CENTRE FOR ADVANCED STUDIES

Dr. A. P. J. Abdul Kalam Technical University,
Lucknow

Diffie-Hellman: Key Exchange and Public Key Cryptosystems

PRATISHTHA SAXENA

Submitted to: Vrijendra Singh,
Associate Professor,
IIIT Allahabad.

Contents

1. Abstract

Diffie-Hellman algorithm is one of the first schemes proposed for the exchange of keys required in asymmetric encryption. It was developed by Whitfield Diffie and Martin Hellman in 1976. This algorithm removes the need of transferring keys between two communicating parties. It enables each party to generate a shared secret key for encryption and decryption of data. The security of this algorithm cannot be compromised because several security protocols and services depend upon Diffie-Hellman key exchange for reliable communication. In this paper, we have used a random parameter to make this algorithm more efficient. The random parameter generates new shared keys for each message that is exchanged between sender and receiver. So, different ciphertext will be produced each time even for the same message. Thus, systems using this scheme will become more tolerant to various attacks.

2. History

The primary researchers to find and publish the ideas of Public Key Cryptology were Whitfield Diffie and Martin Hellman from Stanford University, and Ralph Merkle from the University of California at Berkeley. As so frequently happens in the experimental world, the two gatherings were working autonomously on the same issue - Diffie and Hellman on public key cryptography and Merkle on public key distribution - when they got to know about one another's work and acknowledged there was collaboration in their methodologies. In Hellman's words: "We each had a key piece of the puzzle keeping in mind it's actual one of us first said X, and another of us first said Y, and so on, it was the combination of forward and backward between us that permitted the disclosure."

The first published work on Public Key Cryptography was in a Ground - breaking paper by Whitfield Diffie and Martin Hellman titled "New Directions in Cryptography" in the November, 1976 version of IEEE Transactions on Information Theory, and which additionally referenced Merkle's work. The paper depicted the key ideas of Public Key Cryptography, including the generation of digital signatures and gave some algorithms for execution. This paper revolutionized the world of cryptography research, which had been to some degree controlled up to that point by genuine and saw Government confinements and aroused many analysts around the globe to take a shot at down to earth executions of a public key cryptography algorithms.

3. Diffie-Hellman Key Exchange

The purpose of the algorithm is to enable two users to securely exchange a key that can then be used for subsequent encryption of message. The algorithm is limited to the key exchange of secret values.

The Diffie-Hellman algorithm depends for its effectiveness of the difficulty of computing discrete logarithms. Briefly, we can define the discrete logarithms in the following way,

First, we define a primitive root of a prime number p as one whose power modulo p generate all the integers from 1 to p-1. That is, if a is a primitive root of prime number p, then the numbers

a mod p, $a^2\ mod\ p$, ……….. , a^{p-1} mod p

are distinct and consist of the integers from 1 through p-1 in some permutation.

For any integer b and a primitive root a of prime number p, we can find a unique exponent I such that

B Ξ a^i (mod p) where 0 ≤ I ≤ (p-1)

The exponent i is referred to as the discrete logarithm of b for the base a, mod p. We express this value as $dlog_{a,p}$(b).

4. The Algorithm (steps)

1. Alice chooses a large random number x such that 0 ≤ x ≤ p-1 and calculates R1= g^x mod p.
2. Bob chooses another large random number y such that 0 ≤ y ≤ p-1 and calculates R2= g^y mod p.
3. Alice sends R1 to Bob. Note that Alice does not send the value of x; she sends only R1.
4. Bob sends R2 to Alice. Note that Bob does not send the value of y; he sends only R2.
5. Alice calculate K= $(R2)^x$ mod p.
   (a) K = $(g^y\ mod\ p\ )^x$ mod p
6. Bob calculates K= $(R2)^y$ mod p.
   (a) K = $(g^x\ mod\ p\ )^y$ mod p

K is the symmetric key for session.

