Zero-knowledge and Interactive Proofs

Emmanuel Ekunsumi

COMP 4140

Instructor: Michael Zapp

December 9, 2015

Table of Contents

Abstract

In this paper I describe the zero-knowledge interactive scheme called the Feige-Fiat-Shamir Identification Scheme which enables any user to demonstrate the possession of knowledge without revealing any computational information whatsoever. The scheme is provably secure against any known or chosen plaintext attack if factoring is difficult, and a typical implementation requires only 1% to 4% of the number of modular multiplications required by the RSA scheme. Although it is susceptible to man-in-the-middle attacks, I show that the probability of an adversary successfully posing as a sender is negligible. Due to its simplicity, security and speed, the scheme is ideally suited for microprocessor-based devices such as smart cards, personal computers, and remote control systems.

1. Introduction

The Feige–Fiat–Shamir identification scheme is a type of zero-knowledge proof that proposed by Fiat and Shamir in 1987 and modified a year later by Feige, Fiat and Shamir. This is the best-known zero-knowledge proof of identity. This scheme was suggested for use in the account holder–cash machine interaction. For a brief time the U.S. Army tried to have their paper classified as top secret (and failed).

In the scheme, a trusted center publishes a modulus n which is the product of two large primes of the form 4r + 3. n is a Blum integer, and hence it has the property that -1 is a quadratic nonresidue (i.e. $x^2 = -1 \bmod n$ has no solution) and its Jacobi symbol is 1 mod n. The difficulty of extracting square roots mod n

makes the private key infeasible to guess. Once the modulus ✂ has been published, the center can be closed as it has no further role in the protocol.

In this paper I first explain what interactive proofs are and how they are related to zero-knowledge proofs. Then I go on to explain the first version of the Fiat-Shamir protocol, the more general version and then the Feige-Fiat-Shamir version. I present the security risks of the earlier version of the Fiat-Shamir protocol. I also show how the problem was solved in the latest version of the protocol. Then, I will show that the probability of an adversary successfully posing as the prover in the latest is negligible. Finally, I would present my implementation of the Feige–Fiat–Shamir identification scheme in detail.

2. Interactive Proofs

An interactive proof system for a language L is a two-party game between a verifier and a prover that interact on a common input in a way satisfying the following properties:

a. The verifier strategy is a probabilistic polynomial-time procedure (where time is measured in terms of the length of the common input)

b. Correctness requirements:

- **Completeness**: There exists a prover strategy P, such that for every x ∈ L, when interacting on the common input x, the prover P convinces the verifier with probability at least 2/3.
- **Soundness**: For a false assertion, no convincing proof strategy exists (in the case of NP, if x ∋ L then no witness y exists).

It is assumed that the verifier is always honest.

The specific nature of the system, and so the complexity class of languages it can recognize, depends on what sort of bounds are put on the verifier, as well as what abilities it is given — for example, most interactive proof systems depend critically on the verifier's ability to make random choices. It also depends on the nature of the messages exchanged — how many and what they can contain. Interactive proof systems have been found to have some important implications for traditional complexity classes defined using only one machine.

The complexity class NP may be viewed as a very simple proof system. In this system, the verifier is a deterministic, polynomial-time machine (a P machine). The protocol is:

a. The prover looks at the input and computes the solution using its unlimited power and returns a polynomial-size proof certificate.
b. The verifier verifies that the certificate is valid in deterministic polynomial time. If it is valid, it accepts; otherwise, it rejects.

In the case where a valid proof certificate exists, the prover is always able to make the verifier accept by giving it that certificate. In the case where there is no valid proof certificate, however, the input is not in the language, and no prover, however malicious it is, can convince the verifier otherwise, because any proof certificate will be rejected.

3. Zero-Knowledge Proofs

Not only can interactive proof systems solve problems not believed to be in NP, but under assumptions about the existence of one-way functions, a prover can convince the verifier of the solution without ever giving the verifier information about the solution. This is important when the verifier cannot be trusted with the full solution. At first it seems impossible that the verifier could be convinced that there is a solution when the verifier has not seen a certificate, but such proofs, known as zero-knowledge proofs are in fact believed to exist for all problems in NP and are valuable in cryptography. Zero-knowledge proofs were first mentioned in the original 1985 paper on IP by Goldwasser, Micali and Rackoff, but the extent of their power was shown by Oded Goldreich, Silvio Micali and Avi Wigderson.

