sudo apt update

sudo apt install default-jdk

Mã hóa chuỗi bất kì từ bàn phím

**mhChuoi.java**

import java.io.\*;

import java.security.\*;

import javax.crypto.Cipher;

import java.util.Base64;

import java.util.Scanner;

public class mhChuoi {

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

Scanner sc = new Scanner(System.in);

System.out.print("Nhap chuoi: ");

String message = sc.nextLine();

// Doc public key

byte[] publicKeyBytes = new FileInputStream("publicKey").readAllBytes();

KeyFactory keyFactory = KeyFactory.getInstance("RSA");

PublicKey publicKey = keyFactory.generatePublic(new java.security.spec.X509EncodedKeySpec(publicKeyBytes));

// Ma hoa

Cipher cipher = Cipher.getInstance("RSA");

cipher.init(Cipher.ENCRYPT\_MODE, publicKey);

byte[] encryptedBytes = cipher.doFinal(message.getBytes());

String encryptedBase64 = Base64.getEncoder().encodeToString(encryptedBytes);

// Hoi ten file luu chuoi ma hoa

System.out.print("Nhap ten file luu chuoi ma hoa: ");

String outFile = sc.nextLine();

if (outFile.isEmpty()) outFile = "mahoa.txt";

// Lưu vào file

try (BufferedWriter writer = new BufferedWriter(new FileWriter(outFile))) {

writer.write(encryptedBase64);

}

System.out.println("Chuoi da duoc ma hoa va luu vao: " + outFile);

sc.close();

}

}

**dcChuoi.java**

import java.io.\*;

import java.security.\*;

import javax.crypto.Cipher;

import java.util.Base64;

import java.util.Scanner;

public class dcChuoi {

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

Scanner sc = new Scanner(System.in);

System.out.print("Nhap ten file chua chuoi giai ma: ");

String inputFile = sc.nextLine();

if (inputFile.isEmpty()) inputFile = "mahoachuoi.txt";

String encryptedBase64;

try (BufferedReader reader = new BufferedReader(new FileReader(inputFile))) {

encryptedBase64 = reader.readLine();

}

byte[] privateKeyBytes = new FileInputStream("privateKey").readAllBytes();

KeyFactory keyFactory = KeyFactory.getInstance("RSA");

PrivateKey privateKey = keyFactory.generatePrivate(new java.security.spec.PKCS8EncodedKeySpec(privateKeyBytes));

Cipher cipher = Cipher.getInstance("RSA");

cipher.init(Cipher.DECRYPT\_MODE, privateKey);

byte[] decryptedBytes = cipher.doFinal(Base64.getDecoder().decode(encryptedBase64));

String decryptedMessage = new String(decryptedBytes);

System.out.println("Chuoi sau khi giai ma:");

System.out.println(decryptedMessage);

sc.close();

}

}

Ma hoa tap tin TXT

**mhTapTinTXT.java**

import java.io.\*;

import java.security.\*;

import javax.crypto.Cipher;

import java.util.Base64;

import java.util.Scanner;

public class mhTapTinTXT {

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

Scanner sc = new Scanner(System.in);

// Hoi ten file in

System.out.print("Nhap ten file muon ma hoa: ");

String inputFileName = sc.nextLine();

// Hoi ten file out

System.out.print("Nhap ten file de luu chuoi ma hoa: ");

String outputFileName = sc.nextLine();

// Doc noi dung file

StringBuilder sb = new StringBuilder();

try (BufferedReader reader = new BufferedReader(new FileReader(inputFileName))) {

String line;

while ((line = reader.readLine()) != null) {

sb.append(line).append("\n"); // giu xuong dong

}

}

String message = sb.toString().trim();

// Doc public key

byte[] publicKeyBytes = new FileInputStream("publicKey").readAllBytes();

KeyFactory keyFactory = KeyFactory.getInstance("RSA");

PublicKey publicKey = keyFactory.generatePublic(

new java.security.spec.X509EncodedKeySpec(publicKeyBytes)

);

// Ma hoa

Cipher cipher = Cipher.getInstance("RSA");

cipher.init(Cipher.ENCRYPT\_MODE, publicKey);

byte[] encryptedBytes = cipher.doFinal(message.getBytes());

