A New Scheme for Key Exchange

Debajit Sensarma[1], Subhashis Banerjee [2], Krishnendu Basuli[3]

*(West Bengal University of Technology, West Bengal, India)

** (West Bengal University of Technology, West Bengal, India)

*** (West Bengal State University, West Bengal, India)

ABSTRACT

Key management represents a major and the most sensitive part of cryptographic systems. It includes key generation, key distribution, key storage, and key deletion. It is also considered the hardest part of cryptography. Designing secure cryptographic algorithms is hard, and keeping the keys secret is much harder. Cryptanalysts usually attack both symmetric and public key cryptosystems through their key management. We introduce a protocol to exchange keys over insecure communication channel. This protocol generates keys for symmetric encryption, especially a key exchange based on Binary Decision Diagram.

Keywords - **Binary Decision Diagram, Diffie-Hellman Key exchange, Digital signature.**

1. INTRODUCTION

Authenticated key exchange (AKE) is a cryptographic protocol, which enables two or more parties to establish a shared session key over an insecure channel. Later, the shared session key can be used to ‘efficiently ensure data integrity and confidentiality by symmetric primitives.

It is desirable for an AKE protocol to have the following attributes:

i) **Known-Key Security:** Client and server should generate a unique secret key in each round of key agreement protocol. Each key generated in one protocol round is independent and should not be exposed if other secret keys are compromised.

ii) **Perfect Forward Secrecy:** If secret key is compromised, the previously established session keys are not compromised.

iii) **No Key Control:** The key should be determined jointly by both entities. None of the entities can control the key alone.

1.1 Motivation

The goal of this process is for sender and receiver to be able to agree upon a shared secret that an eavesdropper will not be able to determine. This shared secret is used by sender and receiver to independently generate keys for symmetric encryption algorithms that will be used to encrypt the data stream between them. The main aspect is that neither the shared secret nor the encryption key *do not ever* travel over the network. Here a special data structure called Binary Decision Diagram is used. Here also the advantage of Diffie-Helman key exchange is taken, as well as this approach provides integrity, authenticity and non repudiation.

1.2 Binary Decision Diagram

A Binary Decision Diagram is a rooted directed acyclic graph. It has one or two terminal nodes of out-degree zero labeled 0 or 1, and a set of variable nodes also called as branch node of out-degree two. Fig 1 depicts a BDD. The root node ‘a’ in Fig 1. Have two successors indicated by descending lines. One of the successors is drawn as a dashed line, called **‘low’** and other is drawn as a solid line, called **‘high’**. These branch nodes define a path in the diagram for any values of Boolean variables. The ‘0’ and ‘1’ nodes also called the sink node. If low branch is being followed from the root, then that path will reach to sink node ‘0’ and if high branch is being followed, then the path will reach to sink node ‘1’. The BDD obeys two important restrictions. First, it must be ordered. Second, a BDD must be reduced, in the sense that it doesn’t waste space. BDD’s are well-known and widely used in logic synthesis and formal verification of integrated circuits. Due to the canonical representation of Boolean functions they are very suitable for formal verification problems and used in a lot of tools to date [**2**].

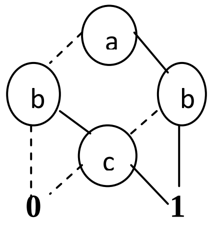


Fig 1: The Binary Decision Diagram

Diffie-Hellman Key Agreement

Public key cryptography was first publicly proposed in 1975 by Stanford University researchers Whitfield Diffie and Martin Hellman to provide a secure solution for confidentially exchanging information online. Fig 2 shows the basic Diffie-Hellman Key Agreement process.

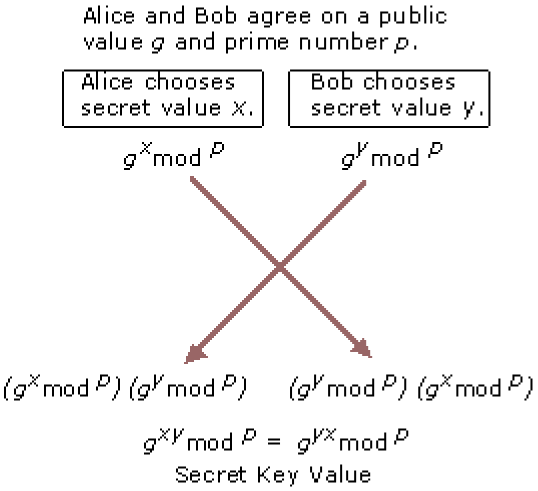

Alice and Bob agree on a public value g and prime number p.
Alice chooses secret value x.
Bob chooses secret value y.
$g^x \bmod p$
$g^y \bmod p$
$(g^x \bmod p)\ (g^y \bmod p)$
$(g^y \bmod p)\ (g^x \bmod p)$
$g^{xy} \bmod p = g^{yx} \bmod p$
Secret Key Value

