A New Efficient Protocol for Authenticated Key Agreement

Fatma Ahmed and Dalia Elkamchouchi

***Abstract*—Key establishment protocols are among the most important security mechanisms via which two or more parties can generate a common session key to in order to encrypt their communications over an otherwise insecure network. In this paper we propose an efficient and secure authenticated key agreement protocol based on DLP (Discrete Logarithm Problem). The main purpose of this paper is to achieve most of goals of key agreement. We show that our protocol meets the security attributes and strong against most of potential attacks. We try in our new protocol to provide the authentication between users with maintain the number of sending message minimum as possible and by using only one operation of multiplication, subtraction and exponentiation. We use the mathematica 9 program to implement the new proposed system.**

***Index Terms*—DLP, key agreement, safe prime.**

I. Introduction

Key agreement protocols are fundamental to establishing secure communications between two or more parties over an insecure network. A key establishment protocol (including key agreement protocol) allows two or more communicating parties to establish a common secret key via public communication channels (e.g., Internet). The established session key can then be used to create a confidential or integrity protected communication channel between the parties. The Key establishment protocols come in various flavors. In key transport protocols, a key is created by one entity and securely transmitted to the second entity, while in key agreement protocols both parties contribute information which is used to derive the shared secrete key[1]. Authenticated key agreement (AK) protocols not only allow parties to compute the session key but also ensure the authenticity of the involved parties [2].

II. Prpposed Key Agreement Protocol

In order to counter most of potential attacks, we design a new efficient authenticated key agreement protocol. Our protocol consists of three phases; The Registration Phase, The Transfer and Substantiation Phase, and The Key Generation Phase.

Manuscript received January 11, 2013; revised March 19, 2013.

Fatma Ahmed is with the Dept. of Electrical engineering, Alexandria Higher Institute of Engineering and Technology (AIET) Alexandria, Egypt (e-mail: moonyally@yahoo.com).

Dalia Elkamchouchi is with the Dept of Electrical engineering, Faculty of Engineering, Alexandria University Alexandria, Egypt (e-mail: Daliakamsh@yahoo.com).

A. Notations Used

p': Long-term secret is large prime usually at least 1024 bits.

p: Long-term public is large safe prime: $(n'p'+1)$.

n': Small prime number (usually taken by 2).

$p1$: Long-term secret, Euler's totient function $p1=(p-1)$.

G: Subgroup of Z_p^* of order p'.

g: Generator of G.

r_A, r_B: Short-term private keys are random integers: $2 \le r_A, r_B < p1$ and $GCD(r, p1)=1$.

t_A, t_B: Short-term public keys: $t_A \equiv g^{r_A} \bmod p$ and $t_B \equiv g^{r_B} \bmod p$.

x_A, x_B: Long-term private keys are random integers: $2 \le x_A, x_B < p1$ and $GCD(x, p1)=1$.

y_A, y_B: Long-term public keys: $y_A \equiv g^{x_A} \bmod p$ and $y_B \equiv g^{x_B} \bmod p$.

K_{AB}: The shared secret key calculated by the principals.

B. The New Protocol Description

In this section we describe a proposed authenticated key agreement protocol between two parties A and B. The protocol works in the following steps:

1) The registration phase

Each user like A and B selects a safe prime p, then calculates generator g. Each user selects two static secret keys x_A and x_B, such that $2 \le x_A, x_B < p1$.

Next calculates

$$y_A \equiv g^{x_A} \bmod p \ , y_B \equiv g^{x_B} \bmod p$$

and registers y_A, y_B to the public file.

2) The transfer and substantiation phasea

- A generates the ephemeral key r_A such that $2 \le r_A < p1$, then calculates $t_A \equiv g^{r_A} \bmod p$.

- B generates the ephemeral key r_B such that $2 \le r_B < p1$, then calculates $t_B \equiv g^{r_B} \bmod p$.
- A calculates:

$$a_1 \equiv (y_B)^{-r_A} \equiv g^{-x_B r_A} \bmod p$$

$$b_1 \equiv (x_A - r_A \bullet t_B) \bmod p1$$

$$d_1 \equiv a_1 \bullet b_1 \equiv \left(g^{-x_B r_A} + (x_A - r_A \bullet t_B)\right) \bmod p$$

and sends d_1 to B.

- B calculates:

$$a_2 \equiv (y_A)^{-r_B} \equiv g^{-x_A r_B} \bmod p$$

$$b_2 \equiv (x_{\mathbf{B}} - r_B \bullet t_A) \bmod p1$$

$$d_2 \equiv a_2 \bullet b_2 \equiv \left(g^{-x_A r_B} + (x_B - r_B \cdot t_A)\right) \bmod p$$

and sends d_2 to A.

