An Alternative Methodology For Authentication And Confidentiality Based On Zero Knowledge Protocols Using Diffie-Hellman Key Exchange

P Lalitha Surya Kumari,
Assistant Professor,
Dept. of Computer Science and engineering
Bharat Institute of Engineering and Technology,
Hyderabad, Andhra Pradesh
vlalithanagesh@yahoo.co.in

Prof. Avula Damodaram,
Dept. of Computer Science and Engineering,
Director, University Academic Audit Cell
JNT University,
Hyderabad, Andhra Pradesh
damodarama@rediffmail.com

Abstract

This paper presents a concept for a new method to provide the authentication and confidentiality using zero knowledge protocol and key exchange. Zero knowledge proof protocol is a essential component of cryptography, which in recent years has increasingly popular amongst scholars. Its applications have widened and it has made inroads in several areas including mathematics and network safety and so on. This simple protocol based on zero knowledge proof by which user can prove to the authentication server that he has the password without having to send the password to the server either clear text or in encrypted format. This is a protocol in which the data learned by one party (i.e., the inspector) allow him/her to verify that a statement is true but does not reveal any additional information. In this paper we first discuss about zero-knowledge protocol proof system of knowledge and also key exchange between users and which then is modified into an authentication scheme with secret key exchange for confidentiality. The whole protocol involves mutual identification of two users, exchange of a random common secret key or session key for the verification of public keys

Keywords: zero knowledge proof; authentication; confidentiality; key exchange; signature; session key; secret key; public key

I. Introduction

An interactive proof [1][2] is a protocol that is defined between a prover, usually called P or Peggy, and a verifier, usually called V or Victor. More formally, an interactive proof is a pair (P, V) of programs that implement the protocol steps that Peggy and Victor are supposed to execute. An interactive proof must be complete and sound. Completeness means that an honest prover succeeds in convincing the honest verifier, and soundness means that a dishonest prover does not succeed in convincing Victor of a false statement. An interactive proof transfers only the conviction that the claimed statement is true but does not leak any further information, in particular not a manageable proof. More precisely, an interactive proof is called zero-knowledge if the verifier could simulate the entire protocol transcript by himself, without interacting with the prover. In particular, this implies that the transcript is not convincing for any other party.

There are two types of interactive proofs [2]: proofs of a mathematical statement and proofs of knowledge. A proof of knowledge proves that Peggy knows a value satisfying a certain predicate (a witness). Often a proof of a mathematical statement (e.g. that a number is a square modulo an RSA modulus) is carried out as a proof of knowledge of a witness for the statement (i.e., of a square root).

The Diffie-Hellman protocol allows two parties to exchange a secret key over an open communication channels without meeting in advance. Symmetric encryption application makes use of the same secret key for secure communication between two parties. It was a brilliant insight of Diffie and Hellman that the difficulty of the discrete logarithm problem provides a possible solution. However, this protocol is becomes vulnerable to a specific man-in-the-middle attack, if the key exchange takes place in certain mathematical environments. To prevent this potentially fatal protocol attack, the two parties have several options. The easiest method is to force authentication prior to the key exchange.

II. Zero Knowledge Protocol.

A. Main Proposal

P (the prover) had some secret information and wanted to prove V (the verifier) by taking other proof process without revealing anything other than the fact that it knows in order to prevent the confidential information from leaking to anyone (including V or any other third party). We call this technology as zero knowledge proof (ZKP) [4] which can achieve the purpose of proof without revealing anything. This method can be a good solution for proving mutual identity,

In Zero Knowledge protocols [4], a Prover tries to prove knowledge of a secret to a Verifier without revealing the secret itself. The Verifier can ask questions to find out if the Prover really knows the secret. It is impossible for the Verifier 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 and does not able to learn anything about the secret, or convince somebody else that he knows the secret if the protocol is secure. There is also a malicious user able to send, modify or destroy messages. A good protocol should be resistant against this user. A protocol has to consider cases 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 by taking the following factors into account:

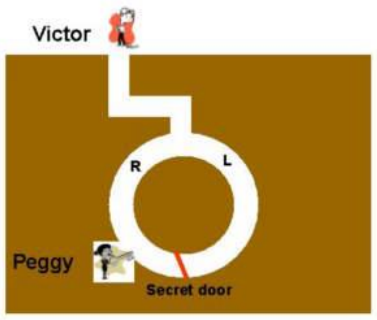
1. If Peggy does not know the secret information, she is unable to pretend to have such knowledge. Many rounds of the protocol should assure that (with probability close to 1) she couldn't cheat Victor.

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

B. Example investigation

(1) Supposing a room [4] can only be opened by a key with nothing else available, the prover P wanted to prove himself owning the room key without revealing the key to the verifier V to prevent leakage. Therefore, if V determined that the room had a certain object P can take out the object to prove himself having the key. This proof process is zero-knowledge proof, and "knowledge" is the key.

(2)The most typical example of zero-knowledge proof is the Cave model [5]. As shown in Figure given 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 arbitrary. It is obvious 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.

A main point here is that Victor is not able to convince anybody else even if he is completely convinced that Peggy knows the secret word. Let's say that he records everything. An Eavesdropper could think that Victor agreed with Peggy about which branch to choose each time. This attests that Victor cannot use any information he got for his own purposes.

(3) Zero-knowledge proof of color blind [4] with red and green ball supposing the verifier V is color blind. There are now two balls, one is red and the other is green. The two balls are exactly the same besides their color. It is required to prove to V that the two balls are truly different because they seem to be identical to him. We adopt the following method: let V hold a ball in each hand, standing opposite P. P can also see these two balls without telling V which is red and which is green. Then, let him take hands behind his back, he can decide whether to swap the balls or not at random. It has the probability of ½ whether exchanging or not. Finally, he takes balls from back and asks P to answer whether V has exchanged two

balls. According to the color of the ball, P can simply judge.
Repeatedly, if P can answer correctly every time, then V will believe that these two balls color are different to a large enough probability. The process also belongs to zero-knowledge proof, and "knowledge" is the color of the ball.

C. The common course of action of zero-knowledge proof protocol is

P **V**

Secret calculation	r	Verification
	e	
	Response for e	

(1)The prover P sends promise random number r to the verifier V.
(2) V sends a random challenge value e to P.
(3) P calculates secretly and sends the result to V as the challenge-response for second step.
(4) V verifies the response. If the verification fails, the process of proof will end. Otherwise, the above steps will be repeated for N times.
If every verification is successful, the probability that V receives P's proof is very high.

D. Characteristics of zero-knowledge proof protocol

Zero-knowledge proof protocol has the following three necessary properties [4] .In other words, the proving system of the zero knowledge proof protocol must meet the following requirements to prove a problem:

(1) Completeness
If the prover P and the verifier V comply with the common course of action of zero- knowledge proof protocol strictly, the proof is considered to be successful and P is credible

(2) Rationality
If it fails once in N times of verification, the proof is regarded as failed and P is a fake prover who is unreliable

(3) Zero-knowledge
During the verification, V can't obtain any privacy or important information, let alone anything about the knowledge, except to believe that P does have it. Even though V verifies repeatedly, he cannot prove the existing fact to others anymore
.

E. Mathematical knowledge applied to zero-knowledge proof protocol

The commonly used algorithm theory in zero-knowledge proof mainly based on the following key mathematic problem:

(1) **The square root problem of modulo n**
Given a positive integer n, a∈Zn, if there exists x∈Zn that makes x^2=a(mod n), then x is called as a square root of modulo n.

(2) **Using problem of discrete logarithm**
Given a prime number p and a, which is one of the primitive element on finite field Zp .To b on Zp, looking for one and only integer c that makes a^c≡b(mod p).
In general, the problem is difficult if you are looking forward p, and there is still no algorithm to calculate polynomial of discrete logarithm. The method based on the elliptic curve discrete logarithm is commonly used.