Fig 2: Diffie-Hellman Key Agreement

Diffie-Hellman key agreement is not based on encryption and decryption, but instead relies on mathematical functions that enable two parties to generate a shared secret key for exchanging information confidentially online. Essentially, each party agrees on a public value *g* and a large prime number *p* . Next, one party chooses a secret value *x* and the other party chooses a secret value *y* . Both parties use their secret values to derive public values, g^x mod *p* and g^y mod *p*, and they exchange the public values. Each party then uses the other party's public value to calculate the shared secret key that is used by both parties for confidential communications. A third party cannot derive the shared secret key because they do not know either of the secret values, *x* or *y* .

For example, Alice chooses secret value *x* and sends the public value g^x mod *p* to Bob. Bob chooses secret value *y* and sends the public value g^y mod *p* to Alice. Alice uses the value g^{xy} mod *p* as her secret key for confidential communications with Bob. Bob uses the value g^{yx} mod *p* as his secret key. Because g^{xy} mod *p* equals g^{yx} mod *p* , Alice and Bob can use their secret keys with a symmetric key algorithm to conduct confidential online communications. The use of the modulo function ensures that both parties can calculate the same secret key value, but an eavesdropper cannot. An eavesdropper can intercept the values of *g* and *p* , but because of the extremely difficult mathematical problem created by the use of a large prime number in mod *p*, the eavesdropper cannot feasibly calculate either secret value *x* or secret value *y* . The secret key is known only to each party and is never visible on the network.

1.3 *RSA and DSS Approach*

Fig 3 and Fig 4 depicts the RSA approach and DSS approaches for generating Digital signature [1]. In the proposed approach M1 and M2 are transferred to the receiver with the RSA approach and the public key is transferred with DSS approach.

2. PREVIOUS WORKS

Diffe-Hellman session key agreement is the first key exchange protocol, proposed by Diffie and Hellman [8]. Diffie-Hellman key exchange by itself achieves perfect forward secrecy because no long-term keying material exists at the end of the session to be disclosed. However, it does not provide authentication of the communicating parties; hence it is vulnerable to a man-in-the-middle attack. Next, in order to fix the security flaw in the Diffie-Hellman protocol, the **Station-To-Station (STS) protocol** was proposed in [9]. To add authentication, the STS protocol requires both the parties to have a pair of public keys for signature generation and verification, and to know a publicly released symmetric key encryption. In contrast, note that the Diffie-Hellman protocol does not have these assumptions. These assumptions can be included into the protocol by sending public key certificates if the keys are not known in advance. In the STS protocol, STS protocol uses signatures to authenticate the communicating parties. It encrypts the signatures with the session key subsequently to show the knowledge of this session key. However, signatures and certificates cause the messages to increase considerably in size. Next comes **Secure Socket Layer (SSL)** [10] involves negotiating and establishing secure connections, and securing the data transmission. SSL handshake uses certificates and PKI [13] for mutual authentication and key exchange. PKI binds public keys with particular user identities by means of a certificate authority (CA). The CA is the trusted entity that signs and issues digital certificates [11] to other parties. A digital certificate contains a public key and the identity of the owner and the validity period of the certificate. Therefore, authentication is performed through sending and verifying certificates which involve a great overhead. SSL key exchange can use an RSA algorithm, an asymmetric technique for session key exchange which encrypts the session key from the client to the server. A Diffie-Hellman key exchange can also be used which is more secure since both parties agree on the session key without having to send the key across the wire. Another protocol is **ID-based Authenticated Key Agreement.** Many protocols were proposed for ID-KEX [12]. Paterson and Price [14] noted that the aim in designing a good ID-KEX protocol is to achieve all the properties of the best usual key agreement protocols while trying to maximize efficiency. The public key can be chosen by any client in the system as it is generated from public information (email address or network address). Each party, then contacts the trusted authority (TA) once to authenticate and get the required private key. A key agreement protocol is said to be authenticated if it offers the guarantee that only the participating parties of the protocol can compute the agreed key. Therefore, this ID-KEX protocol is authenticated because it uses public and private keys to generate a shared secret. Another protocol **SSH (Secure SHell)** is a secure network protocol used by the user to log into a remote computer running an SSH server [15]. It was designed to replace telnet which is an earlier protocol that passes username and password in plain text. However, SSH provides a secure transmission by encrypting the authentication strings and all the other data exchanged between the hosts. Ylonen and Lonvick explored three layers of the SSH protocol; the Transport Layer Protocol provides host authentication, confidentiality (encryption), and integrity; the User Authentication Protocol authenticates the client-side user to the server and provides a number of authentication methods; and the Connection Protocol [16] multiplexes the encrypted tunnel into several logical channels. SSH supports both password authentication and public key authentication. Although passwords are convenient and they require no additional configuration or setup from the users, they can be guessed, and the hacker can get into the system. Public key authentication provides better security as every machine creates a public/private key by itself. SSH clients and servers maintain and check a database containing identifications for all the hosts that have been involved in the interactions. Therefore, the first time when a user connects to a remote entity, the user has to know or trust that the key fingerprint for that entity is correct as SSH does not practice a central authority to assure access for each entity.

