链滴

使用 java 进行 AES 加密 解密

作者：chengzime

原文链接：https://ld246.com/article/1565605357506

来源网站：链滴

最近项目用到了加密技术,需要把数据进行加密传输,用到了AES;

百度百科是这样定义的:

高级加密标准（英语：Advanced Encryption Standard，缩写：AES），在密码学中又称Rijndael密法，是美国联邦政府采用的一种区块加密标准。这个标准用来替代原先的DES，已经被多方分析且为全世界所使用。经过五年的甄选流程，高级加密标准由美国国家标准与技术研究院（NIST）于200年11月26日发布于FIPS PUB 197，并在2002年5月26日成为有效的标准。2006年，高级加密标准已成为对称密钥加密中最流行的算法之一。

下面是代码

```
package chengzi;
import java.io.UnsupportedEncodingException;

import javax.crypto.Cipher;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;

import com.sun.org.apache.regexp.internal.recompile;

import sun.misc.BASE64Decoder;
import sun.misc.BASE64Encoder;

public class AesEncodeUtil {

    //偏移量
    public static final String VIPARA = "1234567876543210";  //AES 为16bytes. DES 为8bytes

    //编码方式
    public static final String CODE_TYPE = "UTF-8";
//    public static final String CODE_TYPE = "GBK";

    //填充类型
    public static final String AES_TYPE = "AES/ECB/PKCS5Padding";
    //public static final String AES_TYPE = "AES/ECB/PKCS7Padding";
    //此类型 加密内容,密钥必须为16字节的倍数位,否则抛异常,需要字节补全再进行加密
//    public static final String AES_TYPE = "AES/ECB/NoPadding";
    //java 不支持ZeroPadding
    //public static final String AES_TYPE = "AES/CBC/ZeroPadding";

    //私钥
    private static final String AES_KEY="1111222233334444";  //AES固定格式为128/192/256 bits
即：16/24/32bytes。DES固定格式为128bits，即8bytes。

    //字符补全
    private static final String[] consult = new String[]{"0","1","2","3","4","5","6","7","8","9","A","B"
"C","D","E","F","G"};


    /**
     * 加密
     *
     * @param cleartext
```

```
     * @return
     */
    public static String encrypt(String cleartext) {
        //加密方式：  AES128(CBC/PKCS5Padding) + Base64, 私钥：1111222233334444
        try {
            IvParameterSpec zeroIv = new IvParameterSpec(VIPARA.getBytes());
            //两个参数，第一个为私钥字节数组， 第二个为加密方式 AES或者DES
            SecretKeySpec key = new SecretKeySpec(AES_KEY.getBytes(), "AES");
            //实例化加密类，参数为加密方式，要写全
            Cipher cipher = Cipher.getInstance(AES_TYPE); //PKCS5Padding比PKCS7Padding效率
，PKCS7Padding可支持IOS加解密
            //初始化，此方法可以采用三种方式，按加密算法要求来添加。（1）无第三个参数（2）第
个参数为SecureRandom random = new SecureRandom();中random对象，随机数。(AES不可采
这种方法)（3）采用此代码中的IVParameterSpec
            //加密时使用:ENCRYPT_MODE;  解密时使用:DECRYPT_MODE;
            cipher.init(Cipher.ENCRYPT_MODE, key); //CBC类型的可以在第三个参数传递偏移量zeroIv
ECB没有偏移量
            //加密操作,返回加密后的字节数组，然后需要编码。主要编解码方式有Base64, HEX, UUE,7bi
等等。此处看服务器需要什么编码方式
            byte[] encryptedData = cipher.doFinal(cleartext.getBytes(CODE_TYPE));

            return new BASE64Encoder().encode(encryptedData);
        } catch (Exception e) {
            e.printStackTrace();
            return "";
        }
    }

    /**
     * 解密
     *
     * @param encrypted
     * @return
     */
    public static String decrypt(String encrypted) {
        try {
            byte[] byteMi = new BASE64Decoder().decodeBuffer(encrypted);
            IvParameterSpec zeroIv = new IvParameterSpec(VIPARA.getBytes());
            SecretKeySpec key = new SecretKeySpec(
             AES_KEY.getBytes(), "AES");
            Cipher cipher = Cipher.getInstance(AES_TYPE);
            //与加密时不同MODE:Cipher.DECRYPT_MODE
            cipher.init(Cipher.DECRYPT_MODE, key);  //CBC类型的可以在第三个参数传递偏移量zeroIv
ECB没有偏移量
            byte[] decryptedData = cipher.doFinal(byteMi);
            return new String(decryptedData, CODE_TYPE);
        } catch (Exception e) {
            e.printStackTrace();
            return "";
        }
    }

    /**
     * 测试
```