// Luu chuoi ma hoa vao file do nguoi dung nhap

String encryptedBase64 = Base64.getEncoder().encodeToString(encryptedBytes);

try (BufferedWriter writer = new BufferedWriter(new FileWriter(outputFileName))) {

writer.write(encryptedBase64);

}

System.out.println("Da ma hoa noi dung file " + inputFileName + " va luu vao " + outputFileName);

sc.close();

}

}

**dcTapTinTXT.java**

import java.io.\*;

import java.security.\*;

import javax.crypto.Cipher;

import java.util.Base64;

import java.util.Scanner;

public class dcTapTinTXT {

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

Scanner sc = new Scanner(System.in);

// Hỏi tên file chứa chuỗi mã hóa

System.out.print("Nhap ten file chua chuoi ma hoa: ");

String inputFileName = sc.nextLine();

// Hỏi tên file để lưu chuỗi đã giải mã

System.out.print("Nhap ten file de luu chuoi da giai ma: ");

String outputFileName = sc.nextLine();

// Doc file mã hóa

String encryptedBase64;

try (BufferedReader reader = new BufferedReader(new FileReader(inputFileName))) {

encryptedBase64 = reader.readLine();

}

// Doc private key

byte[] privateKeyBytes = new FileInputStream("privateKey").readAllBytes();

KeyFactory keyFactory = KeyFactory.getInstance("RSA");

PrivateKey privateKey = keyFactory.generatePrivate(

new java.security.spec.PKCS8EncodedKeySpec(privateKeyBytes)

);

// Giai ma

Cipher cipher = Cipher.getInstance("RSA");

cipher.init(Cipher.DECRYPT\_MODE, privateKey);

byte[] decryptedBytes = cipher.doFinal(Base64.getDecoder().decode(encryptedBase64));

String decryptedMessage = new String(decryptedBytes);

// Luu ket qua vao file do nguoi dung nhap

try (BufferedWriter writer = new BufferedWriter(new FileWriter(outputFileName))) {

writer.write(decryptedMessage);

}

System.out.println("Noi dung da duoc giai ma va luu vao " + outputFileName);

sc.close();

}

}

Ma hoa file .doc

**mhTapTinDoc.java**

import java.io.\*;

import java.security.\*;

import javax.crypto.\*;

import javax.crypto.spec.SecretKeySpec;

import java.util.Base64;

import java.util.Scanner;

public class mhTapTinDoc {

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

Scanner sc = new Scanner(System.in);

System.out.print("Nhap ten file .doc muon ma hoa: ");

String inputFileName = sc.nextLine();

System.out.print("Nhap ten file de luu chuoi ma hoa: ");

String outputFileName = sc.nextLine();

// Đọc file gốc

byte[] fileBytes = new FileInputStream(inputFileName).readAllBytes();

// Tạo khóa AES ngẫu nhiên

KeyGenerator keyGen = KeyGenerator.getInstance("AES");

keyGen.init(128);

SecretKey aesKey = keyGen.generateKey();

// Mã hóa dữ liệu bằng AES

Cipher aesCipher = Cipher.getInstance("AES");

aesCipher.init(Cipher.ENCRYPT\_MODE, aesKey);

byte[] encryptedFile = aesCipher.doFinal(fileBytes);

// Đọc public key RSA

byte[] publicKeyBytes = new FileInputStream("publicKey").readAllBytes();

KeyFactory keyFactory = KeyFactory.getInstance("RSA");

PublicKey publicKey = keyFactory.generatePublic(new java.security.spec.X509EncodedKeySpec(publicKeyBytes));

// Mã hóa khóa AES bằng RSA

Cipher rsaCipher = Cipher.getInstance("RSA");

rsaCipher.init(Cipher.ENCRYPT\_MODE, publicKey);

byte[] encryptedAesKey = rsaCipher.doFinal(aesKey.getEncoded());

