VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology and Engineering

Lab Assessment-V, OCTOBER 2020

B.Tech., Fall-2020-2021

NAME	PRIYAL BHARDWAJ
REG. NO.	18BIT0272
COURSE CODE	ITE3001
COURSE NAME	DATA COMMUNICATION & COMPUTER NETWORKS
SLOT	L15+L16
FACULTY	Prof. DINAKARAN MURUGANANDAM

1. Write a UDP based server code to get the date of birth of the client and calculate the age as on today. The client has to enter the year, month and day of birth. For example, if the date of birth of a user is 1/07/2001 then his age is 14 years 0 months and 17 days if today's date is 18/07/2018. Get today's date from the server.
    - Code for Date formatting
    - Creating UDP sockets for server and client
    - Code for validation of date format and computation of age for UDP transfer of segments.

UDP_Server.java

```
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.time.LocalDate;
import java.time.Period;
public class UDP_Server {
    private static String getAge(String[] dob) {
        LocalDate today = LocalDate.now();
        LocalDate bday = LocalDate.of(Integer.parseInt(dob[2]), Integer.parseI
nt(dob[1]), Integer.parseInt(dob[0]));
        Period p = Period.between(bday, today);
        return "Your Age : " + p.getYears() + " Years " + p.getMonths() + " Mo
nths " + p.getDays() + " Days.";
    }
    public static void main(String[] args) throws Exception {
        DatagramSocket ds = new DatagramSocket(5046);
        while (true) {
            byte[] b = new byte[1024];
            DatagramPacket packet = new DatagramPacket(b, b.length);
            ds.receive(packet);
            String data = new String(packet.getData());
            String[] dob = data.trim().split("-");
            String res = getAge(dob);
            byte[] b1 = res.getBytes();
            DatagramPacket packet1 = new DatagramPacket(b1, b1.length, packet.
getAddress(), packet.getPort());
            ds.send(packet1);
        }
    }
}
```

UDP_Client.java

```
import java.io.Console;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
public class UDP_Client {
    public static void main(String[] args) throws Exception {
        Console c = System.console();
```

```
        DatagramSocket ds = new DatagramSocket();
        while (true) {
            System.out.println("Enter date of birth like dd-MM-yyyy: ");
            String dob = c.readLine();
            byte[] b = dob.getBytes();
            InetAddress ia = InetAddress.getLocalHost();
            DatagramPacket packet = new DatagramPacket(b, b.length, ia, 5046);
            ds.send(packet);
            byte[] b1 = new byte[1024];
            DatagramPacket packet1 = new DatagramPacket(b1, b1.length);
            ds.receive(packet1);
            String res = new String(packet1.getData()).trim();
            System.out.println(res + "\n");
        }
    }
}
```

```
C:\Users\18BIT0272\Desktop\DCCN>javac UDP_Server.java

C:\Users\18BIT0272\Desktop\DCCN>java UDP_Server
```

```
C:\Users\18BIT0272\Desktop\DCCN>javac UDP_Client.java

C:\Users\18BIT0272\Desktop\DCCN>java UDP_Client
Enter date of birth like dd-MM-yyyy:
20-01-2000
Your Age : 20 Years 9 Months 9 Days.

Enter date of birth like dd-MM-yyyy:
27-07-1994
Your Age : 26 Years 3 Months 2 Days.
```

2. A reputed organization has two branches in Vellore. In one of the branch offices, a new manager has been appointed. The Senior Manager from the main office has to send the important records to the branch office. Implement a client-server model to accomplish this.
   - Understating of HTTPS for uploading of files to FTP server.
   - Encryption and decryption of the file's data with security algorithms.
   - Code for generation of a key for encryption and decryption using TCP/IP client server sockets.

   ➢ Https is a protocol used to transfer files from a Web server onto a browser in order to view a Web page that is on the Internet. FTP is a network protocol used to transfer files from one computer to another over a TCP network. In HTTPS there is no visibility of ones IP address. It is hidden under the website name for example: if we take vit website we do not know the IP address as it is hidden under the name and we can know all the complete details of the website in whois.net or to know the IP we can also use ping command. So, if we type “ping” we get the IP address as this. HTTPS is used to hide this IP. If the IP is known there can be a lot of things that can be done and there

will be no security to these organizations as we can get the complete URL information in whois.net or by using ping command.