3.1. Definition of Zero-Knowledge proofs

Zero-knowledge interactive proofs, introduced by Goldwasser, Micali and Rackoff are paradoxical constructions allowing one player (called the prover) to convince another player (called the verifier) of the validity of a mathematical statement $x \in L$, while providing no additional knowledge to the verifier. An interactive proof is called zero-knowledge if on input $x \in L$ no probabilistic polynomial-time verifier (i.e. one that may arbitrarily deviate from the predetermined program) gains information from the execution of the protocol, except the knowledge that x belongs to L. This means that any polynomial-time computation based on the conversation with the prover can be simulated, without interacting with the real prover, by a probabilistic

polynomial-time machine ("the simulator") that gets x as its only input. Table I shows the technical definition of Zero-knowledge proofs.

TABLE I. TECHNICAL DEFINITION OF ZERO-KNOWLEDGE PROOFS

Properties	Assumption		Action done
	Verifier is honest	Prover is honest	
Completeness	Yes	Yes	Verifier will always accept a proof from the prover.
Soundness	Yes	No	Verifier will accept any "incorrect" proof from the prover with negligible probability 2^{-k}.
Zero-knowledge	No	Yes	Verifier will learn nothing about the prover's secret.

T
he first two of these are properties of more general interactive proof systems. The third is what makes the proof zero-knowledge.

a. Soundness Error

The soundness error of interactive zero-knowledge has negligible probability as shown in Table II. If we assume a scenario with an unconditional security wherein, an adversary using unlimited high-speed parallel computational resources, can cheat the verifier even for suggested k=4 and t=5 such that the soundness error may become non-negligible. Hence there will be a strong desire for

"zero" soundness error (as shown in Table II) i.e., successful authentication of an adversary should be made "impossible".

b. Loss of Zero-Knowledge

When Zero-Knowledge proofs are executed in parallel interactive mode, zero-knowledge property may not always be satisfied. Although this is computationally secure, we attempt to improve the scheme with unconditional security so that zero-knowledge property must be achieved without decreasing the accreditation and slowing down the communication.

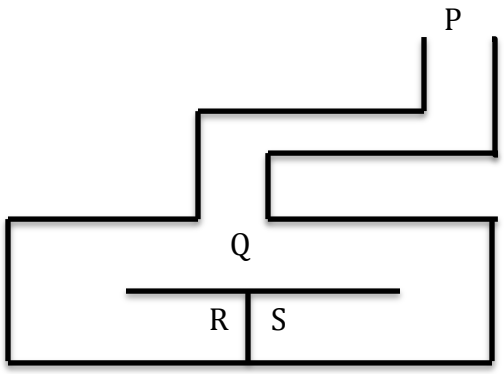
TABLE II CURRENT AND PROPOSED SCENARIO OF SOUNDNESS

Scenario	**Successful Authentication Of Adversary**	**Unsuccessful Authentication of** Adversary
Current	With probability 2^{-kt}	With probability $1 - 2^{-kt}$
Proposed	Impossible	Always

3.2. The Ali Baba cave problem

The concept of Zero-Knowledge proofs is best explained with the Ali Baba's cave problem. This problem is explained with Figure I.

FIGURE I. PROBLEM: ALI BABA's CAVE



Imagine that you (Peggy) are exploring a cave with a friend. You know a secret password to get through the door between R and S. You want to prove to your friend Victor that you know the password, without him finding out the password. Victor stands outside at P and Peggy vanishes into the cave and stands at R or S. Victor comes back in, stands at Q, and calls out which side Peggy should return by. If Peggy picked the correct side, she tricks Victor; otherwise, she has to admit she lied. If Victor does this enough times, Peggy will guess wrong (one trial, 50% she succeeds; two trials, 25%; etc.).

