操作指南

# 加密

```
java -jar encrypt.jar c:\work\Firefox-47.0.1.exe c:\work\id.rsa c:\work\123 null 123456 null true null
```

或

```
java -jar encrypt.jar "c:\work\source.exe" "c:\work\id.rsa" "c:\work\123" "null" "123456" "null" "true" "null"
```

参数说明：
param1： sourceFilePath 源文件，不许为空
param2： rsaFilePath 加密密钥文件，不许为空
param3： targetFilePath 加密后的文件，不许为空
param4： macAddr mac地址，如”6A-00-E3-D7-49-C9”（17位）
param5： password 密码（六位数字字母组合，默认”123456”）
param6： expired 有效期(默认30天,单位：ms)
param7： isForever 是否永久(“false”默认不永久)
param8： start 起始时间(“yyyy-MM-dd HH:mm:ss”)

---

# 解密

```
java -jar decrypt.jar c:\work\123 c:\work\id.rsa c:\work\copy.exe 123456
```

或

```
java -jar decrypt.jar "c:\work\123" "c:\work\id.rsa" "c:\work\copy.exe" "123456"
```

参数说明：
param1： sourceFilePath 源文件，不许为空
param2： rsaFilePath 解密密钥文件，不许为空
param3： targetFilePath 解密后的文件，不许为空
param4： password 密码（六位数字字母组合）

注：注意空格！

# 加解密原理

# 主流加密算法

## 一、对称加密（AES）

采用单钥密码系统的加密方法，同一个密钥可以同时用作信息的加密和解密，这种加密方法称为对称加密，也称为单密钥加密。

所谓对称，就是采用这种加密方法的双方使用方式用同样的密钥进行加密和解密。密钥是控制加密及解密过程的指令。算法是一组规则，规定如何进行加密和解密。

**优点**：算法公开、计算量小、加密速度快、加密效率高。

**缺点**：在数据传送前，发送方和接收方必须商定好秘钥，然后使双方都能保存好秘钥。其次如果一方的秘钥被泄露，那么加密信息也就不安全了。另外，每对用户每次使用对称加密算法时，都需要使用其他人不知道的唯一秘钥，这会使得收、发双方所拥有的钥匙数量巨大，密钥管理成为双方的负担。

## 二、非对称加密(RSA)

而非对称加密算法需要两个密钥来进行加密和解密，这两个秘钥是公开密钥（public key，简称**公钥**）和私有密钥（private key，简称**私钥**）。

公开密钥与私有密钥是一对，如果用公开密钥对数据进行加密，只有用对应的私有密钥才能解密；如果用私有密钥对数据进行加密，那么只有用对应的公开密钥才能解密。因为加密和解密使用的是两个不同的密钥，所以这种算法叫作非对称加密算法。

**优点**：非对称加密与对称加密相比，其安全性更好：对称加密的通信双方使用相同的秘钥，如果一方的秘钥遭泄露，那么整个通信就会被破解。而非对称加密使用一对秘钥，一个用来加密，一个用来解密，而且公钥是公开的，秘钥是自己保存的，不需要像对称加密那样在通信之前要先同步秘钥。

**缺点**：加密和解密花费时间长、速度慢，只适合对少量数据进行加密。

## 三、原理

综合上述两大类加密机制，决定采用AES对称加密文件、RSA非对称加密密钥，适当加入一些验证来保证不同需求。

注：加密解密分开为两个jar包模块encrypt/decrpt，互不依赖，可分开单独使用。encrypt模块用于加密，会生成一个加密后的文件以及加密的密钥；decrypt模块用于验证及解密，验证密钥合法后还原加密文件。

# 加密流程

# encrypt.jar

## 一、AesEncode

作用：加密源文件

```
/**
 * 加密文件.
 */
public static void encrypt(String srcFile, String destFile, String privateKey) {
...
}
```

## 二、RsaEncode

作用：对密钥加密

```
/**
 * 加密密钥.
 */
public static byte[] encrypt(RSAPublicKey publicKey, byte[] srcBytes) throws NoSuchAlgorithmException, NoSuchPadding
Exception, InvalidKeyException, IllegalBlockSizeException, BadPaddingException {
...
}
```

## 三、FileEncrypt

作用：文件加密入口

```
/**
 * 加密.
 */
public static void encode(ValidateBean bean, String rsaFile, String srcFile, String targetFile) throws Exception {
...
}
```

