



TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHỆ

KHOA CÔNG NGHỆ THÔNG TIN

Môn: Bảo Mật Thông Tin

Bài Thực Hành Số 4



Bài1: Viết chương trình trao đổi dữ liệu theo mô hình Client-Server theo yêu cầu sau:

1.1 Sử dụng thuật toán Diffie-Hellman trao đổi quá cho nhau và sinh ra cặp khóa chung .

1.2 Dùng khóa chung mã hóa và giải mã văn bản (văn bản được lưu ở dạng .doc, .dat,...)

1.1 Hướng dẫn thuật toán trao đổi khóa Diffie-Hellman

Diffie-Hellman là một thuật toán dùng để trao đổi khóa chứ không dùng để bảo vệ tính bí mật của dữ liệu. Tuy nhiên, Diffie-Hellman lại có ích trong giai đoạn trao đổi khóa bí mật của các thuật toán mật mã đối xứng.

Thuật toán trao đổi khóa Diffie-Hellman dựa trên phép logarit rời rạc. Cho trước một số g và $x=g^k$, để tìm k ta thực hiện phép logarit : $k=\log_g(x)$. Tuy nhiên, nếu cho trước g , n và $(g^k \bmod n)$, thì quá trình xác định k được thực hiện theo cách khác với cách ở trên và được gọi là logarit rời rạc.

Thuật toán Diffie-Hellman được mô tả như sau:

- Gọi n là một số nguyên tố lớn và g là một cơ số sinh (generator, số nguyên nhỏ) thỏa điều kiện: với mọi $x \in \{1,2,\dots,n-1\}$, ta luôn tìm được số y sao cho $x=g^y \bmod n$.
- Giá trị n và g được phổ biến công khai giữa các thực thể trao đổi khóa. Sau đó user A tạo ra một số riêng $X_a < n$, tính giá trị $Y_a = (g^{X_a} \bmod n)$ và gửi cho B. Tương tự, user B cũng tạo ra một số riêng $X_b < n$ tính giá trị $Y_b = (g^{X_b} \bmod n)$ và

gửi lại cho A. X_a và X_b tương đương khóa private, Y_a và Y_b tương đương khóa public.

- User B xác định được khóa bí mật dùng cho phiên làm việc bằng cách tính giá trị $(g^{x_a} \bmod n)^{x_b} = (g^{x_a x_b} \bmod n)$. Bằng cách tương tự, user A cũng xác định được cùng khóa bí mật này bằng cách tính giá trị $(g^{x_b} \bmod n)^{x_a} = (g^{x_a x_b} \bmod n)$.
- Giả sử trong quá trình trao đổi các giá trị, phía tấn công bắt được $(g^{x_a} \bmod n)$ và $(g^{x_b} \bmod n)$, họ rất khó xác định được X_a và X_b vì độ phức tạp của phép toán logarit rời rạc là rất cao.
- Ví dụ: với $n=31$ và $g=3$

$$A: x=8 \Rightarrow y=3^8 \bmod 31 = 20 \quad \swarrow \quad k = 16^8 \bmod 31 = 4$$

$$B: x=6 \Rightarrow y=3^6 \bmod 31 = 16 \quad \searrow \quad k = 20^6 \bmod 31 = 4$$

Khóa bí mật không được tạo trước và chuyển từ A sang B hoặc ngược lại, khóa bí mật chỉ được tạo ra sau khi A và B trao đổi với nhau Y_a và Y_b . Vì vậy khóa bí mật này thường được gọi là khóa phiên.