The main idea behind this reasoning is this: Peggy wants to prove a certain fact F_2 but she does not want to disclose the proof. She then finds another fact F_2, that may be publicly disclosed, such as F_2 is true if F_1 is true (necessary condition). So she "delegates" the proof of F_1 by proving actually F_2 only. In this example, F_1 is the knowledge of the magic word, and F_2 is the ability of appearing from any of the sides of the tunnel. If Victor agrees that cannot be true without F_1 being true too, then the protocol may start.

FIGURE II. GENERAL STRUCTURE: ALI BABA's CAVE

	Victor stands outside
Peggy: Prepare	Peggy vanishes into the cave
Peggy: Commit	Victor comes back in, …
Peggy: Challenge	… and calls out which side Peggy should return by
Peggy: Respond	Peggy returns as requested

4. Fiat-Shamir Identification Scheme

The Fiat–Shamir identification scheme allows one party, Peggy, to prove to another party, Victor, that she possesses secret information without revealing to Victor what that secret information is. I will first explain the basic version of the protocol, then the more general version and finally the latest version.

4.1 The basic version

This version uses as many public keys as there are centers who issue the keys. We will use the symbol (integer) n for the public key of a center, where n = p·q such that p and q are secret primes only known to the center. There exists a standard keyless (pseudorandom) one-way function f. Let us call I the "name" of an individual (e.g., Peggy) who wants to receive a public key from the center. To be unique, I contain relevant information about the individual; e.g., the name, address and physical description. For each

individual, the center picks a small j such that m = f(I, j) is a quadratic residue (mod n). The center calculates the smallest $\sqrt{m}$ (mod n) and gives it to the individual. We will refer to $\sqrt{m}$ as the secret identification of the individual. If Peggy wants to identify herself to Victor then they use the following Ping-Pong protocol. First she tells Victor her nationality, her "name" (I) and j. So Victor knows which n to use. Victor calculates m corresponding with I and j. Then the Ping-Pong part starts:

a. Peggy chooses a random s (mod n) which we will further call $\sqrt{t}$ and Peggy squares it (mod n) to obtain t. She sends t to Victor.
b. Peggy sends Bob one random bit e.
c. Peggy then sends $\alpha = \sqrt{t} * \sqrt{m}^e$.
d. Victor verifies by squaring. (This is trivial, because he has to verify that $a' = t * m^e$ (mod n) and he knows m and t, because Peggy has sent that.)

Somebody else could have claimed to be Alice with a probability of 1/2. To decrease this success of cheating, the protocol is repeated as many times as required for security. We will call Victor the verifier and Peggy the Prover.

4.2 The general version

In the general version, instead of having only one j, k and j^i exists so that k m_i ($1 \le I \le k$) and k $\sqrt{m_i}$ exist. In the Ping-Pong protocol, Victor sends k e_i in step 2. In step 3 now Peggy sends

$$A = \sqrt{t} * \pi_{ei = 1} \sqrt{m_I} \text{ (mod n)}.$$

Victor verifies (in Step 4 by squaring, this means calculates first (in parallel with Step 3) $\beta = t * \pi_{ei=1} m_i$ and hence he receives α, he squares it to verify that $\alpha^2 = \beta \pmod{n}$.

4.3 Feige-Fiat-Shamir Identification Scheme

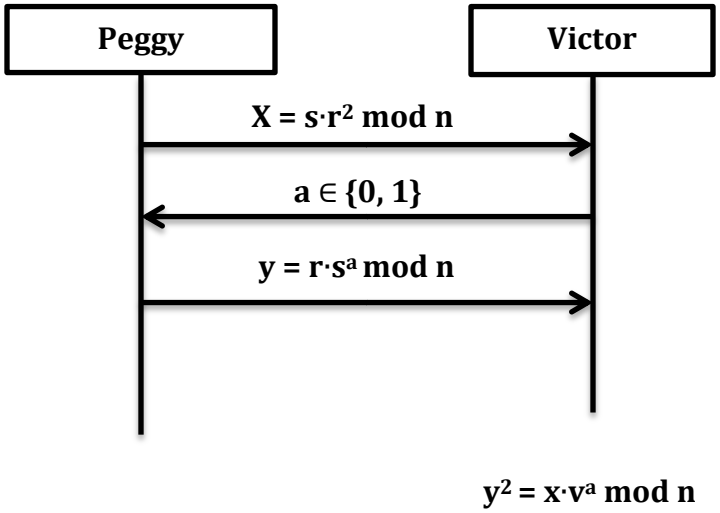
The Feige–Fiat–Shamir identification scheme, however, uses modular arithmetic and a parallel verification process that limits the number of communications between Peggy and Victor. The differences between the Fiat-Shamir and Feige-Fiat-Shamir protocol seem small, however they are important. The important difference is that in their new scheme the role of the center is enormously reduced. The only role of the center is to publish an n of the appropriate form (the product of two large primes each of the form 4r + 3) after which it closes. Each individual chooses k random numbers $s_i \pmod{n}$. He then chooses each $m_i = \pm s_i^2 \pmod{n}$, where the sign is decided randomly and independently. He keeps s_i secret and makes the m_i public.

