ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 1, Issue 2, November 2012

Diffie-Hellman and Its Application in Security Protocols

Maryam Ahmed, Baharan Sanjabi, Difo Aldiaz, Amirhossein Rezaei, Habeeb Omotunde

Abstract -With the wide use of the Internet, virtually everyone is now connected to each other through their computers. This has led to a positive impact in the human environment socially, economically and in their day-to-day transactions. There is, however, a major hindrance in trying to establish an effective and safe communication line: an outside user, not intended to be a part of the connection, might try to steal the information being passed to a legitimate user. This being a security issue, information security therefore plays a vital role in Internet transactions. It can be deduced that secure digital communication is necessary for many aspects relating to web based activities, e-commerce, and secured instant messaging. More so for private, confidential, and vital information, the reality that safe, secure communication between parties communicating over the Internet is now a necessity cannot be overstated. Cryptography is an indispensable tool for protecting information in computer systems. Today's cryptosystems are divided into two categories: symmetric and asymmetric. The difference lies in the keys used in decryption and encryption—symmetric cryptography uses the same key for both of these processes, whereas asymmetric cryptosystems use one key (the public key) to encrypt a message and a different key (the private key) to decrypt it. The Diffie-Hellman key exchange is one of the more well-known asymmetric algorithms, formulated by its namesakes Whitfield Diffie and Martin Hellman in 1976. It is referred to in various ways, e.g. Diffie-Hellman protocol, Diffie-Hellman handshake, or Diffie-Hellman key negotiation, and commonly shortened to D–H, or DH, for convenience.

Keywords: **Cryptography, Diffie-Hellman, SSH, IP sec, SSL, Key Exchange**

I. INTRODUCTION

For information to be secure, one has to be prevent access to it from unauthorised users (the principle of *confidentiality*), prevent it from undergoing unwanted changes (*integrity*), and ensure that it is available to its intended users (*availability*). This can be guaranteed by means of protocols that make use of security primitives such as encryption, digital signatures and hashing [6].

The first researchers to formulate and publish the concepts of *public-key cryptography* (PKC) were Whitfield Diffie and Martin Hellman, both from Stanford University, in parallel with Ralph Merkle, from the University of California at Berkeley. Specifically, Diffie and Hellman worked on public key cryptography while Merkle made his contributions on public key distribution [7]. When they became aware of each other's work they decided to work together in hope for better results. This later led to the publication of their joint paper, titled "New Directions in Cryptography"—published in November 1976 [2]. This paper brought a new idea to the field of cryptography; it described the key concepts of public-key cryptography such as the production of digital signatures, and gave some algorithms for implementation.

The idea of public key cryptography was born as a result of two major challenges. The first of these was the problem of key distribution: if two people who have never met before are to communicate using digital systems as a medium, using conventional cryptography would mean that they must somehow agree on a common key that will be known to themselves and no one else. The other problem was the issue of signatures: this is a method of providing the recipient of a purely digital electronic message with a way of demonstrating to other people that it had come from a particular person, serving as a signature comparable to a written one on a letter.

II. DESCRIPTION

The Diffie-Hellman algorithm can be described formally in relatively simple mathematics. The algorithm itself does not encrypt data, but instead it generates a secret key common to both the sender and the recipient. Although they never agreed on using a particular key, through mathematically linked processes the two parties can independently generate the same secret key and then use it to build a session key for use in asymmetric algorithm [2]. This procedure is called *key agreement*, meaning that the two parties are agreeing on a key to use.

The protocol has two system parameters p and g. They are both public, and may be used by all users in a system. The first of these, parameter p, is a randomly selected prime number (hence "p") while parameter g (this time for "generator") is an integer less than p, but also with the following property: for every number n between 1 and p - 1 (inclusive), there exists a power k of g such that $n = g^k \bmod p$. [4]

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 1, Issue 2, November 2012

In the following illustration, two parties, traditionally called Alice and Bob, want to agree on a shared secret key using the Diffie-Hellman key agreement protocol discussed throughout this report:

At first, Alice generates a random, private value *a* while Bob does the same with *b*. Both of these are drawn from the set of integers {1, ¼, *p* - 2}. They will then combine these private values with *p* and *g* to derive their respective public keys. Alice's public value is $x = g^a \bmod p$, while Bob's is, of course, $y = g^b \bmod p$.

To proceed, the parties will send their public key to one another, in a process of key exchange. Even though both had kept their private values secret, Alice will compute $k_a = y^a \bmod p$, and Bob computes $k_b = x^b \bmod p$. Laws of algebra dictate that $k_a = k_b = k$, and so, Alice and Bob now have a common secret key *k*.