1.2 Hướng dẫn thực hành

1.2.1 Tạo Class CryptoUtil viết phương thức sau:

```
public static final String toHexString(byte[] block)
{
    StringBuffer buf = new StringBuffer();
    int len = block.length;

    for (int i = 0; i < len; i++)
    {
        byte2hex(block[i], buf);
        if (i < len-1)
        {
            buf.append(":");
        }
    }
    return buf.toString();
}
```

Hàm này dùng để chuyển từ kiểu hex sang kiểu String.

```

public static final void byte2hex(byte b, StringBuffer buf)
{
    char[] hexChars = { '0', '1', '2', '3',
                          '4', '5', '6', '7',
                          '8', '9', 'A', 'B',
                          'C', 'D', 'E', 'F' };

    int high = ((b & 0xf0) >> 4);
    int low = (b & 0x0f);
    buf.append(hexChars[high]);
    buf.append(hexChars[low]);
}

```

1.2.2 Bên Alice

1.2.2.1 Thiết kế form Alice

1.2.2.1 Viết hàm xử lý sự kiện Tạo Khóa A

Các biến cần khai báo bên Alice

```

KeyAgreement aliceKeyAgree;
PublicKey bobPubKey;
SecretKey aliceDesKey;
Cipher aliceCipher;

```

```

private void bntkhoaAActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try{
        AlgorithmParameterGenerator paramGen = AlgorithmParameterGenerator.getInstance("DH");
        paramGen.init(512);
        AlgorithmParameters params = paramGen.generateParameters();

        DHParameterSpec dhSkipParamSpec =
            (DHParameterSpec) params.getParameterSpec(DHParameterSpec.class);

        System.out.println("Generating a DH KeyPair...");
        KeyPairGenerator aliceKpairGen = KeyPairGenerator.getInstance("DH");
        aliceKpairGen.initialize(dhSkipParamSpec);
        KeyPair aliceKpair = aliceKpairGen.generateKeyPair();

        System.out.println("Initializing the KeyAgreement Engine with DH private key");
        aliceKeyAgree = KeyAgreement.getInstance("DH");
        aliceKeyAgree.init(aliceKpair.getPrivate());

        byte[] alicePubKeyEnc = aliceKpair.getPublic().getEncoded();
        FileOutputStream fos=new FileOutputStream("D:/A.pub");
        fos.write(alicePubKeyEnc);
        fos.close();
        txtkhoaA.setText(alicePubKeyEnc.toString());

    }catch(Exception ex){}
}

```

1.2.2.2 Viết hàm xử lý sự kiện Hiện Thị KB

```

private void bntkhoaBActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try{
        FileInputStream fis=new FileInputStream("D:/B.pub");
        byte[] bkeyP=new byte[fis.available()];
        fis.read(bkeyP);
        fis.close();
        txtkhoaB.setText(bkeyP.toString());
    }
    catch(Exception ex){
    }
}

```

1.2.2.3 Viết hàm xử lý sự kiện Khóa Chung

```

private void bntkhoaABActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try{
        FileInputStream fis=new FileInputStream("D:/B.pub");
        byte[] bobPubKeyEnc=new byte[fis.available()];
        fis.read(bobPubKeyEnc);
        fis.close();

        KeyFactory aliceKeyFac = KeyFactory.getInstance("DH");
        X509EncodedKeySpec x509KeySpec = new X509EncodedKeySpec(bobPubKeyEnc);
        bobPubKey = aliceKeyFac.generatePublic(x509KeySpec);
        System.out.println("Executing PHASE1 of key agreement...");
        aliceKeyAgree.doPhase(bobPubKey, true);
        byte[] aliceSharedSecret = aliceKeyAgree.generateSecret();

        System.out.println("khoa chung: secret (DEBUG ONLY):" + CryptoUtil.toHexString(aliceSharedSecret));
        txtkhoaachung.setText(CryptoUtil.toHexString(aliceSharedSecret));

    }
    catch(Exception ex){}
}

```

1.2.2.4 Viết hàm xử lý sự kiện Mã Hóa KAB

```

private void bntmahoakabActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    //ma hoa khoa chung bang thuat toan DES
    try{
        aliceKeyAgree.doPhase(bobPubKey, true);
        aliceDesKey = aliceKeyAgree.generateSecret("DES");
        txtmahoakab.setText(aliceDesKey.toString());
        // khoa chung A-B
        BufferedWriter bw = null;
        //ghi van ban da ma hoa
        String fileName = "D:\\KhoaA.txt";
        //luu van ban
        bw = new // van ban sau khi ma hoa
        BufferedWriter(new FileWriter(fileName));
        // ghi van ban
        bw.write(aliceDesKey.toString());
        bw.close();

    }catch(Exception ex){}
}

