OpenSSL编程-RSA编程详解

本文由 大佟 发表于 2014年06月26日，浏览: 30,013次，评论: 0条

一. RSA PEM文件格式

1. PEM私钥格式文件

```
-----BEGIN RSA PRIVATE KEY-----
-----END RSA PRIVATE KEY-----
```

2. PEM公钥格式文件

```
-----BEGIN PUBLIC KEY-----
-----END PUBLIC KEY-----
```

3. PEM RSAPublicKey公钥格式文件

```
-----BEGIN RSA PUBLIC KEY-----
-----END RSA PUBLIC KEY-----
```

二. OpenSSL密钥相关命令

1. 生成密钥

```
openssl genrsa -out key.pem 1024
    -out 指定生成文件，此文件包含公钥和私钥两部分，所以即可以加密，也可以解密
    1024 生成密钥的长度
```

2. 提取PEM格式公钥

```
openssl rsa -in key.pem -pubout -out pubkey.pem
    -in 指定输入的密钥文件
    -out 指定提取生成公钥的文件(PEM公钥格式)
```

3. 提取PEM RSAPublicKey格式公钥

```
openssl rsa -in key.pem -RSAPublicKey_out -out pubkey.pem
    -in 指定输入的密钥文件
    -out 指定提取生成公钥的文件(PEM RSAPublicKey格式)
```

站
搜索.
近
网络编
网络编
网络编
电子邮
Linux
近
大佟发
leilux发
OpenS
大佟发
icewar
分
C/OO
Pytho
互联网
开发环

4. 公钥加密文件

```
openssl rsautl -encrypt -in input.file -inkey pubkey.pem -pubin -out ou
    -in 指定被加密的文件
    -inkey 指定加密公钥文件
    -pubin 表面是用纯公钥文件加密
    -out 指定加密后的文件
```

5. 私钥解密文件

```
openssl rsautl -decrypt -in input.file -inkey key.pem -out output.file
    -in 指定需要解密的文件
    -inkey 指定私钥文件
    -out 指定解密后的文件
```

三. RSA相关API

1. 基本数据结构

```
struct {
    BIGNUM *n;              // public modulus
    BIGNUM *e;              // public exponent
    BIGNUM *d;              // private exponent
    BIGNUM *p;              // secret prime factor
    BIGNUM *q;              // secret prime factor
    BIGNUM *dmp1;           // d mod (p-1)
    BIGNUM *dmq1;           // d mod (q-1)
    BIGNUM *iqmp;           // q^-1 mod p
    // ...
} RSA;
```

2. BN大数系列函数

```
//新生成一个BIGNUM结构
BIGNUM *BN_new(void);

//释放一个BIGNUM结构，释放完后a=NULL;
void BN_free(BIGNUM *a);

//初始化所有项均为0，一般为BN_ init(&c)
void BN_init(BIGNUM *);

//将a中所有项均赋值为0，但是内存并没有释放
void BN_clear(BIGNUM *a);

//相当与将BN_free和BN_clear综合，要不就赋值0，要不就释放空间。
void BN_clear_free(BIGNUM *a);

//设置大数a为整数w
int BN_set_word(BIGNUM *a, unsigned long w);

//假如大数a能表示为long型，那么返回一个long型数
unsigned long BN_get_word(BIGNUM *a);

//产生一个加密用的强bits的伪随机数
//若top=-1,最高位为0,top=0，最高位为1,top=1,最高位和次高位为1,bottom为真,
int BN_rand(BIGNUM *rnd, int bits, int top, int bottom);

//将 a 转化为字符串存入to，to的空间必须大于BN_num_bytes(a)
```

技术架
电子邮
编程开

文

2017年
2017年
2016年
2015年
2014年
2014年
2014年
2014年
2014年
2014年

友

晨帆起
板桥居
运维生

```
int BN_bn2bin(const BIGNUM *a, unsigned char *to);

//将s中的len位的正整数转化为大数
BIGNUM *BN_bin2bn(const unsigned char *s, int len, BIGNUM *ret);

//将大数转化为16进制字符串
char *BN_bn2hex(const BIGNUM *a);

//将大数转化为10进制字符串
char *BN_bn2dec(const BIGNUM *a);

//将16进制字符串转成大数
int BN_hex2bn(BIGNUM **a, const char *str);

//将10进制字符串传成大数
int BN_dec2bn(BIGNUM **a, const char *str);
```