原文链接：使用 java 进行 AES 加密 解密

```
     *
     * @param args
     * @throws Exception
     */
    public static void main(String[] args) throws Exception {
        String content = "它是一只小跳蛙 越过蓝色大西洋,跳到遥远的东方 跳到我们身旁,春夏秋冬 我是最好的伙伴,亲吻它就会变得不一样";
        test(content);
    }

    public static void test(String content) throws UnsupportedEncodingException{
     System.out.println("加密内容：" + content);
     //字节数
     int num = content.getBytes(CODE_TYPE).length;
        System.out.println("加密内容字节数: " + num);
        System.out.println("加密内容是否16倍数: " + (num%16 == 0 ? true : false));

        //字节补全
        if(AES_TYPE.equals("AES/ECB/NoPadding")){
         System.out.println();
         content = completionCodeFor16Bytes(content);
         System.out.println("加密内容补全后: "+content);
        }

        System.out.println();

        // 加密
        String encryptResult = encrypt(content);
        content = new String(encryptResult);
        System.out.println("加密后：" + content);

        System.out.println();

        // 解密
        String decryptResult = decrypt(encryptResult);
        content = new String(decryptResult);
        //还原
        if(AES_TYPE.equals("AES/ECB/NoPadding")){
         System.out.println("解密内容还原前: "+content);
         content = resumeCodeOf16Bytes(content);
        }

        System.out.println("解密完成后：" + content);
    }



    //NoPadding
    //补全字符
    public static String completionCodeFor16Bytes(String str) throws UnsupportedEncodingException{
     int num = str.getBytes(CODE_TYPE).length;
     int index = num%16;
```

```
    //进行加密内容补全操作, 加密内容应该为 16字节的倍数, 当不足16*n字节是进行补全, 差一位时全16+1位
    //补全字符 以 $ 开始,$后一位代表$后补全字符位数,之后全部以0进行补全;
    if(index != 0){
    StringBuffer sbBuffer = new StringBuffer(str);
    if(16-index == 1){
    sbBuffer.append("$" + consult[16-1] + addStr(16-1-1));
    }else{
    sbBuffer.append("$" + consult[16-index-1] + addStr(16-index-1-1));
    }
    str = sbBuffer.toString();
    }
    return str;
   }

   //追加字符
   public static String addStr(int num){
    StringBuffer sbBuffer = new StringBuffer("");
    for (int i = 0; i < num; i++) {
    sbBuffer.append("0");
}
    return sbBuffer.toString();
   }


   //还原字符(进行字符判断)
   public static String resumeCodeOf16Bytes(String str){
    int indexOf = str.lastIndexOf("$");
//     System.out.println(indexOf);
    if(indexOf == -1){
    return str;
    }
    String trim = str.substring(indexOf+1,indexOf+2).trim();
//     System.out.println(trim);
    int num = 0;
    for (int i = 0; i < consult.length; i++) {
    if(trim.equals(consult[i])){
    num = i;
    }
}
    if(num == 0){
    return str;
    }
    return str.substring(0,indexOf).trim();
   }

}
```

本文查阅了几位大神的博文之后自己整理编写测试成功,期间被卡在了NoPadding填充类型上,在在线密解密上有种类型 zeropadding ,翻译成中文与 java中的Nopadding大意近似,一度以为此两种 类型实是一种,经过测试发现确实他们加密后相似度很大,但是还是不一样,后来发现java其实是不支持 zeropdding 填充类型的;

原文链接：使用 java 进行 AES 加密 解密

Nopadding 在使用时发现 此类型下 密钥 和 加密内容 都必须要16字节的倍数,在不满足前条件的情下会抛异常,于是咨询了 "匆忙拥挤repeat" 进行了字符补全的思路加密,再次感谢!!!

原文链接：使用 java 进行 AES 加密 解密