```

1.2.2.5 Viết hàm xử lý sự kiện Mã Hóa/ Giải mã

```

private void bntmahoagiamaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    DESCS des= new DESCS();
    des.setVisible(true);
}

```

1.2.3 Bên Bob

1.2.3.1 Thiết kế form Bob

Bob

Khóa BOB :

Khóa Alice :

Khóa KAB:

Mã Hóa KAB:

Các biến cần khai báo cho bob

```
KeyAgreement bobKeyAgree;  
PublicKey alicePubKey ;  
SecretKey bobDesKey;  
Cipher bobCipher;
```

1.2.3.2 Viết hàm xử lý sự kiện Hiển Thị Khóa KA(Alice)

```
private void bntkhoaAActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    try{  
        FileInputStream fis=new FileInputStream("D:/A.pub");  
        byte[] akeyP=new byte[fis.available()];  
        fis.read(akeyP);  
        fis.close();  
        txtkhoaA.setText(akeyP.toString());  
    }  
    catch(Exception ex){  
    }  
}
```

1.2.3.3 Viết hàm xử lý sự kiện Tạo khóa KB (bob)

```
private void bntkhoaBActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    try{  
        boolean read=false;  
        //doi cho toi khi co tap tin alice.pub  
        while(!read){  
            try{  
                FileInputStream fis=new FileInputStream("D:/A.pub");  
                fis.close();  
                read=true;  
            }catch(Exception ex){}  
        }  
    }  
}
```



```

        FileInputStream fis=new FileInputStream("D:/A.pub");
        byte[] alicePubKeyEnc=new byte[fis.available()];
        fis.read(alicePubKeyEnc);
        fis.close();
        KeyFactory bobKeyFac = KeyFactory.getInstance("DH");
        X509EncodedKeySpec x509KeySpec = new X509EncodedKeySpec(alicePubKeyEnc);
        alicePubKey = bobKeyFac.generatePublic(x509KeySpec);
        DHParameterSpec dhParamSpec = ((DHPublicKey) alicePubKey).getParams();
        System.out.println("Generate DH keypair ...");
        KeyPairGenerator bobKpairGen = KeyPairGenerator.getInstance("DH");
        bobKpairGen.initialize(dhParamSpec);
        KeyPair bobKpair = bobKpairGen.generateKeyPair();
        System.out.println("Initializing KeyAgreement engine...");
        bobKeyAgree = KeyAgreement.getInstance("DH");
        bobKeyAgree.init(bobKpair.getPrivate());
        byte[] bobPubKeyEnc = bobKpair.getPublic().getEncoded();
        FileOutputStream fos=new FileOutputStream("D:/B.pub");
        fos.write(bobPubKeyEnc);
        fos.close();
        txtkhoab.setText(bobPubKeyEnc.toString());
    }catch(Exception ex){}
}

```

1.2.3.4 Viết hàm xử lý sự kiện Khóa Chung và sự kiện Mã Hóa KAB

```

private void bntkhoaABActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try{
        bobKeyAgree.doPhase(alicePubKey, true);
        byte[] bobSharedSecret = bobKeyAgree.generateSecret();
        System.out.println("Khoa chung :Shared secret (DEBUG ONLY): " + CryptoUtil.toHexString(bobSharedSecret));
        txtkhoachung.setText(CryptoUtil.toHexString(bobSharedSecret));
    }
    catch(Exception ex){}
}

private void bntmahoakabActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try{
        bobKeyAgree.doPhase(alicePubKey, true);
        bobDesKey = bobKeyAgree.generateSecret("DES");
        txtmahoakab.setText(bobDesKey.toString());
        // khoa chung A-B
        BufferedWriter bw = null;
        //ghi van ban da ma hoa
        String fileName = "D:\\KhoaB.txt";
        //luu van ban
        bw = new BufferedWriter(new FileWriter(fileName));
        // ghi van ban
        bw.write(bobDesKey.toString());
        bw.close();
    }catch(Exception ex){}
}