4.3.1 Setup

Choose two large prime integers p and q and compute the product n = pq. Create secret numbers $s_1, \cdots, s_k$ with $\gcd(s_i, n) = 1$. Compute $v_i \equiv s_i^2 \pmod{n}$. Peggy and Victor both receive n while p and q are kept secret. Peggy is then sent the numbers s_i. These are her secret login numbers. Victor is sent the numbers v_i by Peggy when she wishes to identify herself to Victor. Victor is unable to recover Peggy's s_i numbers from his v_i numbers due to the difficulty in determining a modular square root when the modulus' factorization is unknown.

4.3.2 Procedure

FIGURE II. The Ping-Pong protocol



a. Peggy chooses a random integer r, a random sign s ∈ {-1, 1} and computes $x \equiv s \cdot r^2 \pmod{n}$. Peggy sends x to Victor.

b. Victor chooses numbers $a_1,\cdots,a_k$ where a_i equals 0 or 1. Victor sends these numbers to Peggy.

c. Peggy computes $y \equiv r s_1^{a1} s_2^{a2} \cdots s_k^{ak} \pmod{n}$. Peggy sends this number to Victor.

d. Victor checks that $y^2 \equiv \pm x v_1^{a1} v_2^{a2} \cdots v_k^{ak} \pmod{n}$.

This procedure is repeated with different r and a_i values until Victor is satisfied that Peggy does indeed possess the modular square roots (s_i) of his v_i numbers.

4.3.3. Alternate versions

An alternate version of the Feige-Fiat-Shamir identification scheme is given by Schneier. In Schneier's version, Peggy chooses k random numbers $s_1,\dots, s_k$ where s_j is a quadratic residue mod n, and publishes them as the public key. Then she calculates the smallest I_j such as $I_j = \sqrt{(1/s_i)} \bmod n$, and keeps $I_1, \dots, I_k$ as the secret key. The following four steps of the protocol take place as explained above. The disadvantage of this version of the protocol lies in the difficulty of computing quadratic residues.

4.3.4. Security

In the procedure, Peggy does not give any useful information to Victor. She merely proves to Victor that she has the secret numbers without revealing what those numbers are. Anyone who intercepts the communication between Peggy and Victor would only learn the same information. The eavesdropper would not learn anything useful about Peggy's secret numbers.

In an early version, the Fiat–Shamir identification scheme, one bit of information was leaked. By the introduction of the sign s this bit was concealed resulting in a zero-knowledge-protocol.

Suppose Eve has intercepted Victor's v_i numbers but does not know what Peggy's s_i numbers are. If Eve wants to try to convince Victor that she is Peggy, she would have to correctly guess what Victor's a_i numbers will be. She then picks a random number y, calculates $x \equiv$

$y^2 v_1^{-a1}\ v_2^{-a2} \cdots v_k^{-ak} \pmod n$ and sends x to Victor. When Victor sends a_i, Eve simply returns her y. Victor is satisfied and concludes that Eve has the secret numbers. However, the probability of Eve correctly guessing what Victor's a_i will be is 1 in 2^k. By repeating the procedure t times, the probability drops to 1 in 2^{kt} . For k = 5 and t = 4 the probability of successfully posing as Peggy is less than 1 in 1 million.

