

# 基于 SHA256 和 RSA 的数字签名

智能 1502 陈东宁 0918150203

## 1. RSA 算法实现

在尝试了自己动手造车轮，用 C++编写大数类和素性检测函数，以及尝试使用 C 及 C++的附加大数类库（譬如 MPUINT, GMP, GUN Crypto, boost 等）之后，终于意识到自己编写的类和函数过于低效，而附加类库又过于庞杂。

在经过无数次失败的尝试后，决定转用标准库自带了大数类的 JAVA。

果然，JAVA 原生的大数库使用简便而且高效。剩下的只需要自己编写 RSA 的算法实现即可。终于让我意识到了 C++的局限性。

不过，这个过程也不是一无所获。查阅了大量的资料，让我深刻地了解了 RSA 算法和素性检测算法，同时也意识到了大数运算的复杂性。

## 2. SHA 函数

相较于 RSA，SHA 的实现过程就没那么坎坷了，但也绝非易事。在 Wikipedia 上查找了 SHA256 的伪代码实现，将之转化为 JAVA 代码。

第一次运行，计算英文 Hello 的 hash 值并于网上的在线 hash 值计算软件比对，发现不一致。遂查看网页的源代码，然后逐条执行，与

自己的程序比对，反复纠错。其中还遇到了负数右移的问题。在查找资料后，将右移>>改为无符号右移>>>，配合其他一些地方的改动，终于顺利解决。

### 3. 签名函数

与前两者比较起来，签名函数是最轻松的。基本没遇到什么问题，按着公式顺利解决。

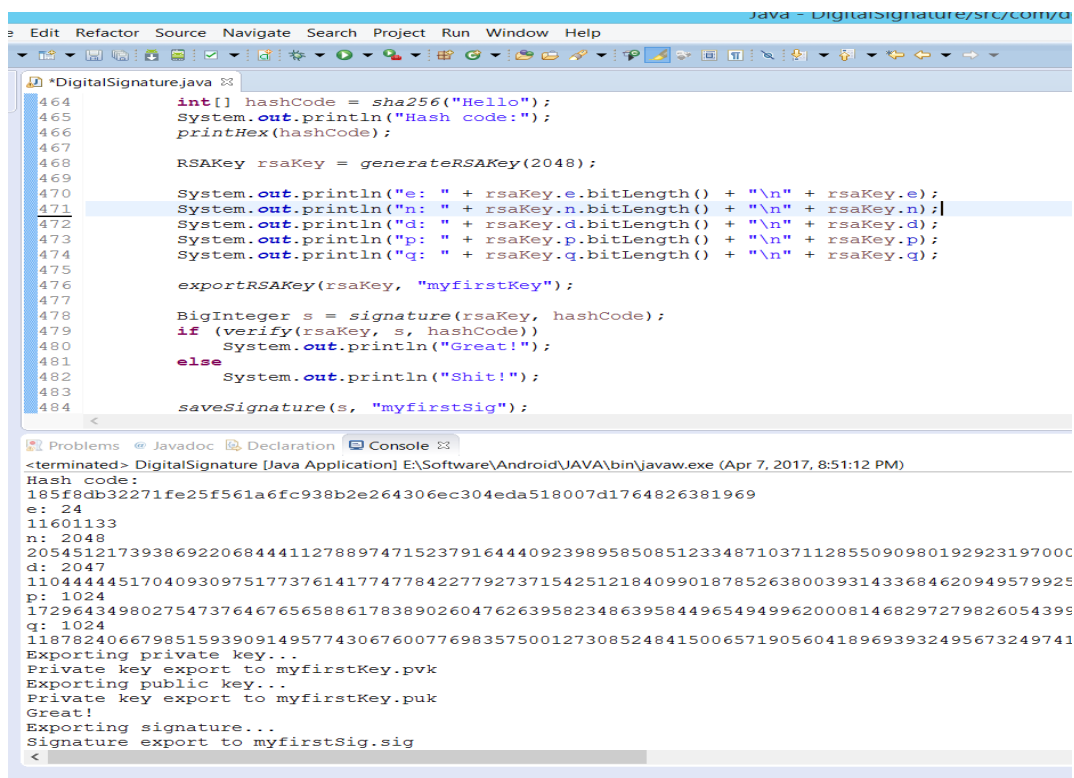


图1 主要功能通过测试  
正确计算出 hash 值，成功生成并导出 RSA 密钥，并对文件进行签名，导出签名

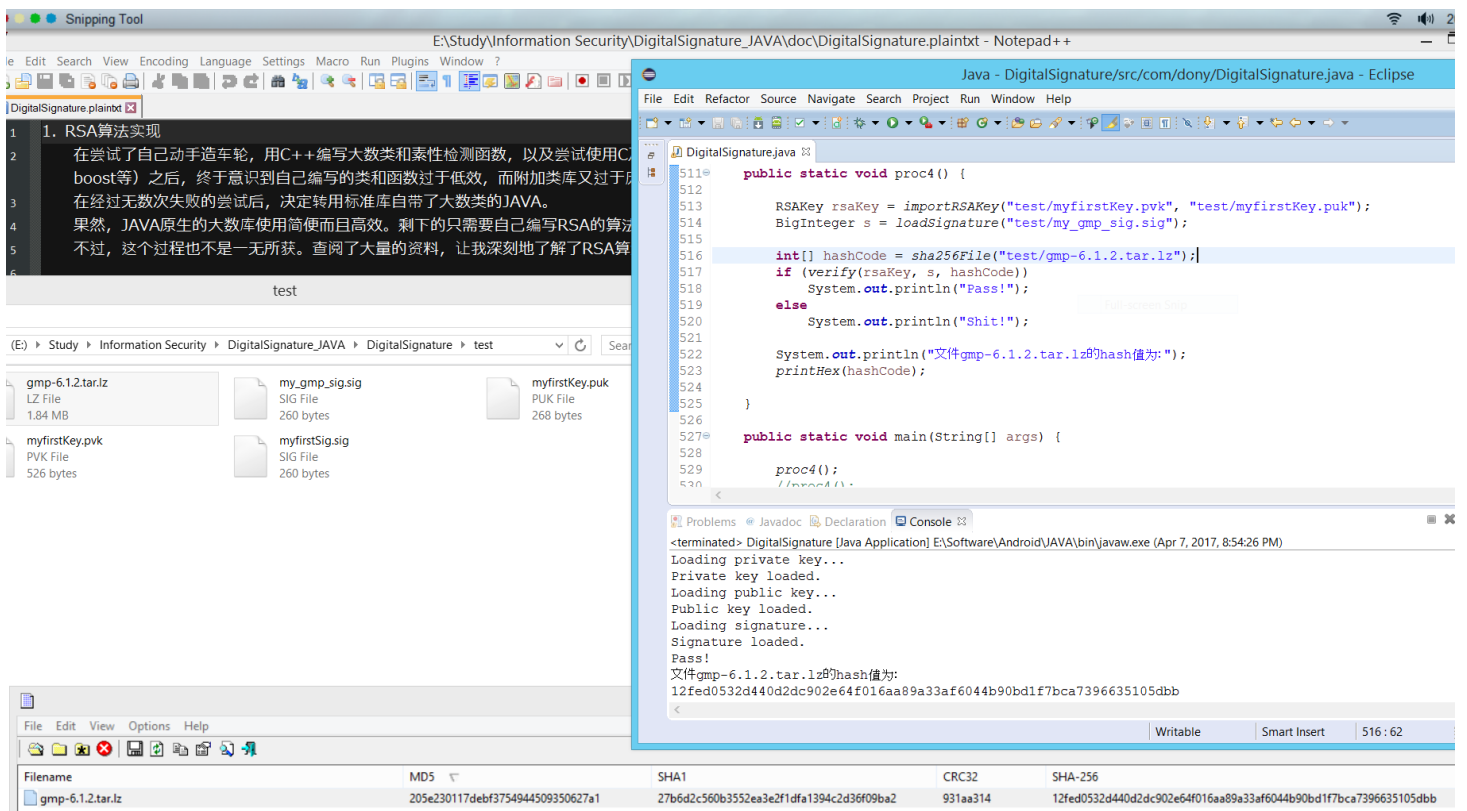


图2 验证签名功能通过测试  
成功读取 RSA 密钥，文件 hash 值正确，签名验证通过

```
C:\Windows\system32\cmd.exe

Verify filename publicKeyFilename signatureFilename

E:\Study\Information Security\DigitalSignature_JAVA\DigitalSignature\bin>java -jar DigitalSignature.jar GenerateRSAKey 2048 testKey1
RSA key information:
e: 24
10205963
n: 2048
21908643028161507022187015369775222827456091738259908863452346364240417218207198
38515474368069520044523202675083598761544657162881533333863457400611495223439373
84583414597393641447563443648171081174661092124639639440446999761129415320994727
05097842945422844899453836774770050762098541827678308119160172141311111276426214
52507218074589472595515780008998711627171523572982563662734007219386468248166342
9569707094628577050176732213237390320385939771045752671644355819342054713108485
30952709723365522699332821331860390006457561992692525635431876314933783122539093
946914304827965552481517365216692310206231455418813109487
d: 2046
70022002871394425653706783770376646765121448951638650267549226611688062411729288