The protocol depends on the discrete logarithm problem for its security. It is relatively safe to assume that to calculate the shared secret key *k* from the known public values alone will prove to be computationally infeasible when the prime *p* is sufficiently large. As shown by Maurer before, breaking the Diffie-Hellman exchange is, under certain assumptions, equivalent to computing discrete logarithms. In that sense, then, it is quite safe—in the real world of course it will be coupled with several other measures, but it would be good enough to be included among those measures.

Using a table, one can summarise the process as follows:

Table 1: The Diffie-Hellman key exchange.

Process	Parameters
Alice and Bob agree on two numbers: **p** and **g**	**p** is a large prime number **g** is the generator, or the base
Alice privately picks a secret number **a**	Alice's secret number = **a**
Bob privately picks a secret number **b**	Bob's secret number = **b**
Alice computes her public key $\mathbf{x} = \mathbf{g}^{\mathbf{a}} \bmod \mathbf{p}$	Alice's public number = **x**
Bob computes his public key $\mathbf{y} = \mathbf{g}^{\mathbf{b}} \bmod \mathbf{p}$	Bob's public number = **y**
Alice and Bob exchange their public keys, to be used for private generations of a common secret key.	(Alice knows **p, g, a, x, y**) (Bob knows **p, g, b, x, y**)
Alice computes the secret key $\mathbf{k_a} = \mathbf{y}^{\mathbf{a}} \bmod \mathbf{p}$	$\mathbf{k_a} = (\mathbf{g}^{\mathbf{b}} \bmod \mathbf{p})^{\mathbf{a}} \bmod \mathbf{p}$ $\mathbf{k_a} = (\mathbf{g}^{\mathbf{b}})^{\mathbf{a}} \bmod \mathbf{p}$ $\mathbf{k_a} = \mathbf{g}^{\mathbf{b}\,\mathbf{a}} \bmod \mathbf{p}$
Bob computes the secret key $\mathbf{k_b} = \mathbf{x}^{\mathbf{b}} \bmod \mathbf{p}$	$\mathbf{k_b} = (\mathbf{g}^{\mathbf{a}} \bmod \mathbf{p})^{\mathbf{b}} \bmod \mathbf{p}$ $\mathbf{k_b} = (\mathbf{g}^{\mathbf{a}})^{\mathbf{b}} \bmod \mathbf{p}$ $\mathbf{k_b} = \mathbf{g}^{\mathbf{a}\,\mathbf{b}} \bmod \mathbf{p}$
By the laws of algebra, Alice's key $\mathbf{k_a}$ is the same with Bob's key $\mathbf{k_b}$. Or, $\mathbf{k_a} = \mathbf{k_b} = \mathbf{k}$	Now Alice and Bob both know the secret value **k**

A more succinct way of understanding it is illustrated by figure 1, where Alice here explicitly works as a host who initiates the *p* and *g* values to be used by Bob and herself.

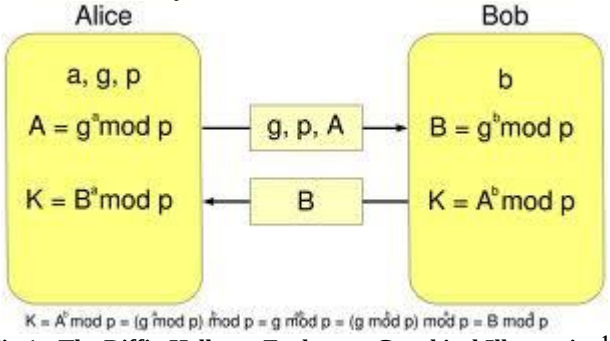

Fig 1: The Diffie-Hellman Exchange, Graphical Illustration[11].

III. ISSUES: MAN-IN-THE-MIDDLE ATTACKS

This algorithm, particularly in its early forms, has a major weakness in the form of man-in-the-middle vulnerability [9]. In this attack, a malicious third party, commonly referred to as "Eve" (for "eavesdropper") retrieves Alice's public key and sends her own public key to Bob. When Bob transmits his public key, Eve interrupts and substitutes the value with her own public key and then sends it to Alice. By now, Alice would

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 1, Issue 2, November 2012

have come to an agreement on a common secret key with Eve instead of Bob. This exchange can be done in reverse, and it is possible for Eve to decrypt any messages sent out by Alice or Bob, and then read and possibly modify them before the re-encryption with the appropriate key and transmitting them to the other party.