5. Analysis of Feige-Fiat-Shamir Identification Scheme

5.1. Approach

We use the theoretical proofs of cheating Alice theorem and cheating Bob theorem for Fig. 1 to prove that soundness error and loss of zero-knowledge respectively may exist. For cheating Alice theorem, adversary will use Eq. 1 to compute x for the commitment phase. If it guesses the correct challenge b, it will randomly generate y_2 such that y_2 satisfies Eq. 2. Once the correct value for y_2 is found, it is easy to compute the secret $s = \sqrt{(v^{-1})}$ using Eq. 3.

$$x = y_1^2 . v^n \qquad (1)$$

$$x = y_2^2 . V^{1-b} \qquad (2)$$

$$\sqrt{(v^{-1})} = y_1.y_2 \bmod n \qquad (3)$$

For cheating Bob theorem, adversary need not guess the challenge b and has a choice to generate it. After receiving x from the Prover, it will randomly generate y_2 such that y_2 satisfies Eq. 2. Once the correct

value for y2 is found, it is easy to compute the secret $s=\sqrt{(v^{-1})}$ using Eq. 3.

5.2. Simulations

We divide our experiment into two simulations ffs_soundness_error and ffs_loss_zk in order to prove that the soundness error and loss of zero-knowledge property respectively exist under parallel interactive composition. These simulations use Eq. 1, 2 and 3 for execution. However it is practically impossible to find these problems unless common modulus n is <=20 bits and security parameters k=3 and t=5. Therefore we use artificially very small parameters. For both the simulations ffs_soundness_error and ffs_loss_zk, data is randomly generated.

The trusted center T selects the primes p = 683, q = 811, and publishes n = pq =553913.

1. Prover does the following.

   (a) Selects 3 random integers s1 = 157, s2 = 43215, s3 = 4646, as private key set and 3 bits b1 = 1, b2 = 0, b3 = 1.

   (b) Computes v1 = -112068, v2 = 338402, and v3 = -429490.

   (c) Prover's public key is (-112068, 338402, 124423; -429490) and private key is (157, 43215, 4646).

2. For ffs_soundness_error

(a) Adversary gets the public key (-112068, 338402, 124423; -429490) and tries to impersonate the prover.

(b) Verifier generates the challenge bit set randomly.

3. For ffs_loss_zk

(a) Adversary gets the public key (-112068, 338402, 124423; -429490) and tries to impersonate the verifier.

(b) Adversary generates the challenge bit set randomly.

5.3 Security Analysis of Soundness Error

When k=1, t=1 the simulation ffs_soundness_error runs according to the cheating Alice theorem. However, when k>1 and t>1, Adversary can find $\sqrt{(v^{-1})}$ only if the challenge bit set contains either

- All 0 bits or
- Single 1 bit

Thus we need to avoid such challenge bit set in order to eliminate the soundness error. Using trial and error method, we re-formulate the Eq. 1, 2 and 3 to Eq. 4, 5 and 6 respectively. We use these equations to find the secret in interactive parallel composition.

$$x = r^2 . v_k^{bk} \bmod n \qquad (4)$$

$$x = y_2^2 . v_k^{1-bk} \bmod n \qquad (5)$$

$$\sqrt{(v^{-1})} = \begin{cases} r.\, y_2^{-1} \bmod n; \text{ if } b_k = 0 \\ r^{-1}.y_2 \bmod n; \text{ otherwise} \end{cases} \quad (6)$$

The adversary could successfully impersonate the prover when it guessed correctly the challenge to be sent by the verifier, at t=2 and t=5. After successful impersonation, the adversary could find the possible values for the private key set after computing $\sqrt{(v^{-1})}$ using Eq. 4, 5, and 6. It is found that the values of $\sqrt{(v^{-1})}$ remain the same in any iteration, even if r value is changed randomly. For an instance, when public key v2=338402, at t=2, the values of $\sqrt{(v^{-1})}$ are (43215, 114930, 438983, 510698) and at t=5, the values of $\sqrt{(v^{-1})}$ are (43215, 438983, 114930, 510698).These are the possible values of private key s_2. If the number of bits of n ranges 512-1024, it is agreed that the soundness error is negligible since it is computationally secure. However this error cannot be negligible forever. This experiment concludes with the need of elimination of soundness error for unconditional security.