The Diffie-Hellman key-exchange protocol has been the subject of many works. Canetti and Krawczyk [17] analyzed key-exchange protocols (Diffie-Hellman and key-transport) authenticated via symmetric or asymmetric techniques to obtain the proof of security. In [18] they presented a security analysis of the Diffie- Hellman key exchange protocol authenticated with digital signatures used by the Internet Key Exchange (IKE) standard. Lee, Malkin, and Nahum [20] focused on the different parts of SSL such as the strength of SSL/TLS servers; Castelluccia, Mykletun, and Tsudik [19] analyzed the performance of SSL/TLS Handshakes and suggested an improvement. Moreover, the performance of pre-shared and Public Key Exchange Mechanisms for TLS protocol has been reported by Kuo, Tschofenig, Meyer, and Fu [21].

3. PROPOSED APPROACH

Suppose ‘A’ wants to exchange key with ‘B’.

Step 1: ‘A‘sends a message M1 of variable length and M2 with RSA approach (Fig: 3) to ‘B’. Here M2 is a prime number. ‘A’ and ‘B’ both share the same ‘secret table’.

Step 2: ‘A’ picks a secret number “a”; ‘B’ picks a secret number “b”.

Step 3: ‘A’ picks up first 8 bit from M1, suppose the number is ‘m’ and computes public number

x = m^a mod M2. ‘B’ computes public number **y** = m^b mod M2.

Step 4: ‘A’ and ‘B’ exchange their public numbers by DSS approach (Fig: 4). ‘A’ knows M1, m, M2, a, x, y. ‘B’ knows M1, m, M2, b, x, y.

Step 5:’A’ computes ka1= y^a mod M2, Ka2=y^{ka1} mod M2.