20348181861667137814031891183128277365947951215806098306773423769308047928697091
54976911612065487718653548888972962926759738694937737285843576511911458198534012
00129126144047156930450032343527381435542095695875804628563429429285881641803900
25310208873951052045940697764233817362475132947703525661116989381400417200825050
39843909659464610638966132974703068566190650217973092749137102281595071895076934
83686570139781756003032876621014484395141370220996556293440276610681156103008878
95820155701260061082789802184188445781838829518986572643
p: 1024
17015545866665560612395185661752388098884395042713629974483270074866975100519914
92386017459517514727231059832353358778534171685802395759896878862213674390313346
36262262809938695765322376663331810783673969805436844426061668077569088278543986
851103066063512622650142942869963991009648045631871860535327800157329
q: 1024
12875662761476143420090746582151543617344461033097736896879437947318416112297590
02725187170939215194088538865818855970192221458419386074088977461428998006437338
08375257933877489551767467188073455742104855871551048898441155215404508793652683
466046860205485323482051222802573060351864952574841593613600143237503
Exporting private key...
Private key export to testKey1.pvk
Exporting public key...
Private key export to testKey1.puk

E:\Study\Information Security\DigitalSignature_JAVA\DigitalSignature\bin>java -jar DigitalSignature.jar GetHashValue DigitalSignature.jar
Hash code of DigitalSignature.jar is:
d5db0886dacb9305750e6b8c5880b11d97dc8406f448a326dc5c609170e9340e

E:\Study\Information Security\DigitalSignature_JAVA\DigitalSignature\bin>java -jar DigitalSignature.jar Signature testKey1.pvk DigitalSignature.jar sign-DigitalSignature
Loading private key...
Private key loaded.
Not loading public key.
Hash code of DigitalSignature.jar is:
d5db0886dacb9305750e6b8c5880b11d97dc8406f448a326dc5c609170e9340e