```


1.2.3.5 Viết hàm xử lý sự kiện Mã Hóa/ Giải mã

```
private void bntmahoaActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    DESCs des= new DESCs();  
    des.setVisible(true);  
}
```

1.2.4 Viết Frame DESCs

1.2.4.1 Thiết kế form sao

The image shows a Java Swing window titled 'DESCs'. It contains the following elements:

- Input Key :** A single-line text input field.
- Buttons:** Four buttons arranged horizontally: 'Mã Hóa', 'Mở Khóa A', 'Mở Khóa B', and 'Ghi File'.
- Plaintext :** A multi-line text area for entering the plaintext.
- CipherText :** A multi-line text area for displaying the ciphertext.
- Bottom Buttons:** Two buttons: 'Giải Mã' and 'All Show'.

1.2.4.2 Viết chức năng Mã Hóa

```
private void bntMaHoaActionPerformed(java.awt.event.ActionEvent evt) {  
    try {  
        String key=txtkhoa.getText() ; // needs to be at least 8 characters for DES  
  
        FileInputStream fis = new FileInputStream("D:\\Des.txt");  
        FileOutputStream fos = new FileOutputStream("D:\\EnDes.txt");  
        encrypt(key, fis, fos);  
        JOptionPane.showMessageDialog(null, " Đã mã hóa văn bản");  
        //FileInputStream fis2 = new FileInputStream("D:\\encrypted.txt");  
        //FileOutputStream fos2 = new FileOutputStream("D:\\decrypted.txt");  
        //decrypt(key, fis2, fos2);  
    } catch (Throwable e) {  
        e.printStackTrace();  
    }  
}
```

1.2.4.3 Viết chức năng mở khóa A

```
private void bntMoKhoaAActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    try {  
        BufferedReader br = null;  
  
        String fileName = "D:\\\\KhoaA.txt"; //GEN-  
        br = new BufferedReader(new FileReader(fileName));  
        StringBuffer sb = new StringBuffer();  
  
        JOptionPane.showMessageDialog(null, " Đã mở file");  
        char[] ca = new char[5];  
        while (br.ready()) {  
            int len = br.read(ca);  
            sb.append(ca, 0, len);  
        }  
        br.close();  
        //xuat chuoi  
        System.out.println("Du Lieu la :" + " " + sb);  
        String chuoi = sb.toString();  
  
        txtkhoa.setText(chuoi);  
    } catch (IOException ex) {  
        Logger.getLogger(DESCS.class.getName()).log(Level.SEVERE, null, ex);  
    }  
}
```

1.2.4.4 Viết chức năng mở khóa B

```
private void bntMoKhoaBActionPerformed(java.awt.event.ActionEvent evt) {  
    try {  
        BufferedReader br = null;  
  
        String fileName = "D:\\\\KhoaB.txt"; //GEN-  
        br = new BufferedReader(new FileReader(fileName));  
        StringBuffer sb = new StringBuffer();  
  
        JOptionPane.showMessageDialog(null, " Đã mở file");  
        char[] ca = new char[5];  
        while (br.ready()) {  
            int len = br.read(ca);  
            sb.append(ca, 0, len);  
        }  
        br.close();  
        //xuat chuoi  
        System.out.println("Du Lieu la :" + " " + sb);  
        String chuoi = sb.toString();  
  
        txtkhoa.setText(chuoi);  
    } catch (IOException ex) {  
        Logger.getLogger(DESCS.class.getName()).log(Level.SEVERE, null, ex);  
    }  
}
```