3. RSA系列函数

```
//初始化一个RSA结构
RSA * RSA_new(void);

//释放一个RSA结构
void RSA_free(RSA *rsa);

//RSA私钥产生函数
//产生一个模为num位的密钥对，e为公开的加密指数，一般为65537（0x10001）
RSA *RSA_generate_key(int num, unsigned long e,void (*callback)(int,in

//判断位数函数，返回RSA模的位数
int RSA_size(const RSA *rsa);

//测试p、q是否为素数
int RSA_check_key(RSA *rsa);
```

4. PEM系列函数

```
//从文件中加载RSAPublicKey格式公钥证书
RSA *PEM_read_RSAPublicKey(FILE *fp, RSA **x, pem_password_cb *cb, voi

//从BIO重加载RSAPublicKey格式公钥证书
RSA *PEM_read_bio_RSAPublicKey(BIO *bp, RSA **x, pem_password_cb *cb,

//输出RSAPublicKey公钥证书到文件
int PEM_write_RSAPublicKey(FILE *fp, RSA *x);

//输出RSAPublicKey公钥证书到BIO
int PEM_write_bio_RSAPublicKey(BIO *bp, RSA *x);
```

5. RSA加密API

```
int RSA_public_encrypt(int flen, unsigned char *from, unsigned char *to

参数说明：
    flen: 要加密信息长度
    from: 要加密信息
    to: 加密后的信息
    padding: 采取的加密方案，分为: RSA_PKCS1_PADDING, RSA_PKCS1_OAEP_PADI
```

6. RSA解密API

```
int RSA_private_decrypt(int flen, unsigned char *from, unsigned char *t

参数说明：
    flen: 要解密的信息长度
    from: 要解密的信息
    to: 解密后的信息
    padding: 采取的解密方案
```

四. RSA编程示例

1. 数据加、密解密示例

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<openssl/rsa.h>
#include<openssl/pem.h>
#include<openssl/err.h>

#define PRIKEY "prikey.pem"
#define PUBKEY "pubkey.pem"
#define BUFFSIZE 4096

/*****************************************************************
 * RSA加密解密函数
 *
 * file: test_rsa_encdec.c
 * gcc -Wall -O2 -o test_rsa_encdec test_rsa_encdec.c -lcrypto -lssl
 *
 * author: tonglulin@gmail.com by www.qmailer.net
 *****************************************************************/

char *my_encrypt(char *str, char *pubkey_path)
{
    RSA *rsa = NULL;
    FILE *fp = NULL;
    char *en = NULL;
    int len = 0;
    int rsa_len = 0;

    if ((fp = fopen(pubkey_path, "r")) == NULL) {
        return NULL;
    }

    /* 读取公钥PEM，PUBKEY格式PEM使用PEM_read_RSA_PUBKEY函数 */
    if ((rsa = PEM_read_RSAPublicKey(fp, NULL, NULL, NULL)) == NULL)
        return NULL;
    }

    RSA_print_fp(stdout, rsa, 0);

    len = strlen(str);
    rsa_len = RSA_size(rsa);

    en = (char *)malloc(rsa_len + 1);
    memset(en, 0, rsa_len + 1);

    if (RSA_public_encrypt(rsa_len, (unsigned char *)str, (unsigned c
        return NULL;
    }

    RSA_free(rsa);
```

```
51      fclose(fp);
52
53      return en;
54  }
55
56  char *my_decrypt(char *str, char *prikey_path)
57  {
58      RSA *rsa = NULL;
59      FILE *fp = NULL;
60      char *de = NULL;
61      int rsa_len = 0;
62
63      if ((fp = fopen(prikey_path, "r")) == NULL) {
64          return NULL;
65      }
66
67      if ((rsa = PEM_read_RSAPrivateKey(fp, NULL, NULL, NULL)) == NULL)
68          return NULL;
69      }
70
71      RSA_print_fp(stdout, rsa, 0);
72
73      rsa_len = RSA_size(rsa);
74      de = (char *)malloc(rsa_len + 1);
75      memset(de, 0, rsa_len + 1);
76
77      if (RSA_private_decrypt(rsa_len, (unsigned char *)str, (unsigned
78          return NULL;
79      }
80
81      RSA_free(rsa);
82      fclose(fp);
83
84      return de;
85  }
86
87  int main(int argc, char *argv[])
88  {
89      char *src = "hello, world!";
90      char *en = NULL;
91      char *de = NULL;
92
93      printf("src is: %s\n", src);
94
95      en = my_encrypt(src, PUBKEY);
96      printf("enc is: %s\n", en);
97
98      de= my_decrypt(en, PRIKEY);
99      printf("dec is: %s\n", de);
100
101     if (en != NULL) {
102         free(en);
103     }
104
105     if (de != NULL) {
106         free(de);
107     }
108
109     return 0;
110 }
```

2. PEM/BIGNUM公钥转换示例