Signaturing DigitalSignature.jar ...
Exception in thread "main" java.lang.NullPointerException
```

生成 RSA 密钥

e 为公钥

n 为模数

d 为私钥

p, q 为所用的素数

计算文件

hash 值

图 3 在控制台生成 RSA 密钥和计算文件 hash 值  
(由于控制台不支持中文因此输出英文)

```

E:\Study\Information Security\DigitalSignature_JAVA\DigitalSignature\bin>java -jar DigitalSignature.jar Signature testKey1.pvk DigitalSignature.jar sign-DigitalSignature
Loading private key...
Private key loaded.
Not loading public key.
Hash code of DigitalSignature.jar is:
6f1fe54ea677e529e7d6bb7693b09c3a54ad5fb2bbc955b5fcedf5c288e1fe21
Signaturing DigitalSignature.jar ...
Signature of DigitalSignature.jar is:
5b46ac00c3a7ed02585058c0e13d498eeb6ac503b860e27c23f0f05cd7f92ac9eb58db65a6555646
7484a0cc057e00b468fed3b3d548cf74c528263ee64328a74e6243076f88c1197620d438864eae9f
21df10cd5ec6ecc728a98994e8a7fb8bb136acfdc72daa77760646f16ef5435781ad3795c40988fa
0bbdfa40771922efb7f743e9371e738f1c904c959d9117f8c00a24b07a5c42e308c31a0d11d3fd69
b94ea4d7179779ad08a1be82b62c19f184f8c73ffc71f9a97ab317f2869bbbbee121ad0208f6eb250
c934a3f454edf4c1bace1a9dd09aa5dcb8101651c2f7463a61ada245f3132adf9461551fd42c91b7
e1f0c92cbfe6c8f81db6169b03a7a7d7
Saving signature...
Signature saved to sign-DigitalSignature.sig

E:\Study\Information Security\DigitalSignature_JAVA\DigitalSignature\bin>java -jar DigitalSignature.jar Verify testKey1.puk DigitalSignature.jar sign-DigitalSignature.sig
testKey1.puk
Not loading private key.
Loading public key...
Public key loaded.
Loading signature...
Signature loaded.
Hash code of DigitalSignature.jar is:
6f1fe54ea677e529e7d6bb7693b09c3a54ad5fb2bbc955b5fcedf5c288e1fe21
Pass!
The file DigitalSignature.jar is verified!

E:\Study\Information Security\DigitalSignature_JAVA\DigitalSignature\bin>

```

用生成的 RSA  
私钥对文件  
签名

用生成的 RSA  
公钥对签名  
进行验证

图 4 用生成的 RSA 密钥签名文件并验证通过