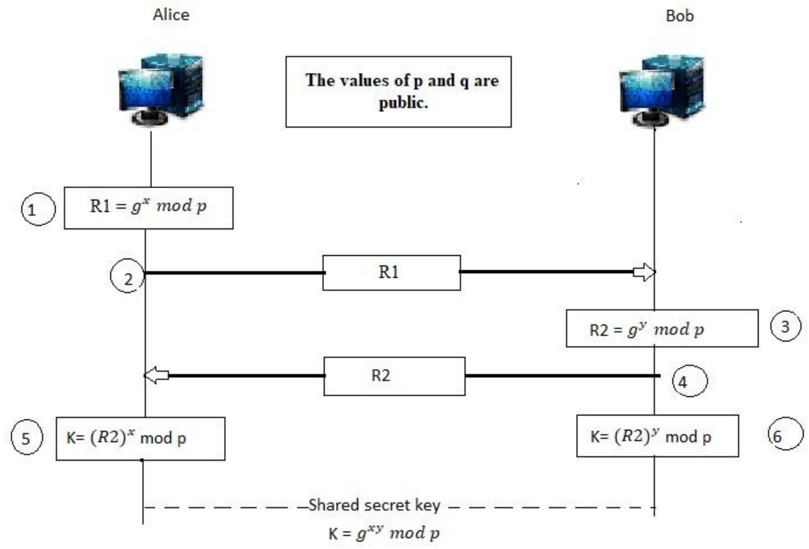

Fig 1

5. Diffie-Hellman Correctness and its Proof

1. Alice has computed

`

R1 = g^x mod p

K1 = $R2^x$ mod p

2. Bod has computed

R2 = g^y mod

K2 = $R1^y$mod p

3. Alice has

K1 = $R2^x$mod p

= $(g^y)^x$mod p

= $(g^x)^y$mod p

= $R1^y$mod p

4. Bod has

K2 = $R1^y$mod p

= $(g^x)^y$mod p

= $(g^y)^x$mod p

= $R2^x$mod p

5. Therefore

$$K1 = K2$$

6. Illustrate Examples

Examples 6.1:

1. Alice and Bob agree on p = 23 and g = 5.
2. Alice chooses a = 6 and sends

$$5^6 \bmod 23 = 8$$

3. Bob chooses b = 15 and sends

$$5^{15} \bmod 23 = 19$$

4. Alice computes

$$19^6 \bmod 23 = 2$$

5. Bob computes

$$8^{15} \bmod 23 = 2$$

Then 2 is the shared secret.

Example 6.2:

Let's assume that Alice wants to establish a shared secret with Bob.

1. Alice and Bob agrees on a prime number, p, and a base, g, in advance. For our example, let's assume that p=23 and g=5.

2. Alice chooses a secret integer a whose value is 6 and computes

$$A = g^x \bmod p = 8$$

3. Bob chooses a secret integer b whose value is 15 and computes

$$B = g^y \bmod p = 19$$

4. Alice sends A to Bob and Bob sends B to Alice.
5. To obtain the shared secret, Alice computes

$$s = B^x \bmod p = 2$$

6. To obtain the shared secret, Bob computes

$$s = A^y \bmod p = 2$$

The algorithm is secure because the values of a and b, which are required to derive s are not transmitted across the wire at all.

7. Issues and Security

7.1 Issues

7.1.1 *Man-In-The-Middle Attacks*

The protocol depicted in Fig 2 is insecure against man-in-the-middle attack. Suppose Alice and bob wish to exchange keys, and Darth is the adversary. The attack proceeds as Follows.

1. Darth prepares for the attack by generating two random private keys X_{D1} and X_{D2}, and then computing the corresponding public keys $Y_{D1}\ and\ Y_{D2}$.
2. Alice transmit Y_A to Bob.
3. Darth intercepts Y_A and transmits Y_{D1} to Bob. Darth also calculate K2 = $(Y_{D1}\ )^{X_{D2}}$ mod q.
4. Bob receives Y_{D1} and calculates K1 = $(Y_{D1}\ )^{X_B}$ mod q
5. Bob transmit X_A to Alice.
6. Darth intercepts X_A and transmits Y_{D2} to Alice. Darth calculate K = $(Y_B\ )^{X_{D1}}$ mod q.
7. Alice receives Y_{D2} and Calculate k2 = $(Y_{D2}\ )^{X_A}$ mod q.