(3) **Large integer factorization problem**
The factorization of a large integer M which is N digit, is usually impossible to be done in O (N), but rather to up to O(exp(N)) level.

III. The Diffie-Hellman Protocol

The Diffie-Hellman protocol [3][6] provided the first practical solution to the key distribution problem. It allows two parties to establish a shared secret by exchanging messages over an open channel without having met in advance or shared keying material,. The key can then be used to encrypt subsequent communications using a symmetric key cipher. The security rests on the intractability of the Diffie-Hellman problem and the related problem of computing discrete logarithms]. We will call the two parties conducting the key exchange "Alice" and "Bob."

A. The steps performed in Diffie Hellman Key exchange protocol [3][6]:
1. A prime number p and generator 'a' of Z^*_P ($2 \leq a \leq p - 2$) are selected and published.
2. Alice chooses a random secret x, $1\leq x \leq p- 2$, and sends Bob a^x mod p
A→ B: a^x mod p

3. Bob chooses a random secret y,$1 \leq y \leq p - 2$, and sends Alice $a^y \bmod p$
$B \rightarrow A$: $a^y \bmod p$
4. Bob receives a^x and computes the shared key as
$K = (a^x)^y \bmod p$
5. Alice receives a^y and computes the shared key as
$K = (a^y)^x \bmod p$
Because $(a^x)^y = (a^y)^x$, Alice and Bob have arrived at the same secret key. Only x, y, and a^{xy} are kept secret. All other values are sent in the clear.

B. **Example illustrates the Diffie Hellman Key exchange protocol [3][6]**

1. Alice and Bob agree on p= 353 and a=3.
2. Alice chooses x=97 and sends Bob 40= (3^{97} mod 353).
A->B: 40
3. Bob chooses y=233 and sends Alice 248= (3^{233} mod 353).
B->A: 248
4. Bob receives 40 and computes 40^{233} mod 353= 160
5. Alice receives 248 and computes 248^{97} mod 353 =160
Alice and Bob have agreed upon 160 as their secret key.
Figure demonstrates which parties know what information. The man-in the-middle will be called Eve from here on out.

Alice			Bob			Eve	
Knows	Does not Know		Knows	Does not Know		Knows	Does not Know
p=353	y=233		p=353	x=97		p=353	x=97
a=3			a=3			a=3	y=233
x=97			y=233			a^x=40	K=160
a^x=40			a^x=40			a^y=248	
a^y=248			a^y=248				
$(a^y)^x$=K =160			$(a^x)^y$=K= 160				

Obviously, a much larger value of p is required than used in the example to make the key agreement potentially secure. If the prime number 353 was used, Eve could simply try all possible values of $(3^x)^y$ mod 353. Because 3 is a primitive root modulo 353, this can take 352 values. A key space with only 352 possibilities can be exhausted with ease. However, if the prime number used is big enough, so that no computing power available today can consume the key space. For instance, most applications recommend 1024-bit primes. This correlates to a number of about 300 digits and makes searching the key space one by one infeasible. Table given below demonstrates how long it would take a modern personal computer (PC) and a super-computer (SC) to exhaust various sizes of key spaces. We assume a PC can search approximately one million (106) keys per second, while a super-computer can search approximately one trillion (1012) keys per second. For instance, if a prime of 64 bits was used, it would correlate to a base ten number of approximately 19 digits. The key space would be all the numbers 1, 2,..., p-1, which would be on the order of 10^{19} numbers. Therefore, a PC would take$10^{19}/10^6=10^{13}$ seconds to completely search the entire key space.

Considering most applications use prime of 1024 bits or greater, it is obviously infeasible to conduct a random search of an entire key space. Of course, one could get lucky and the key could be one of the first numbers searched by the computer. However, as indicated by the enormous times listed in the table, it is more likely a random key search would take longer than most scientists believe the universe has existed.