Server.java

```
import java.io.BufferedInputStream;
import java.io.Console;
import java.io.DataInputStream;
import java.io.EOFException;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;
public class Server {
    private static final String secretkey = "123secretKey4567";
    private static final String serverFolderPath = "C:\\Users\18BIT0272\\Deskt
op\\DCCN\\";
    public static void main(String[] args) throws Exception {
        int bytesRead;
        int current = 0;
        ServerSocket ss = new ServerSocket(5046);
        FileOutputStream output = null;
        while (true) {
            Socket s = null;
            s = ss.accept();
            try {
                DataInputStream dis = new DataInputStream(s.getInputStream());
                String fileName = dis.readUTF();
                String ext = dis.readUTF();
                String fname = fileName.split("\\.")[0];
                File file = new File(serverFolderPath + "\\" + fname + "_" + n
ew Date().getTime() + ".encrpyted");
                File resFile = new File(serverFolderPath + "\\" + fname + "_"
+ new Date().getTime() + "." + ext);
                output = new FileOutputStream(file);
                long size = dis.readLong();
                byte[] b = new byte[1024];
                while (size > 0 && (bytesRead = dis.read(b, 0, (int) Math.min(
b.length, size))) != -1) {
                    output.write(b, 0, bytesRead);
                    size -= bytesRead;
                }
                Crypto.decrypt(secretkey, file, resFile);
                output.close();
                // file.delete();
```

```
            } catch (EOFException e) {
                continue;
            }
        }
    }
}
```

Client.java

```
import java.io.BufferedInputStream;
import java.io.Console;
import java.io.DataInputStream;
import java.io.DataOutput;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.OutputStream;
import java.net.Socket;
import java.util.Date;
public class Client {
    private static final String secretkey = "123secretKey4567";
    public static void main(String[] args) throws Exception {
        Socket s = new Socket("127.0.0.1", 5046);
        Console c = System.console();
        System.out.print("Enter The Absolute Path : (e.g: 'C:\\DCCN\\LA5\\exam
ple.txt') - >");
        String inputFile = c.readLine();
        try {
            File file = new File(inputFile);
            int idx = inputFile.lastIndexOf(".");
            String extension = null;
            if (idx >= 0)
                extension = inputFile.substring(idx + 1);
            String fname = file.getName().split("\\.")[0];
            File encryptedFile = new File("C:\\Users\18BIT0272\\Desktop\\DCCN\
\" + fname + ".encrypted");
            Crypto.encrypt(secretkey, file, encryptedFile);
            byte[] b = new byte[(int) encryptedFile.length()];
            FileInputStream fis = new FileInputStream(encryptedFile);
            BufferedInputStream bis = new BufferedInputStream(fis);
            DataInputStream dis = new DataInputStream(bis);
            dis.readFully(b, 0, b.length);
            DataOutputStream dos = new DataOutputStream(s.getOutputStream());
            dos.writeUTF(encryptedFile.getName());
            dos.writeUTF(extension);
            dos.writeLong(b.length);
            dos.write(b, 0, b.length);
            dos.flush();
            fis.close();
```