- A receives B 's value and checks:

$$a_{22} \equiv (t_B)^{x_A} \equiv g^{r_B x_A} \bmod p$$

$$b_{22} \equiv a_{22} \bullet d_2 \equiv (x_B - r_B \bullet t_A) \bmod p$$

$$v_2 \equiv {t_B}^{t_A} \bullet g^{b_{22}} \equiv g^{r_B \bullet t_A} \bullet g^{(x_B - r_B \bullet t_A)} \bmod p \equiv y_B$$

If the comparison is true, it accepts the received vector.

- B receives A 's value and checks:

$$a_{11} = (t_a)^{x_B} \equiv g^{r_A x_B} \bmod p$$

$$b_{11} \equiv a_{11} \bullet d_1 \equiv (x_A - r_A \bullet t_B) \bmod p$$

$$v_1 \equiv {t_A}^{t_B} \bullet g^{b_{11}} \equiv g^{r_A \bullet t_B} \bullet g^{(x_A - r_A \bullet t_B)} \bmod p \equiv y_A$$

If the comparison is true, it accepts the received vector.

3) The key generation phase

- A calculates the session key,

$$K_{AB} \equiv {y_B}^{x_A} \bullet {t_B}^{r_A} \equiv g^{x_A r_B + r_A r_B} \bmod p$$

Unless the comparison is true, A will reject the received vector.

- B calculates the session key,

$$K_{AB} \equiv {y_A}^{x_B} \bullet {t_A}^{r_B} \equiv g^{x_A x_B + r_A r_B} \bmod p$$

Unless the comparison is true, B will reject the received vector.

In our protocol, we have only one message sends from one entity to another. The message sends from A to B and the message sends from B to A both have the same structure and independent on each other. The total number of transmitted bits (communication overhead) is $|p|$. The following Fig. 1 shows the overall operation in our new protocol.

A		B
Step 1		
$a_1 \equiv (y_B)^{-r_A} \equiv g^{-x_B r_A}$		$a_2 \equiv (y_A)^{-r_B} \equiv g^{-x_A r_B}$
$b_1 \equiv (x_A - r_A \bullet t_B) \bmod p1$	$\overset{d_1}{\rightleftarrows}$	$b_2 \equiv (x_B - r_B \bullet t_A) \bmod p1$
$d_1 \equiv a_1 \bullet b_1$	d_2	$d_2 \equiv a_2 \bullet b_2$
Step 2		
$a_{22} \equiv (t_B)^{x_A} \equiv g^{r_B x_A}$		$a_{11} = (t_a)^{x_B} \equiv g^{r_A x_B}$
$b_{22} \equiv a_{22} \bullet d_2 \equiv (x_B - r_B \bullet t_A)$		$b_{11} \equiv a_{11} \bullet d_1 \equiv (x_A - r_A \bullet t_B)$
$v_2 \equiv {t_B}^{t_A} \bullet g^{b_{22}} \equiv y_B$		$v_1 \equiv {t_A}^{t_B} \bullet g^{b_{11}} \equiv y_A$
Step 3		
$K_{AB} \equiv {y_B}^{x_A} \bullet {t_B}^{r_A}$		$K_{AB} \equiv {y_A}^{x_B} \bullet {t_A}^{r_B}$
$K_{AB} \equiv g^{x_B x_A + r_A r_B}$		$K_{AB} \equiv g^{x_A x_B + r_A r_B}$

Fig. 1. Overall operation in the proposed protocol

In the first step, the number of scalar multiplications required is one, the number of exponentiation required is one and the total number of sending message is one. In the second step, each user will be verified from the other one because in the first step each user uses the Short-term private key belongs to him in calculation.

III. PRPPOSED KEY AGREEMENT PROTOCOL

Our Protocol involves DL cryptographic assumption. The security of this protocol depends on the complexity of a DL [3]. Here we prove our protocol meets the following desirable security attributes [4].

A. Known-Key Security (K-KS)

Suppose an established session key between two parties is disclosed, the adversary is unable to learn other established session keys.

The protocol provides known-key security. Each run of the protocol between two parties A and B should produce a unique session key which depends on r_A and r_B. Although an adversary has learned some other session keys, he can't compute ephemeral private keys r_A and r_B. Therefore the protocol still achieves its goal in the face of the adversary.

B. (Perfect) Forward Secrecy

If both secret keys of two parties are disclosed, the adversary is unable to derive old session keys established by two parties.