Bits	Digits (approximate)	PC time (approximate)	SC time (approximate)
64	19	317,098 years	115 days
128	39	3x10^(25) years	3 x 10^(19) years
256	77	3x10^(63) years	3x10^(57) years
512	154	3x10^(140)years	3x10^(134) years
1024	308	3x10^(294) years	3x10^(288) years
2048	616	3x10^(602) years	3x10^(596) years

IV. An alternative methodology for authentication and confidentiality

The method involves mutual identification of two users, exchange of a random common secret key or session key for the verification of public keys over an open channel. It enhances security by preventing the Man in the middle attack when the Diffie Hellman protocol is used. The new protocol sends shared secret or session key through Zero Knowledge Protocol. Hence, An Eavesdropper is a third party that listens to the conversation but, he is not able to learn anything about the secret (public keys send between two parties to generate secret of shared key) or convince somebody else that he knows the secret. It also enhances security by providing the session key or secret key used by the both the parties along with the information (information send using ZKP) used for identification of both users. That means it provides authentication among both users without creating any signature for the verification of the public keys.

A. Procedure

1. Both parties prove themselves as authenticate using Zero Knowledge proof identity protocol
2. Once both the parties proved themselves as authenticated then they exchange their information
3. Exchange of information will be done using Diffie – Hellman key exchange protocol.
4. Both parties create shared secret key using their known information
5. That shared secret key will be used for encryption and decryption

B. Proposed Algorithm

1. A prime numbers **p** and **q**, generator **a** $\mathbf{Z^*_P}$ **($2 \leq a \leq p - 2$)** are selected and published.
2. **g** is a generator of an order **–q** subgroup of $\mathbf{Z^*p}$
3. The prover P Prover Knows s such that **$t=g^s$ mod p** and wants to prove this fact to verifier V
4. P selects a random no **r** in [1.. q] and calculates X_1= **g^r mod p** sends to V
5. V chooses a random secret c in$[1..2^n]$ and sends to P
6. P calculates **z=r + sc**
7. V verifies the response X_2=**$g^z t^c$ mod p** [i.e. **$g^{r+sc}(g^s)^{-c}$ mod p = g^r mod p]**
8. If the verification fails, the process of proof will end. Otherwise, the above steps will be repeated for N times
9. If every verification is successful then V will receive P's proof in great probability.
10. P chooses a random secret x, $1 \leq x \leq p-1$ and calculates a^x mod p
11. V chooses a random secret y, $1 \leq y \leq$ p - 2, and calculates a^y mod p
12. P calculates shared key k= $(a^x)^y$ mod p
13. V calculates shared key k= $(a^y)^x$ mod p

$(a^x)^{y =} (a^y)^x$ and hence both P and V have arrived at the same shared key.

P V

	P sends **x = g^r mod p**	
	V sends random number **c**	
	P sends **z=r + sc**	
	V verifies X=**$g^z t^c$ mod p** And sends to P	
	If verified, P calculates **a^x mod p** and sends	
PROVER	V calculates **a^x mod p** and sends to P	VERIFIER
	P calculates k= $(a^x)^y$ mod p	
	V calculates k= $(a^y)^x$ mod p	
	P and V have arrived at the same shared key	

For efficiency, it is suitable to select values of g and r are small, i. e 3. Then the only really time consuming part is finding g^r mod p from X. Note that the adaptation with key exchange does not require more time, if g is small, P can pre-compute **g^s mod p**, and the protocol will then only require 2 extra exponentiations to the g'th power, which is negligible(insignificant). With a little care, it is possible to use g = 2, which will be even more efficient. Finally, note that P can prove himself to V while V is proving himself to P. In particular we can ensure that "decryption" of X takes place simultaneosly for P and V. Thus this will not take more time than the basic version of the protocol. It will be natural to obtain the final key as the K of the two produced keys, since this will ensure that the key is known to precisely P and V, and no one else. The basic operations needed for this protocol have been implemented on a standard16 MHz Intel 80386 processor. The results of this procedure show that the whole protocol including verification of public-key certificates can be completed in very less time.