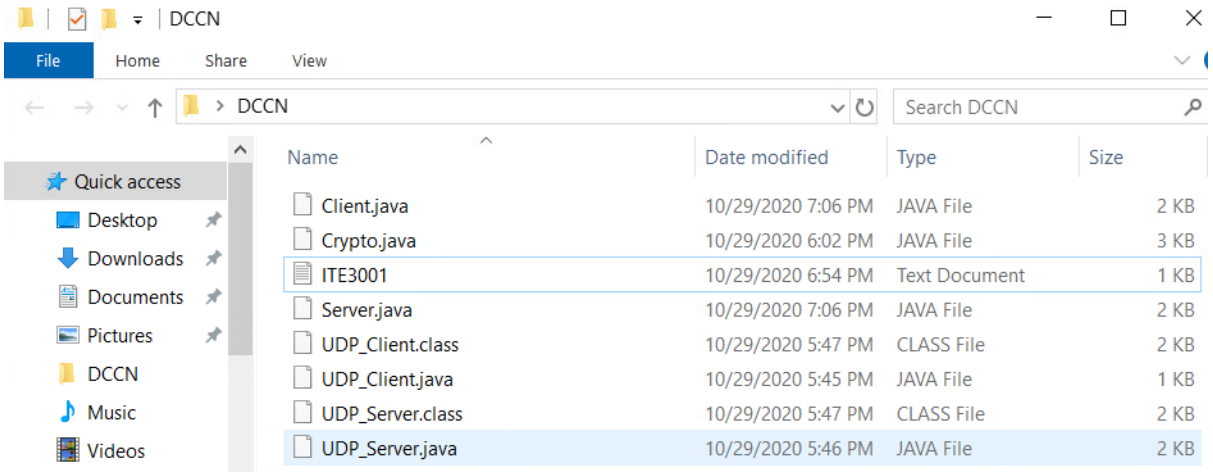
```
            encryptedFile.delete();
            System.out.println("File successfully uploaded on server.");
            s.close();
        } catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
    }
}
```

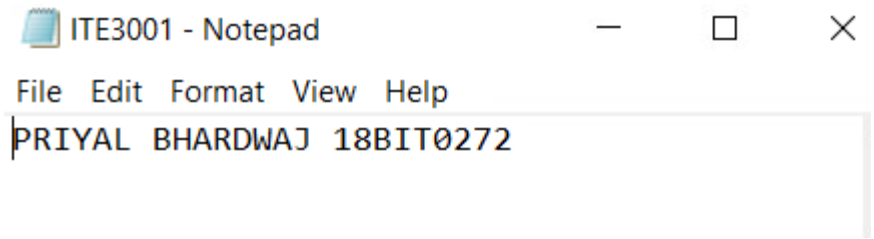
Crypto.java

```
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.security.InvalidKeyException;
import java.security.Key;
import java.security.NoSuchAlgorithmException;
import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.spec.SecretKeySpec;
public class Crypto {
    private static final String ALGORITHM = "AES";
    private static final String TRANSFORMATION = "AES";
    public static void encrypt(String key, File inputFile, File outputFile) th
rows CryptoException {
        doCrypto(Cipher.ENCRYPT_MODE, key, inputFile, outputFile);
    }
    public static void decrypt(String key, File inputFile, File outputFile) th
rows CryptoException {
        doCrypto(Cipher.DECRYPT_MODE, key, inputFile, outputFile);
    }
    private static void doCrypto(int cipherMode, String key, File inputFile, F
ile outputFile) throws CryptoException {
        try {
            Key secretKey = new SecretKeySpec(key.getBytes(), ALGORITHM);
            Cipher cipher = Cipher.getInstance(TRANSFORMATION);
            cipher.init(cipherMode, secretKey);
            FileInputStream inputStream = new FileInputStream(inputFile);
            byte[] inputBytes = new byte[(int) inputFile.length()];
            inputStream.read(inputBytes);
            byte[] outputBytes = cipher.doFinal(inputBytes);
            FileOutputStream outputStream = new FileOutputStream(outputFile);
            outputStream.write(outputBytes);
            inputStream.close();
            outputStream.close();
```

```
        } catch (NoSuchPaddingException | NoSuchAlgorithmException | InvalidKe
yException | BadPaddingException | IllegalBlockSizeException | IOException ex)
 {
            System.out.println(ex);
            throw new CryptoException("Error encrypting/decrypting file", ex);
        }
    }
}
class CryptoException extends Exception {
    public CryptoException() {
    }
    public CryptoException(String message, Throwable throwable) {
        super(message, throwable);
    }
}
```

File before encryption:

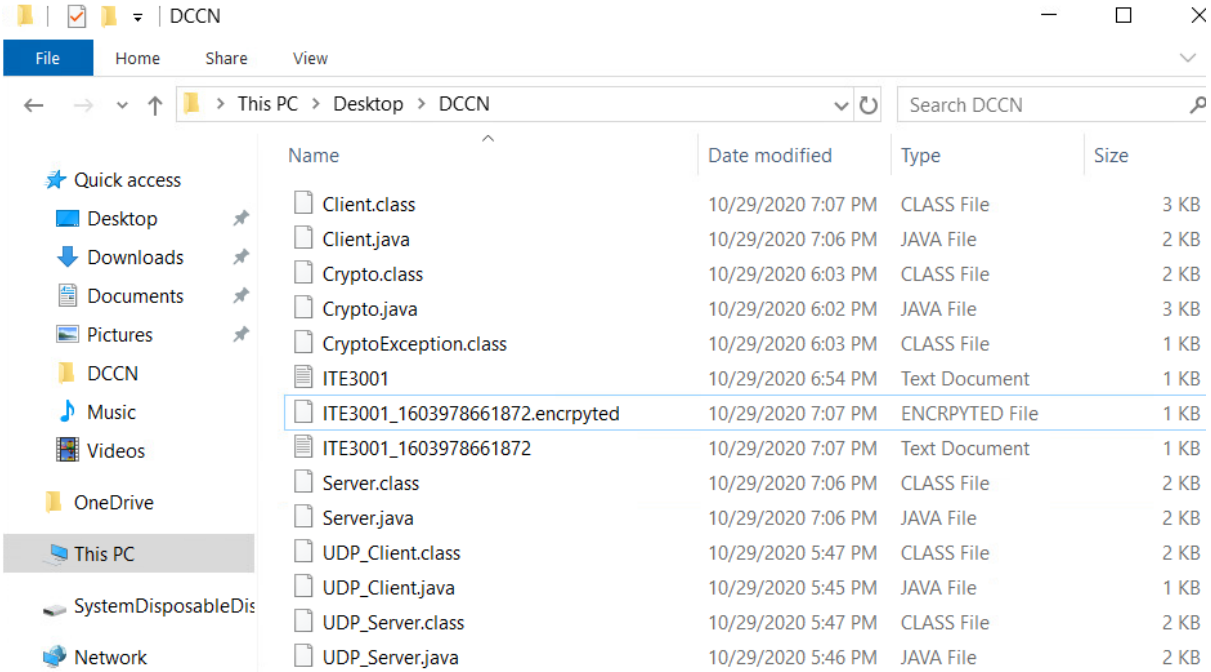


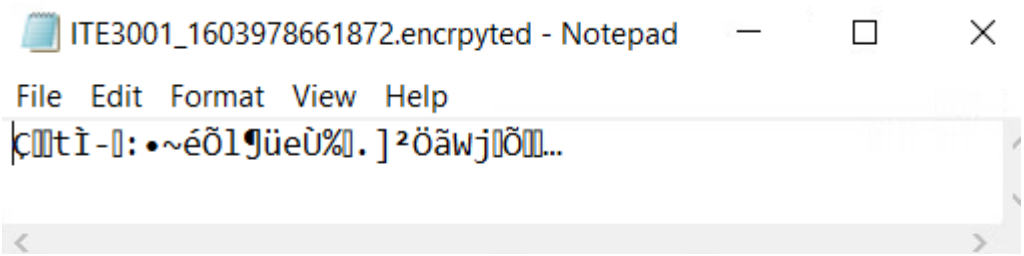



```
C:\Users\18BIT0272\Desktop\DCCN>javac Server.java

C:\Users\18BIT0272\Desktop\DCCN>java Server
```

```
C:\Users\18BIT0272\Desktop\DCCN>javac Client.java

C:\Users\18BIT0272\Desktop\DCCN>java Client
Enter The Absolute Path : (e.g: 'C:\DCCN\LA5\example.txt') : C:\\Users\\18BIT0272\\Desktop\\DCCN\\ITE3001.txt
File successfully uploaded on server.
```

File after encryption:





****