Rsa接口使用介绍及性能测试

**Rsa库的编译环境**

编译环境硬件信息

服务器型号

PowerEdge R620

CPU

类型 Intel(R) Xeon(R) CPU E5-2630 v2 @ 2.60GHz

数量 2个

核数 6核

内存

64G

软件信息

主机名

10.68.8.39

操作系统

CentOS release 6.4

内核版本

2.6.32-358.el6.x86\_64

Gcc环境

gcc-gfortran-4.4.7-3.el6.x86\_64

gcc-4.4.7-3.el6.x86\_64

libgcc-4.4.7-3.el6.x86\_64

gcc-java-4.4.7-3.el6.x86\_64

gcc-c++-4.4.7-3.el6.x86\_64

libgcc-4.4.7-3.el6.i686

依赖文件

PWSL/include文件下的octets.h文件

PolarSSL源码库中裁剪的bignum.c bignum.h rsa.c rsa.h config.h bn\_mul.h

**Rsa库的接口介绍**

相关接口是对PolarSSL库中的Rsa算法做了简单包装，主要接口函数在rsa.h文件中，感兴趣的可以看一下PolarSSL库及其相关测试用例。

加密接口函数

GNET::Octets rsa\_encode(const GNET::Octets& pubkey, const GNET::Octets& in)

输入参数：GNET::Octets& pubkey

Pubkey为输入公钥

输入参数：GNET::Octets& in

in为输入的加密原始字符串

in的最大长度为114 (128 – 14)

这14个字节包括11个padding字节，1个结束符字节，1个长度字节

返回值 表示公钥加密后的字符串

解密接口函数

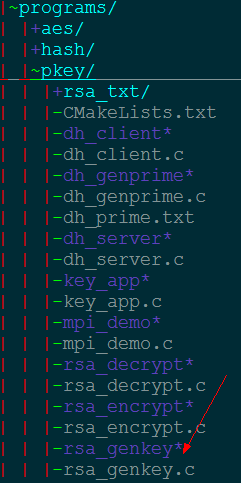
GNET::Octets rsa\_decode(const GNET::Octets& prikey, const GNET::Octets& in)

输入参数：GNET::Octets& prikey

Prikey为输入私钥

这里选取128字节长度的私钥

代码中用到的私钥是用PolarSSL库编译后中rsa\_genkey生成的，如下图



输入参数：GNET::Octets& in

In为加加密后的字符串

返回值 表示解密后的字符串 也等于原始字符串

**Rsa加密和解密性能测试**

测试环境

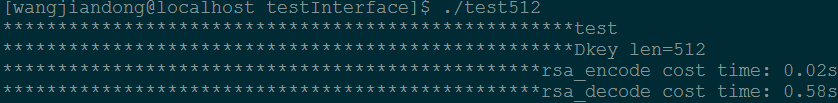
Rsa测试环境与Rsa库编译环境一样

测试内容

对于公钥 （N，e） 分别选取N的长度为64字节，128字节，256字节进行加密，

对于私钥（N，d）分别选取对应长度的字节解密，重复1000次，分别计算出所需要的时间

测试分析

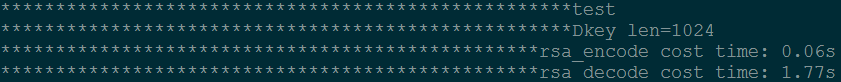


上述结果是对私钥长度为64字节的进行加密结果，上图分别是对一个32位字符串累计1000次加密所的到结果。

0.02s为进行1000次加密的累计时间

0.58s为进行1000次解密的累计时间

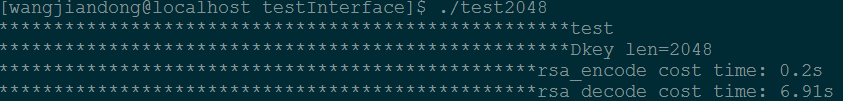
故可得知rsa加密算法消耗的主要时间是在解密上



私钥长度为128字节的加密结果如上图

0.06为进行1000次加密的累计时间

1.77s为进行1000次解密的累计时间



私钥长度为256字节的加密结果如上图

0.2s为进行1000次加密的累计时间

6.91s为进行1000次解密的累计时间

结果分析

Rsa加密算法的主要时间消耗在解密上，其次还和私钥长度有关系。

以下是本库开发中用到的N,E,D

N:

"9292758453063D803DD603D5E777D788" \

"8ED1D5BF35786190FA2F23EBC0848AEA" \

"DDA92CA6C3D80B32C4D109BE0F36D6AE" \

"7130B9CED7ACDF54CFC7555AC14EEBAB" \

"93A89813FBF3C4F8066D2D800F7C38A8" \

"1AE31942917403FF4946B0A83D3D3E05" \

"EE57C6F5F5606FB5D4BC6CD34EE0801A" \

"5E94BB77B07507233A0BC7BAC8F90F79"

E:

"10001"

D:

"24BF6185468786FDD303083D25E64EFC" \

"66CA472BC44D253102F8B4A9D3BFA750" \

"91386C0077937FE33FA3252D28855837" \

"AE1B484A8A9A45F7EE8C0C634F99E8CD" \

"DF79C5CE07EE72C7F123142198164234" \

"CABB724CF78B8173B9F880FC86322407" \

"AF1FEDFDDE2BEB674CA15F3E81A1521E" \

"071513A1E85B5DFA031F21ECAE91A34D"

**扩展：**

目前长度为1024公钥长度的加密，基本很难破解，如果需要用其他私钥

，可修改rsa.h 文件中的宏定义

KEY\_LEN

RSA\_N

RSA\_E

RSA\_D

RSA\_P

RSA\_Q

RSA\_DP

RSA\_DQ

RSA\_QP

这些参数可以通过polar库中rsa\_genkey生成。