1.2.4.5 Viết chức năng ghi File

```

private void bntGhiFileActionPerformed(java.awt.event.ActionEvent evt) {
    try {

        BufferedWriter bw = null;
        //ghi van ban da ma hoa
        String fileName = "D:\\Des.txt";
        //luu van ban
        String s = txtvanban.getText();
        bw = new // van ban sau khi ma hoa
        BufferedWriter(new FileWriter(fileName));
        // ghi van ban
        bw.write(s);
        bw.close();
        JOptionPane.showMessageDialog(null, " Đã ghi file");
        txtmahoa.setText(s);
    } catch (IOException ex) {
        Logger.getLogger(DESCS.class.getName()).log(Level.SEVERE, null, ex);
    }

}

```

1.2.4.6 Viết chức năng giải mã

```

private void bntGiaiMaActionPerformed(java.awt.event.ActionEvent evt) {
    FileInputStream fis2 = null;
    try {
        String key = txtkhoa.getText();
        fis2 = new FileInputStream("D:\\EnDes.txt");
        FileOutputStream fos2 = new FileOutputStream("D:\\DeDes.txt");
        decrypt(key, fis2, fos2);
        BufferedReader br = null;
        String fileName = "D:\\DeDes.txt"; //GEN-
        br = new BufferedReader(new FileReader(fileName));
        StringBuffer sb = new StringBuffer();
        JOptionPane.showMessageDialog(null, " Đã Giải Mã");
        char[] ca = new char[5];
        while (br.ready()) {
            int len = br.read(ca);
            sb.append(ca, 0, len);
        }
        br.close();
        //xuat chuoi
        System.out.println("Du Lieu la : " + " " + sb);
        String chuoi = sb.toString();
        txtmahoa.setText(chuoi);
    } catch (Throwable ex) {
    }

}

```

1.2.4.7 Viết chức năng hiển thị

```
private void bnthienthitatcaActionPerformed(java.awt.event.ActionEvent evt) {  
    try {  
        BufferedReader br = null;  
        String fileName = "D:\\DeDes.txt"; //GEN-  
        br = new BufferedReader(new FileReader(fileName));  
        StringBuffer sb = new StringBuffer();  
        JOptionPane.showMessageDialog(null, " Đã mở file");  
        char[] ca = new char[5];  
        while (br.ready()) {  
            int len = br.read(ca);  
            sb.append(ca, 0, len);  
        }  
        br.close();  
        String ff = "D:\\EnDes.txt";  
        br = new BufferedReader(new FileReader(ff));  
        StringBuffer sb1 = new StringBuffer();  
        char[] ca1 = new char[5];  
        while (br.ready()) {  
            int len = br.read(ca1);  
            sb1.append(ca1, 0, len);  
        }  
        //xuat chuoai  
        System.out.println("Du Lieu la : " + " " + sb);  
        String chuoai = sb.toString();  
        String chuoil = sb1.toString();  
        txtvanban.setText(chuoai);  
        txtmahoa.setText(chuoil);  
    } catch (IOException ex) {  
    }  
}
```

1.2.5 Kết Quả

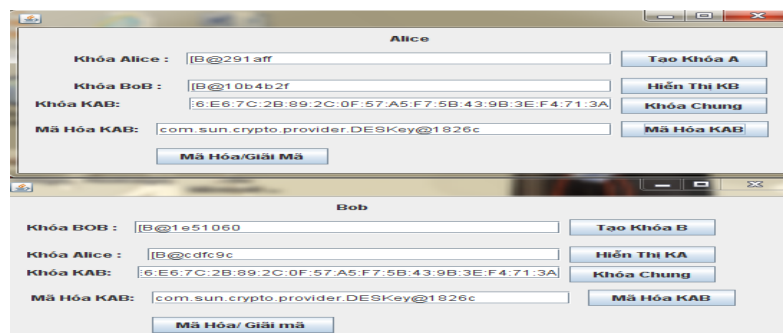
B1: Run Frame Alice: check button tạo khóa A.

B2 : Run Frame Bob: check button tạo khóa B.

