Ex.No.5	**IMPLEMENTATION OF DIFFIE HELLMAN KEY EXCHANGE**

AIM:

To simulate the working of Diffie Hellman in Virtual lab environment and to implement the same in Client/ Server model using Java/Python

THEORY:

Steps Involved:

1. SELECTION OF PRIME NUMBERS:

Two communicating parties agree on:

• A large prime number q.

• A primitive root alpha of q.

2. PRIVATE KEYS:

Each party selects a private key:

• One party chooses a private key a.

• The other party chooses a private key b.

3.PUBILC KEYS:

Both parties compute public keys:

• The first party computes their public key as A=g a mod p and sends it.

• The second party computes their public key as B=g b mod p and sends it.

➢ These public keys are shared between the two parties.

4. Exchange and Computation of Shared Secret:

After exchanging public keys, each party calculates the shared secret:

- The first party computes S=B^a mod p.
- The second party S = A ^b mod p.

ALGORITHM:

Process 1:

- **Initializes a socket to listen for a connection.**
- **Sends the public values p and g.**
- **Generates a private key a and computes the public key A=g^ a mod p.**
- **Receives the client's public key B.**
- **Sends its public key A to the client.**
- **Computes the shared secret S = B^a mod p.**

Process 2:

- **Connects to the server.**
- **Receives the public values p and g.**
- **Generates a private key b and computes the public key B=g^ b mod p.**
- **Sends its public key B to the server.**
- **Receives the server's public key A.**
- **Computes the shared secret S= A^ b mod p.**

Screen Shots of simulation in Virtual labs

Coding

Server:

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.math.BigInteger;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Random;
```

```
public class UserA {
   public static void main(String[] args) throws IOException {
      ServerSocket ss = new ServerSocket(2729);
      System.out.println("Server (User A) is Running... Waiting For UserB to connect.");
      Socket s = ss.accept();
      System.out.println("Connected to UserB.");

      BufferedReader in = new BufferedReader(new
InputStreamReader(s.getInputStream()));
      PrintWriter out = new PrintWriter(s.getOutputStream(), true); // true for auto-flushing

      BigInteger q = new BigInteger("162259276829213363391578010288127");
      BigInteger alpha = new BigInteger("5");

      out.println(q);
      out.println(alpha);
      System.out.println("q:"+q);
      System.out.println("Alpha:"+alpha);

      BigInteger Xa;
      Random rand = new Random();
      do {
         Xa = new BigInteger(q.bitLength(), rand);
      } while (Xa.compareTo(q) >= 0 || Xa.equals(BigInteger.ZERO));

      System.out.println("UserA's private Key: " + Xa);
      BigInteger Ya = alpha.modPow(Xa, q);
      System.out.println("Computed UserA's Public Key: " + Ya);
      out.println(Ya);


      BigInteger Yb = new BigInteger(in.readLine());
      BigInteger K = Yb.modPow(Xa, q);
      System.out.println("UserA's Shared Secret Key: " + K);

      in.close();
      out.close();
```

```
        s.close();
        ss.close();
    }
}
```

Client:

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.math.BigInteger;
import java.net.Socket;
import java.util.Random;

public class UserB {
    public static void main(String[] args) throws IOException {
        Socket socket = new Socket("localhost", 2729);
        System.out.println("Connected To Server (UserA)...");

        BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
        PrintWriter out = new PrintWriter(socket.getOutputStream(), true); // true for auto-
flushing

        BigInteger q = new BigInteger(in.readLine());
        BigInteger alpha = new BigInteger(in.readLine());
        BigInteger Ya = new BigInteger(in.readLine());
        System.out.println("Received q:"+q);
        System.out.println("Received alpha:"+alpha);
        System.out.println("Received UserA's Public Key:"+Ya);
        Random rand = new Random();
        BigInteger Xb;
        do {
            Xb = new BigInteger(q.bitLength(), rand);
        } while (Xb.compareTo(q) >= 0 || Xb.equals(BigInteger.ZERO));

        System.out.println("UserB's Private Key: " + Xb);
```

```
        BigInteger Yb = alpha.modPow(Xb, q);
        System.out.println("Computed UserB's Public Key: " + Yb);
        out.println(Yb);
        BigInteger K = Ya.modPow(Xb, q);
        System.out.println("UserB's Shared Secret Key: " + K);

        in.close();
        out.close();
        socket.close();
    }
}
```

SCREEN SHOTS:

Server:

```
Server (User A) is Running... Waiting For UserB to connect.
Connected to UserB.
q:162259276829213363391578010288127
Alpha:5
UserA's private Key: 168870973464833862815001019724 31
Computed UserA's Public Key: 155650212649049507840171683325091
UserA's Shared Secret Key: 158918547440966098562889592525401
```

Client:

```
Connected To Server (UserA)...
Received q:162259276829213363391578010288127
Received alpha:5
Received UserA's Public Key:155650212649049507840171683325091
UserB's Private Key: 607942111852553008856528895681 65
Computed UserB's Public Key: 263973229709131305856983385524 0
UserB's Shared Secret Key: 158918547440966098562889592525401
```

RESULT:

Thus, simulated the working of Diffie Hellman in Virtual lab environment and implemented the same in Client/ Server model using Java.

Evaluation

Parameter	Max Marks	Marks Obtained
Uniqueness of the Code	40	
Completion of experiment on time	5	
Documentation	10	
Simulation in Vlabs	20	
Total	75	
Signature of the faculty with Date		