At this point, Bob and Alice think that they share a secret key, but instead Bob and Darth share key K1 and Alice and Darth share secret key K2. All future communication between Bob and Alice is compromised in following way:

Alice sends an encrypted message M: E(K2, M).
Darth intercepts the encrypted message and decrypts it, to recover M.
Darth sends Bob E (K1, M) or E (K1, M'), where M' is any message. In the first case, Darth simply wants to eavesdrop on the communication without altering it. In the second case, Darth wants to modify the message going to Bob.

The key exchange protocol is vulnerable to such an attack because it does not authenticate the participants.

7.1.2 *Discrete Logarithm Attack*

The security of the key exchange is based on the difficulty of the discrete logarithm problem. Darth can intercept R1 and R2. If she can find x from R1 = g^x mod p and y from R2 = g^y mod p, then she can calculate the symmetric key K= g^{xy} mod p. The secret key is not secret anymore. To make Diffie-Hellman safe from the discrete logarithm attack, the following are recommended.

1. The prime p must be very large (more than 300 decimal digits).

2. The prime must be chosen such that p-1 has at least one large prime factor (more than 60 decimal digits).
3. The generator must be chosen from the group <Z_{p*},X>.
4. Bob and Alice must destroy x and y after they have calculated the symmetric key. The values of x and y must be used only once.

7.2 Security Against Attacks

The basic Diffie-Hellman protocol we have shown is not secure against a man-in-the-middle attack. In fact, impossible to achieve security against such an attacker unless some information is shared in advance E.g., private- key setting or public-key setting. To address this issue, generally a process of authentication will be expected to guarantee that, at whatever point Alice wishes to send a message to Bob, the beneficiary must be Bob and not an Eve, and the other way around. It is also important - and generally the norm - to discard the keys after use, so that there will be no long - term keys that can be revealed to bring about issues later on. Different concerns ordinarily rotate around upgrading the mathematics involved. That is, to properly generate the randomly choose values with the goal that they are
(1) large enough to achieve computational infeasibility for attackers, and
(2) random enough, as pseudo-random numbers can greatly ease Eve due to their eventual predictability.

"Generally talking, the fundamental thought is as per the following. Preceding execution of the protocol, the two gatherings Alice and Bob each acquire a public/private key pair and a certificate for the public key. During the protocol, Alice computes a signature on certain messages, covering the public value g^xmod p , Bob proceeds in a similar way. Despite the fact that Eve is still ready to intercept messages in the middle of Alice and Bob, she can't forge signatures without Alice's private key and Bob's private key. Henceforth, the upgraded enhanced protocol defeats the man-in-the-middle attack."

8. Advantages and Disadvantages

8.1 Advantages

Its advantages are

- The security factors with respect to the fact that solving the discrete logarithm is very challenging, and
- That the shared key (i.e. the secret) is never itself transmitted over the channel.

8.2 Disadvantages

The algorithm has its share of drawbacks including

- The fact that there are expensive exponential operations involved, and the algorithm cannot be used to encrypt messages - it can be used for establishing a secret key only.
- There is also a lack of authentication.
- There is no identity of the parties involved in the exchange.
- It is easily susceptible to man-in-the-middle attacks. A third-party C, can exchange keys with both A and B, and can listen to the communication between A and B.
- The algorithm is computationally intensive. Each multiplication varies as the square of n, which must be very large. The number of multiplications required by the exponentiation increases with increasing values of the exponent, x or y in this case.
- The computational nature of the algorithm could be used in a denial- of-service attack very easily.

9. Conclusion

Designing a Key exchange algorithm with 100% Accuracy is not possible. Our Algorithm utilizes basic scientific ideas making execution simpler and in addition avoidance from common Attacks. Security change is useful in light of the fact that Diffie Hellman Algorithm is the premise of a few security standards and services on the internet, and if the security of the Diffie Hellman
algorithm is compromised, such frameworks will collapse. Diffie Hellman key trade approach for key distribution gives off an impression of being one of the favored systems utilized as a part of practice today. The Diffie-Hellman key exchange algorithm has turned out to be a stand-out amongst the most fascinating key distribution schemes being used today.

Nonetheless, one must know about the way that in spite of the Algorithm is safe against passive eavesdropping, it is not necessarily protected from active attacks. Diffie-Hellman algorithm should be complemented with an authentication mechanism. This methodology for key distribution gives off an impression of being one of the favored routines utilized as a part of practice today.