

Diffie Hellman Key Exchange Algorithm Implementation

Definition: Diffie-Hellman key exchange, commonly known as polynomial key exchange, is a form of technology. Encoding that uses actual raised to particular values to generate a secret key based on never explicitly conveyed elements makes a would-be code breaker's work mathematically onerous.

The user will disclose the values of p , q , and public key A in the program following. On the other hand, the server will collect the values and compute its public key B before sending it to the client.

Both hosts will generate the secret key for symmetric encryption by utilizing the public key.

Clarification in further Steps:

Pardeep	Eqdeep
Public Keys available = P, G	Public Keys available = P, G
Private Key Selected = a	Private Key Selected = b
Key generated = $x = G^a \text{ mod } P$	Key generated = $y = G^b \text{ mod } P$
Exchange of generated keys takes place	
Key received = y	Key received = x
Generated Secret Key = $k_a = y^a \text{ mod } P$	Generated Secret Key = $k_b = x^b \text{ mod } P$

It is possible to demonstrate algebraically that

$$k_a = k_b$$

Users now have a symmetric secret key with which to encrypt their data.

I took Example and implementation with this example.

Example>

Step 1> Pardeep and Eqdeep get public numbers $P = 23, G = 9$

Step 2> Pardeep selected a private key $a = 4$ and

Eqdeep selected a private key $b = 3$

Step 3> Pardeep and Eqdeep calculate public values

Pardeep: $x = (9^4 \bmod 23) = (6561 \bmod 23) = 6$

Eqdeep: $y = (9^3 \bmod 23) = (729 \bmod 23) = 16$

Step 4> Pardeep and Eqdeep exchange public numbers

Step 5> Pardeep receives public key $y = 40$ and

Eqdeep receives public key $x = 42$

Step 6> Pardeep and Eqdeep compute symmetric keys

Pardeep: $ka = y^a \bmod p = 65536 \bmod 23 = 9$

Eqdeep: $kb = x^b \bmod p = 216 \bmod 23 = 9$

Step 7> 9 is the shared secret.

OUTPUT>

Screenshot of implementation:

```
// Both individuals will agree on the
// public keys G and P
P = 23;
G = 9;
cout << "The value of P : 23\n";
cout << "The value of G : 9\n";

// Pardeep's private key a
a = 4;
cout << "The private key a for Pardeep : 4\n";

// Eqdeep's private key b
b = 3;
cout << "The private key b for Eqdeep : 3\n";

// Calculating public values
x = pow(G, a, P);
y = pow(G, b, P);

// After exchanging public values
ka = pow(y, a, P);
kb = pow(x, b, P);
cout << "Secret key for the Pardeep is : 9\n";
cout << "Secret key for the Eqdeep is : 9\n";

return 0;
```

Process exited after 0.09297 seconds with return value 0
Press any key to continue . . .

Compilation results:
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Pardeep\Desktop\5th\5th Program-Deffie-2.exe
- Output Size: 1.8563060760498 MiB
- Compilation Time: 0.91s

Program code:

```
/* This program calculates the Key for two persons
using the Diffie-Hellman Key exchange algorithm using C++ */

#include <cmath>

#include <iostream>

using namespace std;

// Power function to return value of  $a^b \bmod P$ 
long long int power(long long int a, long long int b,
                    long long int P)
{
    if (b == 1)
        return a;

    else
        return (((long long int)pow(a, b)) % P);
}

// Start the program
int main()
{
    long long int P, G, x, a, y, b, ka, kb;

    // Both individuals will agree on the
    // public keys G and P
    P = 23; // A prime number P is taken
    cout << "The value of P : " << P << endl;

    G = 9; // A primitive root for P, G is taken
```

```

cout << "The value of G : " << G << endl;

// Pardeep will choose the private key a
a = 4; // a is the chosen private key
cout << "The private key a for Pardeep : " << a << endl;

x = power(G, a, P); // catch the generated key

// Eqdeep will choose the private key b
b = 3; // b is the chosen private key
cout << "The private key b for Eqdeep : " << b << endl;

y = power(G, b, P); // catch the generated key

// After the keys have been exchanged, the secret key is generated.
ka = power(y, a, P); // Secret key for Pardeep
kb = power(x, b, P); // Secret key for Eqdeep
cout << "Secret key for the Pardeep is : " << ka << endl;

cout << "Secret key for the Eqdeep is : " << kb << endl;

return 0;
}

// The End of code

```