// Lưu cả khóa AES đã mã hóa + dữ liệu đã mã hóa

try (FileOutputStream fos = new FileOutputStream(outputFileName)) {

fos.write(Base64.getEncoder().encode(encryptedAesKey));

fos.write("\n".getBytes());

fos.write(Base64.getEncoder().encode(encryptedFile));

}

System.out.println("Đã mã hóa file " + inputFileName + " và lưu vào " + outputFileName);

sc.close();

}

}

**dcTapTinDoc.java**

import java.io.\*;

import java.security.\*;

import javax.crypto.\*;

import javax.crypto.spec.SecretKeySpec;

import java.util.Base64;

import java.util.Scanner;

public class dcTapTinDoc {

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

Scanner sc = new Scanner(System.in);

// Hỏi tên file mã hóa

System.out.print("Nhap ten file chua chuoi ma hoa: ");

String inputFileName = sc.nextLine();

// Hỏi tên file đầu ra

System.out.print("Nhap ten file de luu ket qua giai ma (.doc): ");

String outputFileName = sc.nextLine();

// Đọc toàn bộ file mã hóa

BufferedReader reader = new BufferedReader(new FileReader(inputFileName));

String encryptedAesKeyBase64 = reader.readLine(); // dòng đầu là AES key đã mã hóa

StringBuilder sb = new StringBuilder();

String line;

while ((line = reader.readLine()) != null) {

sb.append(line);

}

reader.close();

String encryptedFileBase64 = sb.toString();

// Giải mã khóa AES bằng private key RSA

byte[] privateKeyBytes = new FileInputStream("privateKey").readAllBytes();

KeyFactory keyFactory = KeyFactory.getInstance("RSA");

PrivateKey privateKey = keyFactory.generatePrivate(

new java.security.spec.PKCS8EncodedKeySpec(privateKeyBytes)

);

Cipher rsaCipher = Cipher.getInstance("RSA");

rsaCipher.init(Cipher.DECRYPT\_MODE, privateKey);

byte[] aesKeyBytes = rsaCipher.doFinal(Base64.getDecoder().decode(encryptedAesKeyBase64));

SecretKey aesKey = new SecretKeySpec(aesKeyBytes, "AES");

// Giải mã nội dung file bằng AES

Cipher aesCipher = Cipher.getInstance("AES");

aesCipher.init(Cipher.DECRYPT\_MODE, aesKey);

byte[] decryptedBytes = aesCipher.doFinal(Base64.getDecoder().decode(encryptedFileBase64));

// Ghi kết quả ra file

try (FileOutputStream fos = new FileOutputStream(outputFileName)) {

fos.write(decryptedBytes);

}

System.out.println("Đã giải mã file " + inputFileName + " và lưu vào " + outputFileName);

sc.close();

}

}

**CreateKey.java**

import java.io.FileOutputStream;

import java.security.KeyPair;

import java.security.KeyPairGenerator;

import java.security.PrivateKey;

import java.security.PublicKey;

public class CreateKey {

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

// Tạo bộ sinh khóa

KeyPairGenerator keyGen = KeyPairGenerator.getInstance("RSA");

keyGen.initialize(2048); // Độ dài khóa

// Sinh cặp khóa

KeyPair pair = keyGen.generateKeyPair();

PublicKey publicKey = pair.getPublic();

PrivateKey privateKey = pair.getPrivate();

// Lưu public key dạng X.509 (DER)

try (FileOutputStream fos = new FileOutputStream("publicKey")) {

fos.write(publicKey.getEncoded());

}

// Lưu private key dạng PKCS8 (DER)

try (FileOutputStream fos = new FileOutputStream("privateKey")) {

fos.write(privateKey.getEncoded());

}

System.out.println("Đã tạo xong publicKey và privateKey!");

}

}

import java.io.\*;

import java.security.\*;

import javax.crypto.\*;

import javax.crypto.spec.SecretKeySpec;

import java.util.Base64;

import java.util.Scanner;

public class dcThuMuc {

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

Scanner sc = new Scanner(System.in);

System.out.print("Nhap duong dan thu muc da ma hoa: ");

String inputFolderPath = sc.nextLine();

System.out.print("Nhap duong dan thu muc de luu file giai ma: ");

String outputFolderPath = sc.nextLine();