## 四、工具类/javaBean

1.util包：
(1)SecutityUtils
用于生成本机mac地址

```
/**
 * 获取本机mac地址：6A-00-E3-D7-49-C9（17位）.
 */
public static String getLocalMac(){...}
```

获取AES的密钥器Cipher

```
/**
 * 获取AES的Cipher.
 */
public static Cipher getCipher(int mode, String key){...}
```

生成密钥文件

```
/**
 * 生成文件.
 */
public static File mkdirFiles(String filePath) {...}
```

(2)MD5Utils
md5不可逆加密，主要用于验证validateBean的macAddr和password加密，增加安全性和RSA加密内容不能超过117位。

```
public static void Md5(String plainText) {...}
```

2.bean包：
(1)ValidateBean
合法验证bean,主要属性有：

```
//mac地址(格式：6A-00-E3-D7-49-C9（17位）)
private String macAddr;
//起始时间(单位：ms，默认当前时间)
private long start = System.currentTimeMillis();
//有效期(默认30天)
private long expired = 2592000000L;
//是否永久(默认不永久，如果设置永久则有效期不起效)
private boolean isForever = false;
//密码（六位数字字母组合，默认123456）
private String password = "123456";
//通过md5不可逆加密macAddr和password (32位)
private String md5;
```

另重载了一些构造方法，适用不同场合。

(2)RsaBean
验证文件对应bean，主要包含RSA私钥和加密密文。

```
private RSAPrivateKey priKey;
private byte[] resultBytes;
```

## 五、调用加密方法

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

    ValidateBean srcBean = new ValidateBean(SecurityUtils.getLocalMac(), 360000, System.currentTimeMillis(), true, "com
.111111fu");
    FileEncrypt.encode(srcBean, "C:\\work\\id.rsa", "C:\\work\\源文件.txt", "C:\\work\\encode\\加密文件");
```

# 解密流程

# decrypt.jar

## 一、RsaDecode

作用：对密钥解密

```
1.  public static byte[] decrypt(RSAPrivateKey privateKey, byte[] srcBytes) throws NoSuchAlgorithmException, NoSuchPaddin
gException, InvalidKeyException, IllegalBlockSizeException, BadPaddingException {...}
```

## 二、AesDecode

作用：解密加密文件

```
1.    public static void decrypt(String srcFile, String destFile, String privateKey) {
2.   ...
3.   }
```

## 三、FileDecrypt

作用：文件解密入口

```
1.  public static void decode(ValidateBean bean, String rsaFile, String srcFile, String targetFile) throws Exception {..}
```

其中包含verify()一些合法验证。

## 四、工具类/javaBean

1.util包：
(1)SecutityUtils
用于生成本机mac地址

```
1.  /**
2.   * 获取本机mac地址：6A-00-E3-D7-49-C9（17位）.
3.   */
4.  public static String getLocalMac(){...}
```

获取AES的密钥器Cipher

```
1.  /**
2.   * 获取AES的Cipher.
3.   */
4.  public static Cipher getCipher(int mode, String key){...}
```

(2)MD5Utils
md5不可逆加密，主要用于验证validateBean的macAddr和password加密，增加安全性和RSA加密内容不能超过117位。

```
1.  public static void Md5(String plainText) {...}
```

2.bean包：
(1)ValidateBean

合法验证bean,主要属性有：

```
//mac地址(格式：6A-00-E3-D7-49-C9（17位）)
private String macAddr;
//起始时间(单位：ms，默认当前时间)
private long start = System.currentTimeMillis();
//有效期(默认30天)
private long expired = 2592000000L;
//是否永久(默认不永久，如果设置永久则有效期不起效)
private boolean isForever = false;
//密码（六位数字字母组合，默认123456）
private String password = "123456";
//通过md5不可逆加密macAddr和password (32位)
private String md5;
```

另重载了一些构造方法，适用不同场合。

(2)RsaBean
验证文件对应bean，主要包含RSA私钥和加密密文。

```
private RSAPrivateKey priKey;
private byte[] resultBytes;
```

## 五、调用解密方法

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

    ValidateBean bean = new ValidateBean();
    bean.setPassword("com.111111fu");
    FileDecrypt.decode(bean, "C:\\work\\id.rsa", "C:\\work\\encode\\加密文件", "C:\\work\\decode\\解密后的文件.txt");
```