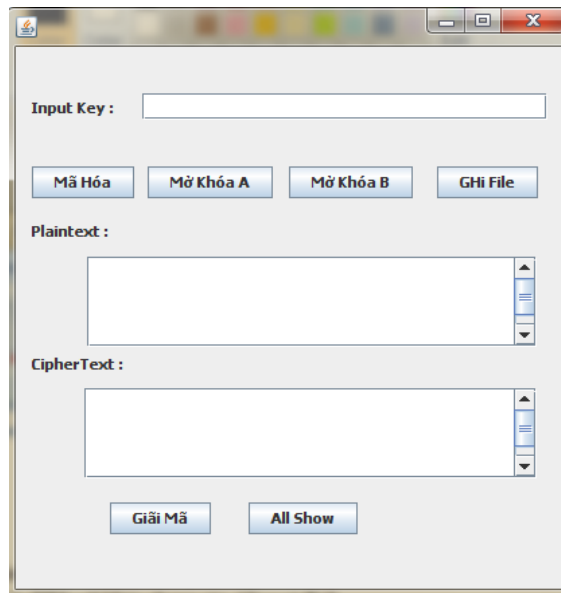
B3: Frame Alice: Check button hiển thị KB; Frame Bob: Check button hiển thị KA.

B4: Frame Alice và Frame Bob lần lượt check button tạo khóa Chung.

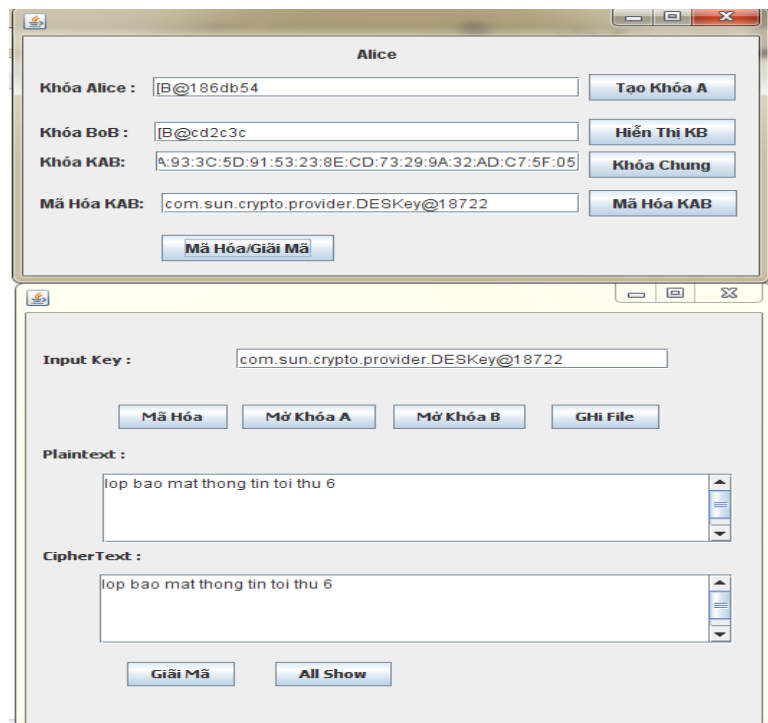
B5: Frame Alice và Frame Bob lần lượt check button mã hóa KAB



B5. Frame Aice: check button mã hóa và giải mã , button cho phép hiển thị form mã hóa và giải mã văn bản với thuật toán DES với khóa chung của Alice và Bob.



B6: Nhập nội dung văn bản cần mã hóa và ghi xuống File, trước khi nhập cần check button Mở Khóa A (khóa Alice) để dùng khóa Alice mã hóa văn bản.



B7: Frame Bob check button mã hóa và giải mã , button cho phép hiển thị form mã hóa và giải mã văn bản với thuật toán DES với khóa chung của Alice và Bob. Check button Mở Khóa B , dung khóa của Bob để giải mã văn bản do Alice mã hóa.

Bob

Khóa BOB :

Khóa Alice :

Khóa KAB:

Mã Hóa KAB:

Input Key :

Plaintext :

CipherText :

TC DaiViet 17/09/2011 7:37 AM