{ka1= $(m^b \bmod M2)^a$ mod M2

ka1= $(m^b)^a$ mod M2

ka1= m^{ba} mod M2 }

‘B’ computes kb1= x^b mod M2, kb2= x^{kb1} mod M2

{kb1= $(m^a \bmod M2)^b$ mod M2

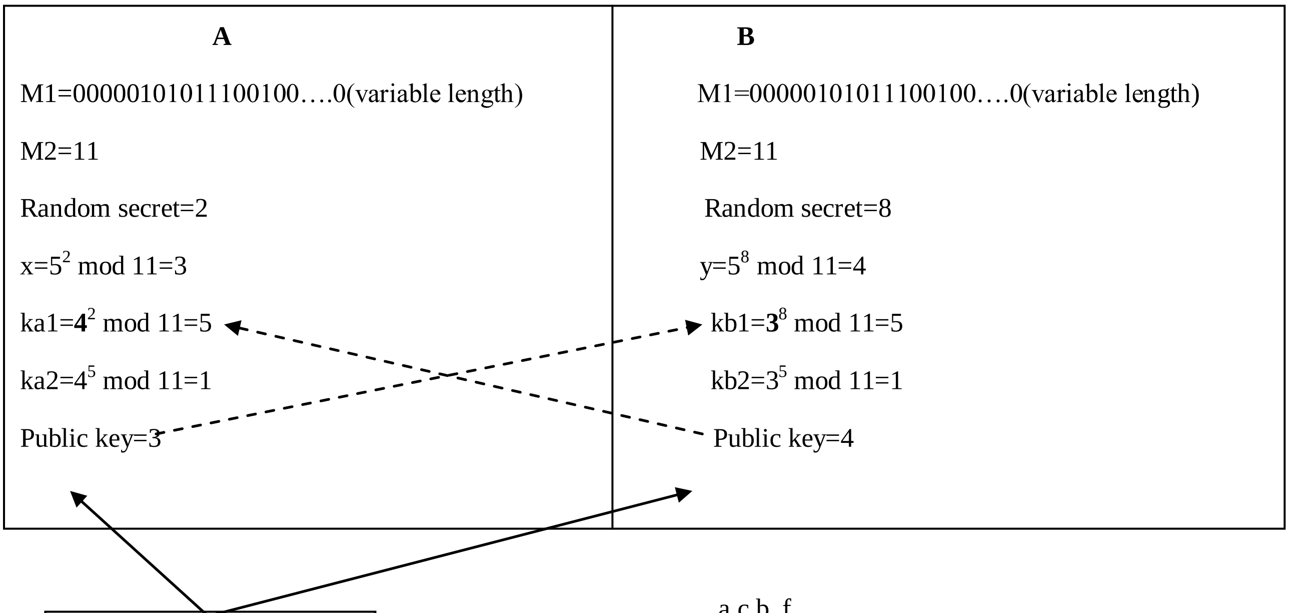
kb1= $(m^a)^b$ mod p

kb1= m^{ab} mod p}

Step 5: x represents the number of variables in the secret table, ka1 and kb1 represents the starting bit position in M1 which will be treated as the temporary key and lastly ka2 and kb2 represents the permutation number in the secret table.

Step 6: From the above information a Binary Decision Diagram is build with alphabetic variable ordering. Lastly from the codes of the variable in the secret table ‘A’ and ‘B’ will find their **secret key**, which is same for both ‘A’ and ‘B’.

3.1 Example



A	B
M1=00000101011100100….0(variable length)	M1=00000101011100100….0(variable length)
M2=11	M2=11
Random secret=2	Random secret=8
$x=5^2 \bmod 11=3$	$y=5^8 \bmod 11=4$
ka1=$\mathbf{4}^2 \bmod 11=5$	kb1=$\mathbf{3}^8 \bmod 11=5$
ka2=$4^5 \bmod 11=1$	kb2=$3^5 \bmod 11=1$
Public key=3	Public key=4

Permutation	codes
1. a c b	a=00
2. b a c	b=100
3. c a b	c=1100
4. a b c	
5. c b a	
6. b c a	

Secret Table

Here m= $(00000101)_2= (5)_{10}$.

Number of variables=3

Starting bit position of temporary key=5

Permutation number=1(which is a,c,b in this case).

In this case temporary key length will be 8(2^3) and it will be **01010111**.

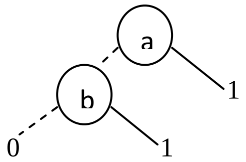
So, with this ordering and temporary key a truth table is generated,