The protocol also possesses forward secrecy. Suppose that static private keys x_A and x_B of two parties are compromised. However, the secrecy of previous session keys established by honest parties is not affected, because an adversary who captured their private keys x_A or x_B should extract the ephemeral keys r_A or r_B from the exchanged values to know the previous or next session keys between them. However, this is DLP (Discrete Logarithm Problem).

C. Key-Compromise Impersonation (K-CI)

Assume that parties A and B are two principals. Suppose A 's secret key is disclosed. Obviously, an adversary who knows this secret key can impersonate A to other parties.

However, it is desired that this disclosure does not allow the adversary to impersonate other parties to A.

Suppose A 's long-term private key x_A, is disclosed. Now an adversary who knows this value can clearly impersonate A. But he can't impersonates B to A without knowing the B 's long-term private key x_A. For the success of the impersonation, the adversary must know A 's ephemeral key r_A. So, also in this case, the adversary should extract the value r_A from $t_A \equiv g^{r_A} \bmod p$, this is DLP.

D. Unknown Key-Share (UK-S)

Entity A cannot be coerced into sharing a key with entity B without A 's knowledge, i.e., when A believes the key is shared with some entity $C \neq B$, and B (correctly) believes the key is shared with A.

Our protocol also prevents unknown key-share. Corresponding to B 's public static and ephemeral keys y_B, t_B an adversary can't register B 's public keys y_B, t_B as its own and according to the assumption of this protocol that d_2 has verified that B possesses the private static and ephemeral keys x_B, r_B respectively. So an adversary can't deceive A into believing that B 's messages are originated from him. Therefore A cannot be coerced into sharing a key with entity B without A 's knowledge.

E. Subgroup Confinement Attack

Also small subgroup attack [5], the generator g in is a primitive root of the prime p. If the selected prime p is such that $p-1$has several small prime factors, then some values between 1 and $p-1$ do not generate groups of order $p-1$, but of subgroups of smaller orders. If the public parameter of either A or B lies within one of these small subgroups, then the shared secret key would be confined to that subgroup. The intruder may launch a brute force attack to determine the exact value of the shared secret key. The Solution to counter this kind of an attack is to choose a Safe Prime and use g that generates a large prime order subgroup or at the very least make sure that composite order subgroup are not vulnerable e.g. the order's prime number factorization contains only large primes, which we provided in our protocol, we use safe prime and we use generator with order p' which is the largest prime factor of Euler's totient function $p1$.

IV. Conclusion

In this paper we proposed a new and efficient key agreement protocol. It is secure in the sense that it meets some desirable security attributes under the assumption that the DL problem. Our protocol is more efficient and provides desirable performance attributes [5] which is, minimal number of passes because every entity sends only one message to other entity. Low communication overhead because each transmitted message has length $|P|$. Each message transmitted has the same structure (role symmetry) and are independent of each other (non-interactiveness). So our protocol can be used to improve the security in an open Internet network.

References

[1] S. B. Wilson and A. Menezes, "Authenticated diffie-hellman key agreement protocols," in *Proc. the 5th Annual Workshop on Selected Areas in Cryptography (SAC '98)*, Kingston, Canada, 1999, pp. 339-361.

[2] S. B. Wang, Z. F. Cao, Z. H. Cheng, and K. K. R. Choo, "Perfect forward secure identity-based authenticated key agreement protocol in the escrow mode," *Science in China Series F: Information Sciences*, vol. 52, no. 8, pp. 1358-1370, 2009.

[3] T. Beth, M. Frisch, and G. Simmons, *Public-Key Cryptography: State of the Art and Future Directions*, Springer-Verlag, New York, USA, 1991.

[4] Y. M. Tseng. "On the security of an efficient two-pass key agreement protocol," *Computer Standards and Interfaces*, vol. 26, no. 4, pp. 371-374, 2004.

[5] A. P. Kate, P. S. Kalekar, and D. Agrawal, "Weak keys in diffie-hellman protocol," in *Proc. Indian Institute of Technology*, Powai, Mumbai -400076, pp. 3-12, November 15, 2004.

Fatma Ahmed held a Masters' of science in Electrical Engineering from Faculty of Engineering, Alexandria University. She works on Alexandria Higher Institute of Engineering and Technology. She studies for Ph.D. in Electrical Engineering from Faculty of Engineering, Alexandria University.

Dalia Elkamchouchi held a Masters' of science in Electrical Engineering from Faculty of Engineering, Alexandria University. She works on Alexandria Higher Institute of Engineering and Technology. She Held a Ph.D. in Electrical Engineering from Faculty of Engineering, Alexandria University.