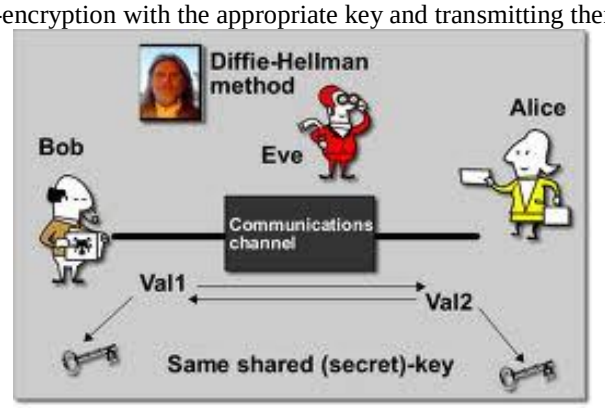

Fig 2: Man-In-The-Middle Attack on A D-H Key Exchange [1]

To address this problem, generally a process of authentication will be needed to ensure that, whenever Alice wishes to send a message to Bob, the recipient can only be Bob and not an Eve, and vice versa. It is also important—and generally the norm—to discard the keys after usage, so that there will be no long-term keys that can be disclosed to cause problems in the future [4].

Other concerns typically revolve around optimising the mathematics involved. That is, to properly generate the randomly selected values so that they are (1) large enough to achieve computational infeasibility for attackers, and (2) random enough, as pseudorandom numbers can greatly ease Eve due to their eventual predictability.

IV. USAGE IN SECURE INTERNET PROTOCOLS

The Diffie-Hellman protocol has been applied to many security protocols including the *Security Sockets Layer (SSL)*, *secure shell (SSH)*, and *IP Sec*.

A. Secure Sockets Layer (SSL)

The SSL is the standard security technology developed by Netscape in 1994 to establish an encrypted link between a web server and a browser. This link ensures privacy and integrity of all data passed between the web server and browsers. SSL is used by millions of websites in the protection of their online transactions with their customers [9].

SSL is all about encryption. SSL uses certificates, private/public key exchange pairs and Diffie-Hellman key agreements to provide privacy (key exchange), authentication and integrity with *Message Authentication Code (MAC)*. This information is known as a *cipher suite* and exists within a *Public Key Infrastructure (PKI)*.

SSL is useful for business/financial traffic, e.g. credit card transactions. SSL ensures confidentiality (it prevents eavesdropping), authenticity (the sender is really who he says he is), and integrity (the message has not been changed *en route*). It is possible that a user might not know SSL is used in the course of communication but they are likely to notice some blockages.

SSL/TLS is composed of two layers: the lower layer, called the *record protocol*, rides on TCP and manages the symmetric (private) cryptography so the communication is private and reliable. The upper layer is called the *handshake protocol* and it is in this layer that D-H is used. The handshake allows the server to authenticate itself to the client using public-key techniques, also called asymmetric encryption. It also allows the client and the server to cooperate in the creation of symmetric keys, which are used for rapid encryption and decryption. This implies that while communication is in progress the client and server exchange unencrypted handshake messages that include hellos and then information about which encryption, key exchange, and compression options they each accept and prefer [8].

During the SSL handshake, each computer generates a set of codes to encrypt information. From these codes, each computer creates two keys, one private and one public. Your computer keeps the private key secret, but sends out the public key to the other computer, which uses that key to encode subsequent messages so that only your computer can read them. The public key cannot, however, be used to decode the message; the decoding can only be done using the private key. These keys allow you and the other computer to lock and unlock information so that only the holder of the private key can read messages encrypted by the public key. Since only you and the other computer have a copy of your respective private keys, there is no way for anybody else to intercept and decode your messages. This is in agreement with the methods described earlier in section 3.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 1, Issue 2, November 2012

In SSL, the key exchange process uses D-H algorithm which is asymmetric (that is, public key) cryptography to ensure to each party that the other is who they say they are. After this exchange, keys are computed and the parties begin encrypting all traffic between them, using the computed keys and agreed upon methods.

B. Secure Shell (SSH)

SSH is a network security protocol very common for secure remote login on the Internet. The secure shell has come to replace the unsecured Telnet on the network and FTP on the system, mostly because both Telnet and FTP do not encrypt data, and instead send them in plaintext. SSH, on the other hand, can automatically encrypt, authenticate and compress transmitted data [10].

The key exchange protocol itself is a component of the SSH as a whole, particularly responsible for parties agreeing upon the keys used by the various primitives later in the SSH protocol. [2] This is the first stage of the SSH algorithm, and it happens before the establishment of session keys.