a	c	b	f
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

After ordering variables in alphabetical order, the truth table becomes,

a	b	c	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

With this variable ordering the Binary Decision diagram will be,



From the secret table,

a=00

b=100

So, **Secret Key=00100**

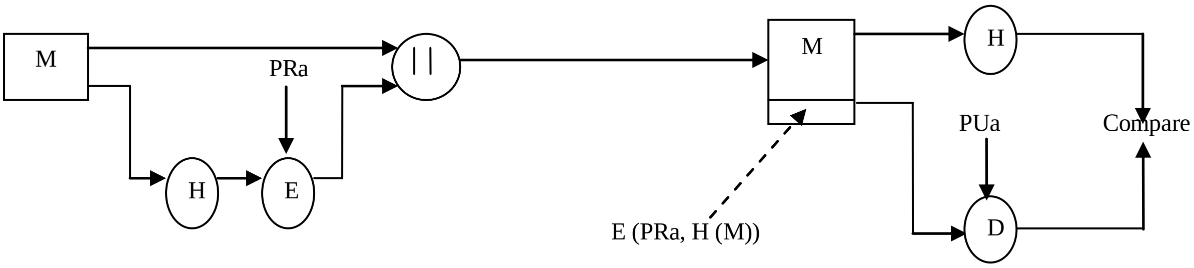


Fig 3: RSA approach

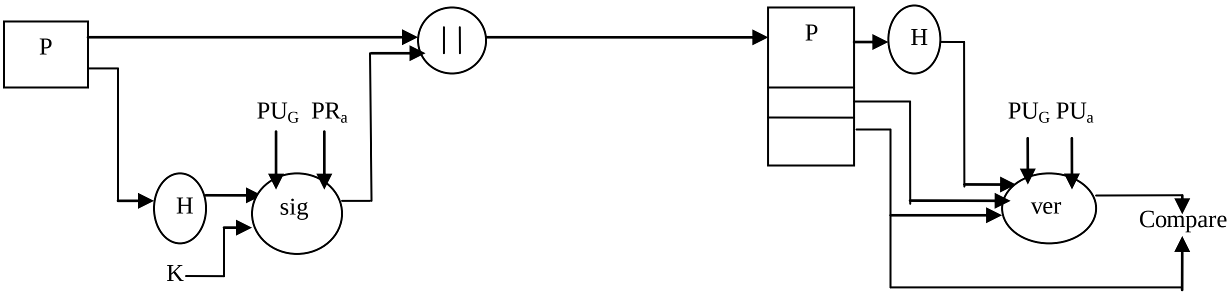


Fig 4: DSS approach

4. CONCLUSION

Designing secure cryptographic algorithms is hard, and keeping the keys secret is much harder. Cryptanalysts usually attack both symmetric and public key cryptosystems through their key management. We introduced a protocol to exchange keys over insecure communication channel. This approach takes the advantage of **Diffie-Hellman** key exchange and as well as provides integrity, authenticity and non repudiation when transferring the message and public key.

5. ACKNOWLEDGMENTS

The authors would like to thank West Bengal State University, West Bengal, India, West Bengal University of Technology, West Bengal, India. The authors also thank the reviewers for their constructive and helpful comments and specially the Computer without which no work was possible.

6. REFERENCES

[1] Willum Stalligs, Cryptography and Network Security principles and practice, Fifth edition,pearson,2011.

[2] Debajit Sensarma, Subhashis Banerjee, Krishnendu Basuli,Saptarshi Naskar, Samar Sen Sarma "On an optimization technique using Binary Decision Diagram", International Journal of Computer Science, Engineering and Applications (IJCSEA), Volume 2, Number 1,73-86,February 2012.

[3] Whitfield and Martin E. Hellman, " New directions in cryptography", IEEE transactions in Information theory, IT-22(6), pp 644-654, Nov., 1966.

[4] Henrik Reif Andersen. "An Introduction to Binary Decision Diagrams". Lecture Notes (Technical University of Denmark, October 1997).

[5] R.E. Bryant. "Graph-based algorithms for Boolean function manipulation". IEEE Transactions on Computers, C-35-8, pp.677-691, August, 1986.

[6] RANDAL E, BRYANT. "Symbolic Boolean Manipulation with Ordered Binary-Decision Diagrams". ACM Computing Surveys, 1992.

[7] Donald E. Knuth,"The Art of Computer Programming", Volume 4, Addison-Wesley.

[8] Diffie, W., Hellman, M.E., New directions in cryptography, 1976.

[9] Diffie, W., Van Oorschot, P.C., Wiener, M.J. Authentication and authenticated key exchanges. Des. Codes Cryptography 2(2), 107-125, 1992.

[10] Frier, A., K.P., Kocher, P.. The secure socket layer. Technical report, Netscape Communications Corp,1996.

[11] Feghhi, J., Feghhi, J., Williams, P.. Digital Certi_cates: Applied Internet Security. Addison Wesley Longman,1999.

[12] Shim, K.. Efficient id-based authenticated key agreement protocol based on the weil pairing. Electronics Letters 39(8), 653-654, 2003.

[13] Younglove, R.. Public key infrastructure. how it works. Computing & Control Engineering Journal 12, 99-102, 2001.

[14] Paterson, K., Price, G.. A comparison between traditional public key infrastructures and identity-based cryptography. Information Security 8(16), 57-72, 2003.

[15] Barrett, D., Silverman, R.. SSH: The Secure Shell (The Definitive Guide). O'Reilly, 2nd edition edn, 2005.

[16] Ylonen, T., Lonvick, C.E.. The secure shell (ssh) connection protocol, rfc 4254,2006.

[17] Canetti, R., Krawczyk, H.. Analysis of key-exchange protocols and their use for building secure channels. In: EUROCRYPT '01: Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques. pp. 453-474. Springer-Verlag, London, UK, 2001.

[18] Canetti, R., Krawczyk, H.. Security analysis of ikes signature-based key-exchange protocol. In: In: Proc. CRYPTO02, Springer LNCS 2442. pp. 143-161. Springer- Verlag,2002.

[19] Castelluccia, C., Mykletun, E., Tsudik, G.. Improving secure server performance by re-balancing ssl/tls handshakes. In: in Proceedings of the 10th Annual USENIX Security Symposium. pp. 26-34, 2005.

[20] Lee, H.K., Malkin, T., Nahum, E.. Cryptographic strength of ssl/tls servers: cur- rent and recent practices. In: IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement. pp. 83-92. ACM, New York, NY, USA, 2007.

[21] chun Kuo, F., Tschofenig, H., Meyer, F., Fu, X.. Comparison studies between pre- shared and public key exchange mechanisms for transport layer security. In: 25th IEEE International Conference on Computer Communications. pp. 1-6, 2006.