color, and no more than four colors are used. However, if we are limited to only three colors, then some maps can be colored and some cannot.

Suppose that your bank manager sends you an exceedingly complex map, and you wish to convince her that you know how to three-color it *without* revealing which regions have which colors. Then you can construct an elaborate electronic device, linked to and controlled by two touch-sensitive screens, one in the bank, one at your home. This device is configured to do the following, and *only* the following, things. (See **Fig.1**).

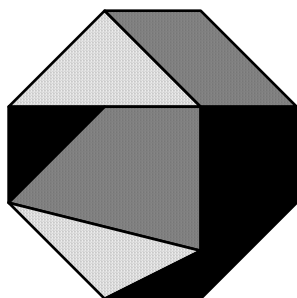
YOU



original coloring

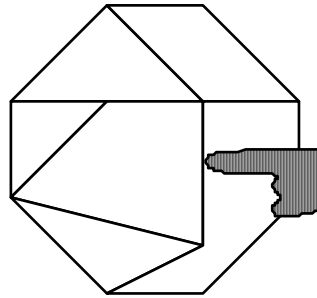


permuted coloring

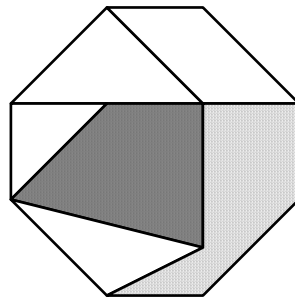


permuted coloring

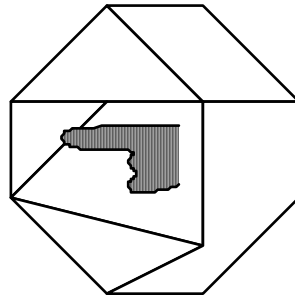
BANK MANAGER



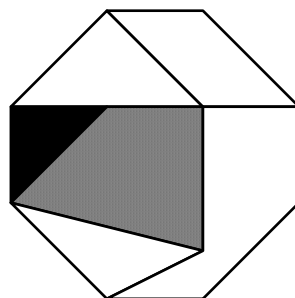
choose a border



check (permuted) colors at that border are different



choose another border



check (permuted) colors at that border are different

Convincing your bank manager that you can three-color a map. Repeat until all borders have been selected.

First, you program into the machine your map-coloring (say by touching regions of your screen — one touch for red, two for blue, three for yellow).

Next, the bank manager selects a border where two countries meet. The machine performs a random permutation of your coloring scheme — for example, systematically replacing your red color by blue, your blue by red, and leaving yellow unchanged. There are six possible ways to permute the colors, and your bank manager does not know which permutation the machine has selected. Then the manager's screen displays the *new* colors of the two countries adjacent to the selected border, all other countries being left uncolored. If your original coloring was a valid one, then these two colors should be different.

The manager then repeats the same operations until every border has been tested, and can then determine whether your claim to have three-colored the map is correct. In fact, if your original coloring is faulty, with two adjacent countries having the same color, then at some stage the bank manager will select their common border and the two permuted colors revealed by the machine will be identical. If, on the other hand, the two permuted colors are different for every border, then your original map must be a valid one.

However, because the permutations are random, there is no way for the manager to deduce your original coloring. The machine's responses merely confirm that various pairs of adjacent countries have different colors on your map: they don't tell her what the colors are.

Workers in the field of zero knowledge protocols prefer a more rigorous argument based on the idea of 'simulation'. Imagine a superficially identical set-up, in which the machine's responses are not determined by your choice of map, but by choosing two different colors at random and putting them on the screen. This fake system might produce many different sequences of pairs of colors, but one of the possibilities is the actual sequence of responses based on your map. Suppose for a moment that your bank manager could determine your map from the real machine's responses. Then she could also determine your map on that rare occasion when the fake machine produced the same responses. But for the fake machine, there is no such thing as 'your map', so such a deduction must be impossible.

Observe now that if your bank manager cannot deduce your three-coloring from the machine's responses, then neither can an illegitimate observer.

A more elaborate zero knowledge protocol allows you to convince your bank manager that you know the two prime factors p and q of a particular number $n = pq$, without revealing what they are. Provided n is fairly large — a typical size is around 200 digits — then there is no known algorithm that will find the factors p and q within the lifetime of the universe. However, there are very quick algorithms to test p and q to ensure that they are prime.

So your bank manager can cook up two primes p and q , work out $n = pq$, and treat p and q as a kind of PIN (which you are told when you open your account). Over a suitable communication channel you can then convince her that you know this PIN,

without divulging p and q to her or any eavesdropper. The method involves a certain amount of number theory (see BOX) and requires a further technique known as 'oblivious transfer'.

An oblivious transfer channel lets you send your bank manager two encrypted items of information, in such a manner that (a) she can decipher and read exactly one of the items, (b) you don't know which item she can read, and (c) you are both convinced that (a) and (b) are true. Subject to a few plausible conjectures, there are simple number-theoretic ways to construct an oblivious transfer channel, but I won't describe them here. (For more details see Neal Koblitz, *A Course in Number Theory and Cryptography*, Springer-Verlag 1994.)

This method does require a certain amount of preparation, and your usual 4-digit PIN is replaced by two 100-digit numbers, upon which you must carry out quite a lot of arithmetic, flawlessly. However, any lap-top is more than capable of such a task. There are more practical methods than the one I've just outlined, but they aren't as simple to describe. What is clear is that in an age of digital communications, security systems must be *provably* secure: experiment alone is not sufficiently convincing. And once you start asking for proofs, you're talking mathematics.

=====

BOX Proving knowledge of prime factors by oblivious transfer.

You both know a number n which is a product $n = pq$ of two primes p and q , and you both know p and q . A trusted, independent source supplies you both with a sequence of random bits 0 or 1 from which you can construct any random numbers required in the protocol. You can convince your bank manager that you know p and q , without revealing what they are, as follows.

1. The independent source generates a random integer x , and sends you and your manager the remainder r on dividing x^2 by n (that is, $r = x^2 \bmod n$).
2. According to number theory, r has exactly *four* different square roots modulo n . You use your knowledge of p and q to find them. One of them is x , and the other three are $n-x$, y , and $n-y$, for some y . (If you don't know p and q , there is no efficient algorithm to find these square roots; indeed, if you know all four then you can easily deduce p and q .)
3. You choose one of these four numbers at random: call it z .
4. You choose a random integer k and send your manager the integer $s = k^2 \bmod n$. You then set $a = k \bmod n$, $b = kz \bmod n$, and send these two numbers to your bank manager by oblivious transfer.
5. The manager can read exactly one of the two messages. She checks that its square mod n is either s (if she reads message a) or rs (if she reads message b).

6. These steps are repeated T times. At the end of this, your manager is convinced (with probability $1-2^{-T}$) that you know the factorization.

Notice that there is no communication from your bank manager back to you; that is, the protocol is not interactive.

=====