The protocol proceeds in three stages. The first of these is the "*Hello*" phase, where the first identification is done. A list of supported algorithms is involved here after the first "*Hi*" message, and this list details the supported Diffie-Hellman key groups, among other things. The second stage sees the two parties agree upon a shared secret key *x*, which is done by an implementation of a Diffie-Hellman exchange. At the final stage, the shared secret key, session identifier and digest are used to generate the application keys. [5]

Currently, the "*diffie-hellman-group1-sha1*" method is practised in the key exchange, prescribing a fixed group on which all operations are performed. The key exchange is then signed with the host key to provide host authentication.

C. IP Security (IPSec)

IPSec (IP Security) is an extension of the *Internet Protocol (IP)*—it is a suite of protocols introduced by the *Internet Engineering Task Force (IETF)* to aid in configuring a communications channel between multiple machines. Operating at the IP layer of the seven-layer model, it does its job by authenticating and encrypting IP packets [11].

Like the previous protocols, IPSec uses D-H and asymmetric cryptography to establish identities, preferred algorithms, and a shared secret. Before IPSec can begin encrypting the data stream, some preliminary information exchange is necessary. This is accomplished with the *Internet Key Exchange (IKE)* protocol. IKE uses DH to produce a shared secret via the usual mechanisms, and then authenticate each other; after that, the secret key is used for encryption purposes. This shared secret key is never exchanged over the insecure channel [8].

V. SUMMARY AND CONCLUDING STATEMENTS

The Diffie-Hellman key exchange exploits mathematical properties to produce a common computational result between two (or more) parties wishing to exchange information, without any of them providing all the necessary variables. By agreeing on two variables and providing each other with a computed public key, the resulting secret key will be identical throughout the exchange.

It is, of course, possible to intervene by either masquerading or by sheer brute force, but the first is a common concern — authentication — which must be addressed separately, and the second is, when done right, computationally infeasible (hence the alternative name "exponential key exchange"). With proper authentication mechanisms, proper prime generation, and true randomness in picking variables, the D-H protocol can be a powerful component in many a security measure. Good implementations include usages in Secure Sockets Layer, Secure Shells, IP Security, and others.

Passages on short introductory and background information, description, issues, as well as common usages were produced in this report, and it is hoped that they were all of sufficient clarity.

REFERENCES

[1] Burnett, S. and Paine, S. (2001) RSA Security's Official Guide to Cryptography. McGraw-Hill.

[2] Carts, D.A. (2001) A Review of the Diffie-Hellman Algorithm and its Use in Secure Internet Protocols. [Online] Available at http://www.sans.org/reading_room/whitepapers/vpns/review-diffie-hellman-algorithm-secure-internet-protocols_751 [Accessed 25 October 2012].

[3] Choo, K.K.R. (2009) Secure Key Establishment. New York: Springer.

[4] EMC Corporation (2012) "What Is Diffie-Hellman?", RSA Laboratories [Online] Available at http://www.rsa.com/rsalabs/node.asp?id=2248 [Accessed 26 October 2012].

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 1, Issue 2, November 2012

[5] Friedl, M., Provos, N., and Simpson, W. (2006) Diffie-Hellman Group Exchange for the Secure Shell (SSH) Transport Layer Protocol. [Online] Available at http://www.ietf.org/rfc/rfc4419.txt [Accessed 14 March 2012].

[6] Kahate, A. (2008) Cryptography and Network Security. McGraw-Hill.

[7] Living Internet (1996-2012) "Public Key Cryptography (PKC) History", Living Internet. [Online] Available at http://www.livinginternet.com/i/is_crypt_pkc_inv.htm [Accessed 16 March 2012].

[8] Oracle (2008-12) "Diffie-Hellman", Oracle Think Quest. [Online] Available at http://library.thinkquest.org/C0126342/dh.htm [Accessed 12 March 2012].

[9] Raymond, J.F. and Stiglic, A. (2000) Security Issues in the Diffie-Hellman Key Agreement Protocol. [Online] Available at http://crypto.cs.mcgill.ca/~stiglic/publications.html [Accessed 17 March 2012].

[10] Williams, S. (2011) Analysis of the SSH Key Exchange Protocol. [Online] Available at http://eprint.iacr.org/2011/276.pdf [Accessed 15 March 2012].

[11] Florian Tegeler, (2008) Security Analysis, Prototype Implementation and Performance Evaluation of a New Ip Sec Session Resumption Method. [Online] Available at: http://www.net.informatik.uni-goettingen.de/publications/1512/FTegeler_MScThesis.pdf [Accessed 29 October 2012].