An implementation of ZKPS authentication scheme of Fiat-Shamir

Eray Özkural

January 28, 2017

This program implements a cryptographic network authentication scheme based on the work of Fiat-Shamir. The system is described in Section 6.8 of Network Security: Private Communication in a Public World by Kaufman et al. Originally, I intended to implement the graph isomorphism based zero knowledge proof system. However, this system is regarded a little inefficient.

Generally, the advantage of a zero knowledge proof system is that one can prove he knows a secret without revealing it. As one may have noticed, this is also the basic idea in public-key cryptography where the computational load of not knowing the secret determines the strength of the crypto-system. The Fiat-Shamir scheme is essentially a public-key cryptographic authentication scheme. Alice chooses a public key and a secret key, and Alice can prove her identity only through her public key she shares with Bob. No other information is required to achieve the authentication and an eavesdropper cannot replay an authentication session.

Although it is described in detail in the textbook, I will describe the scheme once again here for sake of completeness. Just like in RSA, Alice first chooses two large random primes p and q. Let $n = pq$. Alice also picks a secret s. Let $v = s^2 \bmod n$. Alice's public key is (n, v) and her secret key is s. The factors p and q can be deleted after key computation.

The protocol is very simple in essence and as we will see has a challenge-response conversation typical in zero knowledge proof systems. Here is the protocol initiated by Alice to prove her identity to Bob :

1. Alice chooses k random numbers $r_1, r_2, ..., r_k$. For each r_i, she sends $r_i^2 \bmod n$ to Bob

2. Bob chooses a random subset of the r_i^2 and tells Alice which subset he has selected to be known as subset 1. The others will be known as subset 2

3. Alice sends sr_i mod n for each r_i^2 of subset 1, and sends r_i mod n for each r_i^2 of subset 2

4. Bob squares Alice's replies mod n. For those r_i^2 in subset 1 he checks that the square of the reply is vr_i^2 mod n. For those r_i^2 in subset 2, he checks that the square of the reply is r_i^2 mod n.

The protocol's security depends on the fact that square root mod n is at least as hard as factoring n. That is, given an efficient square root mod n algorithm one can factor n efficiently. The problem transformation is provided in the textbook.

Let Trudy be an eavesdropper who wishes to impersonate Alice. By overhearing the conversation she will have collected pairs $< r_i^2, sr_i >$. However, as can be verified from the protocol, she cannot have r_i. Therefore, she cannot at the same time compute both $sr_i mod n$ and $r_i mod n$ for Bob chooses the partition of random numbers, randomly. Trudy's chance in getting the answer right is 50% for each challenge. For this reason the probability that it she will succeed in impersonation is only 2^{-k}, which is enormously low for typical values of k. In the textbook a value of 30 is being given as example.

The Fiat-Shamir scheme has the added advantage of being as secure as RSA, while requiring much less modular operations on big numbers. A comparison of typical scenarios in the textbook show that the work for Alice is about the same while the work for Alice is hundred times less.

The implementation was done in the high level efficient functional language ocaml. I've used bits of cryptokit by Xavier Leroy to implement such things as generating random primes. The authentication occurs over the network. The client and server program work not unlike *ssh* or *gpg*. I haven't built an application on the authentication scheme. However, the UNIX CLIs allow for key generation and then authentication over TCP/IP.