Diffie-Hellman Key Exchange

Stephane Moore

November 6, 2010

1] Background

Diffie-Hellman key exchange is one of the earliest practical key exchange algorithms implemented in the field of cryptography. It allows two parties with no prior knowledge of each other to jointly establish a shared secret key, which can then be used in symmetric key cryptographic algorithms, over an insecure channel. Diffie-Hellman key exchange relies on modular arithmetic and the differing computational complexity of discrete exponents and discrete logarithms.

2] Multiplication Modulo n and Discrete Exponentiation

Modular arithmetic is a system of arithmetic for integers where numbers are confined by a modulus, resulting in a wraparound when they reach the integer n. For arithmetic modulo n, the congruence class of an integer a can be denoted by the set $\overline{a}_n$ which contains all $\{a + cn | c \in \mathbb{Z}\}$. The set of congruence classes modulo n can be denoted and defined by $\mathbb{Z}_n = \{\overline{a}_n | a \in \mathbb{Z}\}$. Diffie-Hellman relies primarily on one notable property of modular multiplication:

$$\overline{a}_n \overline{b}_n = \overline{(ab)}_n$$

This property is useful for discrete exponentiation because this implies that $(\overline{a}_n)^k = \overline{(a^k)}_n$. This means that we can use traditional exponentiation algorithms, such as exponentiation by squaring, to efficiently compute exponents modulo n in $O(log_2 k)$ time or better.

3] Discrete Logarithm

Discrete logarithms are the group theory analogs of ordinary logarithms. Let us consider a generator b of the multiplicative group of integers modulo n, denoted by $\mathbb{Z}_n^\times$. Every element in $\mathbb{Z}_n$ can be expressed by exponentiation of b.

$$\forall a \in \mathbb{Z}_n (\exists k \in \mathbb{Z} \mid b^k \in \overline{a}_n)$$

In other words, for any $a \in \mathbb{Z}_n$, there is an integer k such that $a = b^k \pmod{n}$. Now consider a function $log_b : \mathbb{Z}_n \to \mathbb{Z}_n$ which, for a generator b of $\mathbb{Z}_n$ and an input $a \in \mathbb{Z}_n$, solves for x in the equation $b^x = a$. As of yet, there is no known efficient algorithm for computing general discrete logarithms. The naive algorithm, *trial multiplication*, yields results in $O(n)$ time. More efficient algorithms exist but none can compete with the efficiency of discrete exponentiation.

4] Diffie-Hellman Key Exchange

The discrepancy between the computational complexities of discrete exponents and discrete logarithms form the foundation for this key exchange algorithm. Consider the following simple example of the Diffie-Hellman protocol, used to establish a private shared key between two parties, Alice and Bob.

1. Alice and Bob agree on a prime p and a generator g of $\mathbb{Z}_p^\times$
2. Alice chooses a secret integer a and sends Bob $A = g^a \mod p$
3. Bob chooses a secret integer b and sends Alice $B = g^b \mod p$
4. Alice computes $s = B^a = (g^b)^a = g^{ba} \mod p$
5. Bob computes $s = A^b = (g^a)^b = g^{ab} \mod p$

At the end of the exchange, Alice and Bob have established a shared key because g^{ab} and g^{ba} are equal modulo p. This shared key is private because it is difficult for an eavesdropper to calculate $g^{ab} \mod p$ given p, g, $A = g^a \mod p$, and $B = g^b \mod p$. This is called the Diffie-Hellman problem. Alice and Bob can efficiently compute $g^{ab} \mod p$ through discrete exponentiation whereas an eavesdropper typically has to perform a costly discrete logarithm first. For properly chosen values of p and g, there is currently no known efficient way for an eavesdropper to calculate the shared key. This makes Diffie-Hellman Key Exchange a powerful tool in asymmetric cryptography.