C. Analysis of the Proposed Protocol

Recall the definition of ZKP discussed in (II.D), the proposed protocol satisfies the ZKP properties as follows:

a. Completeness: If Alice (the prover) and Bob (the verifier) are honest, then on performing the protocol steps, it must ends with {accept, reject} decision. That depends on the computed value of the X calculated by Prover and verified by Verifier ($\mathbf{g^{r+sc}(g^s)^{-c} \bmod p = g^r \bmod p}$), which can be either identical [accept] or different [reject].

b. Soundness: if the prover fail to compute the correct value of X_1 and z, the reply (X_2=$\mathbf{g^z t^c \bmod p}$) will be different from the value computed by Verifier. The values of X_1 and X_2 can't be guessed, there are two possibilities; either ($X_1=X_2$) or ($X_1 \neq X_2$).

c. Zero-Knowledge: on completion, both parties; the prover and the verifier would not have any further information other than their own secret numbers and calculated secret key. Secret numbers x, y and K were not revealed.

d. The proposed protocol analysis follows the analysis of Diffie-Hellman Key Exchange algorithm. It can be protected against discrete logarithm attack. The proposed protocol is protected from man-in-the-middle attack.

D. **Applications of proposed methodology**

This methodology can be applied where secret knowledge that is too sensitive to reveal needs to be verified such as RFID tags, passports, PIN numbers.This methodology can also be applied where key exchange is required like in Secure Sockets Layer (SSL)/Transport Layer Security (TLS), Secure Shell (SSH), Internet Protocol Security (IPSec), Public Key Infrastructure (PKI).

Though no single application can be pinpointed as a compelling application for use of this protocol, it can be a better choice for several applications considering the absence of the third party and the involvement of less mathematical calculations. A few real world applications of the proposed protocol would be Network Authentications, Smart Cards and Key Exchanges.

V. Conclusion

The paper provides the knowledge of the abstraction of different protocols such as ZKP and Diffie Hellman Key Exchange protocol. We have seen a zero knowledge proof is a way to prove that user knows a secret without revealing the secret. 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. A practical scenario is proving that a public key is good without revealing the private key. We have seen some examples of classical identification scheme and some new techniques using zero knowledge theory

The Diffie-Hellman protocol provides practical solution to the key distribution problem, allowing two parties, never having met in advance or shared keying material, to establish a shared secret by exchanging messages over an open channel.

The new protocol involves mutual identification between two users. This protocol involves authentication using ZKP and confidentially using ZKP and D-H-Key protocol. This new protocol provides confidentiality without using any encryption techniques (such as RSA). Hence, it is more efficient because it reduces complexity of algorithm used for encryption techniques. It also increases the performance due to less mathematical calculations used when compared to public key cryptographic systems. This protocol works with low memory when compared with PK encryption algorithms. Moreover a third party may not be able to impersonate sender (Alice) to convince receiver (Bob).

VI. References

[1] SIAM Journal on Computing 18,86–208 (1989) Goldwasser, S., Micali, S., Rackoff, C.”The knowledge complexity of interactive proof systems”

[2] Ueli Maurer, Department of Computer Science, ETH Zurich, CH-8092 Zurich, Switzerland, maurer@inf .ethz.ch “Unifying Zero-Knowledge Proofs of Knowledge “

[3] By Aaron C. Geary Co-Advisors: Pantelimon Stanica Valery Kanevsky “Analysis Of A Man-In-The-Middle Attack On The Diffie-Hellman Key Exchange Protocol “ September 2009,

[4] Wang Huqing, Sun Zhixin “Research on Zero-Knowledge Proof Protocol”

[5] Security Essentials - Version 1.2e, Annarita Giani “Identification With Zero Knowledge Protocols”

[6] ” Cryptography and Network Security” William Stallings