5.4. Security analysis of loss of Zero-knowledge

The simulation ffs_loss_zk tests the cheating Bob theorem with k>1 and t>1 using Eq. 5 and 6 . We find that adversary impersonating as the verifier could find the secret of

the prover using Eq. 5 and 6 in every iteration. This is because the adversary need not guess the challenge and have choice to randomly generate the challenge to find the secret. This practically proves that zero- knowledge is not closed under parallel interactive composition. Although n with 512-1024 bits, it is computationally secured, this experiment concludes with the need to achieve complete zero-knowledge for unconditional security.

6. Implementation

To test properly the Feige-Fiat-Shamir Identification Scheme, I wrote a small implementation of it in java language, on a UNIX based machine.

My program is based on the original paper by Feige, Fiat, and Shamir titled "Zero-Knowledge Proofs of Identity". To keep the implementation simple, I let t = 1, that is the program makes only one round of the protocol.

5.1. Architecture

NumberGenerator.java creates a pseudorandom number generator. Method **generatePrimeNumber()** takes as parameters a minimum and maximum value and generates a list of prime numbers. Method **getLargePrimeNumber()** gets a large prime from the list of primes.

FFS.java simulates the communication between the prover and the verifier, and handles the whole protocol

5.2. Procedure

Here is a step-by-step explanation of my implementation:

a. Generate a pseudo random number generator

b. Choose two large primes form the pseudo random number generator → p and q

c. Take the product of p and q to get n → n = p*q

d. Generate a secret number s such that s is a coprime of n. To get s, you look for a number s such that gcd(s, n) = 1

e. Then you compute v = s^2 (mod n).

f. Peggy and Victor both receive n while p and q are both kept secret

g. Peggy is sent the number s which is her secret login numbers while Victor is sent the number v by Peggy when she wants to communicate with him.

h. Peggy chooses a random number r and a random sign s and computes x = s * r^2 (mod n) and sends x to Victor

i. Victor chooses a random number a, where a = 0 or 1. Victor then sends this number to Peggy

j. Peggy receives a and computes y = r *(s^a) (mod n). Peggy then sends y to Victor

k. Victor then receives y and checks if y^2 == + or – (x* (v^a)) (mod n). If they are equal, the authentication was a success else it was a failure

6.3. Output

The program makes one round of the FFS protocol and prints on the screen the results. The following are printed on order:

- The modulus
- The secret number
- The witness from the prover
- The challenge from the verifier
- The response from the prover
- The verification value computed by the verifier
- The result of the authentication: Successful is the verification value matches the witness.

7. Conclusions

The basic problem with this type of identification technique is that it is subject to man-in-the-middle attacks, in which a dishonest verifier Eve makes a copy of the proof of identity given by Peggy, to misrepresent successfully herself to another verifier Vincent. This is done by Eve relaying every single message from Peggy to Vincent and vice versa.

The counterattack for this kind of attack is a strong synchronization: a certain time limit is imposed for the replies, in the purpose that there will not be enough time for relaying the communications. Another counterattack, which may be used in addition to the first, is to require all identifications to take place inside protected zones (shielded rooms, Faraday cages) to prevent communication relay.

8. Bibliography

1. Feige, Uriel; Fiat, Amos; Shamir, Adi (1988). "Zero-knowledge proofs of identity". Journal of Cryptology 1 (2): 77–94.
2. O. Goldreich, S. Micali, A. Wigderson. Proofs that yield nothing but their validity. Journal of the Association for Computing Machinery. Vol 38, No 1, July1991, pp 691-729
3. Kai-Min Chung, Rafael Pass, and Wei-Lung Dustin Tseng. The Knowledge Tightness of Parallel Zero-Knowledge
4. Yvo Desmedt, Claude Goutier and Samy Bengio. SPECIAL USES AND ABUSES OF THE FIAT-SHAMIR PASSPORT PROTOCOL
5. Daniele Raffo, (2002). Digital Certificates and the Feige-Fiat-Shamir zero-knowledge protocol
6. Dhanya R. Sarath, Megha V. Ainapurkar, An Improved Parallel Interactive Feige-Fiat-Shamir Identification Scheme with almost zero soundness error and complete zero-knowledge