File inputFolder = new File(inputFolderPath);

if (!inputFolder.exists() || !inputFolder.isDirectory()) {

System.out.println("Thu muc khong hop le!");

return;

}

byte[] privateKeyBytes = new FileInputStream("privateKey").readAllBytes();

KeyFactory keyFactory = KeyFactory.getInstance("RSA");

PrivateKey privateKey = keyFactory.generatePrivate(

new java.security.spec.PKCS8EncodedKeySpec(privateKeyBytes)

);

String encryptedAesKeyBase64 = new String(new FileInputStream(inputFolderPath + "/encrypted\_aes\_key.txt").readAllBytes());

Cipher rsaCipher = Cipher.getInstance("RSA");

rsaCipher.init(Cipher.DECRYPT\_MODE, privateKey);

byte[] aesKeyBytes = rsaCipher.doFinal(Base64.getDecoder().decode(encryptedAesKeyBase64));

SecretKey aesKey = new SecretKeySpec(aesKeyBytes, "AES");

new File(outputFolderPath).mkdirs();

Cipher aesCipher = Cipher.getInstance("AES");

aesCipher.init(Cipher.DECRYPT\_MODE, aesKey);

for (File file : inputFolder.listFiles()) {

if (file.isFile() && file.getName().endsWith(".enc")) {

String encodedData = new String(new FileInputStream(file).readAllBytes());

byte[] decryptedData = aesCipher.doFinal(Base64.getDecoder().decode(encodedData));

String originalName = file.getName().replace(".enc", "");

try (FileOutputStream fos = new FileOutputStream(outputFolderPath + "/" + originalName)) {

fos.write(decryptedData);

}

System.out.println("Da giai ma: " + originalName);

}

}

System.out.println("Da luu o : " + outputFolderPath);

sc.close();

}

}

import java.io.\*;

import java.security.\*;

import javax.crypto.\*;

import javax.crypto.spec.SecretKeySpec;

import java.util.Base64;

import java.util.Scanner;

public class mhThuMuc {

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

Scanner sc = new Scanner(System.in);

System.out.print("Nhap duong dan thu muc muon ma hoa: ");

String folderPath = sc.nextLine();

System.out.print("Nhap ten thu muc de luu file ma hoa: ");

String outputFolder = sc.nextLine();

File folder = new File(folderPath);

if (!folder.exists() || !folder.isDirectory()) {

System.out.println("Thu muc khong hop le!");

return;

}

KeyGenerator keyGen = KeyGenerator.getInstance("AES");

keyGen.init(128);

SecretKey aesKey = keyGen.generateKey();

byte[] publicKeyBytes = new FileInputStream("publicKey").readAllBytes();

KeyFactory keyFactory = KeyFactory.getInstance("RSA");

PublicKey publicKey = keyFactory.generatePublic(new java.security.spec.X509EncodedKeySpec(publicKeyBytes));

Cipher rsaCipher = Cipher.getInstance("RSA");

rsaCipher.init(Cipher.ENCRYPT\_MODE, publicKey);

byte[] encryptedAesKey = rsaCipher.doFinal(aesKey.getEncoded());

new File(outputFolder).mkdirs();

try (FileOutputStream keyOut = new FileOutputStream(outputFolder + "/encrypted\_aes\_key.txt")) {

keyOut.write(Base64.getEncoder().encode(encryptedAesKey));

}

Cipher aesCipher = Cipher.getInstance("AES");

aesCipher.init(Cipher.ENCRYPT\_MODE, aesKey);

for (File file : folder.listFiles()) {

if (file.isFile()) {

byte[] data = new FileInputStream(file).readAllBytes();

byte[] encryptedData = aesCipher.doFinal(data);

String encodedData = Base64.getEncoder().encodeToString(encryptedData);

try (BufferedWriter writer = new BufferedWriter(

new FileWriter(outputFolder + "/" + file.getName() + ".enc"))) {

writer.write(encodedData);

}

System.out.println("Da ma hoa: " + file.getName());

}

}

System.out.println("Da ma hoa toan bo thu muc va luu vao : " + outputFolder);

sc.close();

}

}