```c
#include <stdlib.h>
#include <string.h>

#include <openssl/rsa.h>
#include <openssl/pem.h>

/*****************************************************************
 * RSA PEM/BIGNUM公钥转换函数
 *
 * file: test_rsa_pubkey.c
 * gcc -Wall -O2 -o test_rsa_pubkey test_rsa_pubkey.c -lcrypto -lssl
 *
 * author: tonglulin@gmail.com by www.qmailer.net
 *****************************************************************/

const char *n = "C7301B330C4E123E4FA9F54F49121E8CE07974D8BFEF1D39EC92

const char *pubkey = "-----BEGIN RSA PUBLIC KEY-----\nMIGJAoGBAMcwGzM

int main(int argc, char *argv[])
{
    RSA    *rsa = NULL;
    BIO    *bio = NULL;
    BIGNUM *bne = NULL;
    BIGNUM *bnn = NULL;
    FILE *fp = NULL;
    unsigned long e = 65537;

    if (argc < 2) {
        printf("%s pem|bignum args\n", argv[0]);
        return -1;
    }

    /* 将PEM转换为大数字符串 */
    if (strcasecmp(argv[1], "bignum") == 0) {
        if (argc == 3) {
            /* 从外部文件读 */
            fp = fopen(argv[2], "r");
            if (fp == NULL) {
                return -1;
            }

            rsa = PEM_read_RSAPublicKey(fp, &rsa, NULL, NULL);
            if (rsa == NULL) {
                return -1;
            }
        }
        else {
            /* 从内存数据读 */
            bio = BIO_new(BIO_s_mem());
            BIO_puts(bio, pubkey);

            rsa = PEM_read_bio_RSAPublicKey(bio, &rsa, NULL, NULL);
            if (rsa == NULL) {
                return -1;
            }
        }

        RSA_print_fp(stdout, rsa, 0);
        printf("%s\n", BN_bn2hex(rsa->n));
        printf("%s\n", BN_bn2hex(rsa->e));

        if (argc == 3) {
```

```
64              fclose(fp);
65          }
66          else {
67              BIO_free(bio);
68          }
69          RSA_free(rsa);
70      }
71      /* 将大数字符串转换为PEM文件 */
72      else if (strcasecmp(argv[1], "pem") == 0) {
73
74          bne = BN_new();
75          if (bne == NULL) {
76              return -1;
77          }
78
79          bnn = BN_new();
80          if (bnn == NULL) {
81              BN_free(bne);
82              return -1;
83          }
84
85          rsa = RSA_new();
86          if (rsa == NULL) {
87              BN_free(bnn);
88              BN_free(bne);
89              return -1;
90          }
91
92          rsa->e = bne;
93          rsa->n = bnn;
94
95          /* 设置模数 */
96          BN_set_word(bne, e);
97          if (argc == 3) {
98              BN_hex2bn(&bnn, argv[2]);
99          }
100         else {
101             BN_hex2bn(&bnn, n);
102         }
103
104         PEM_write_RSAPublicKey(stdout, rsa);
105
106         RSA_free(rsa);
107     }
108     else {
109         return -1;
110     }
111
112     return 0;
113 }
```

3. 密钥生成示例

```
1  #include <stdio.h>
2  #include <string.h>
3  #include <stdlib.h>
4  #include <openssl/bn.h>
5  #include <openssl/rsa.h>
6  #include <openssl/pem.h>
7
8  /******************************************************************
9   * RSA密钥生成函数
10  *
```

```c
 * file: test_rsa_genkey.c
 * gcc -Wall -O2 -o test_rsa_genkey test_rsa_genkey.c -lcrypto
 *
 * author: tonglulin@gmail.com by www.qmailer.net
 ************************************************************************/
int main(int argc, char *argv[])
{
    /* 产生RSA密钥 */
    RSA *rsa = RSA_generate_key(1024, 65537, NULL, NULL);

    printf("BIGNUM: %s\n", BN_bn2hex(rsa->n));

    /* 提取私钥 */
    printf("PRIKEY:\n");
    PEM_write_RSAPrivateKey(stdout, rsa, NULL, NULL, 0, NULL, NULL);

    /* 提取公钥 */
    unsigned char *n_b = (unsigned char *)calloc(RSA_size(rsa), sizeof
    unsigned char *e_b = (unsigned char *)calloc(RSA_size(rsa), sizeof

    int n_size = BN_bn2bin(rsa->n, n_b);
    int b_size = BN_bn2bin(rsa->e, e_b);

    RSA *pubrsa = RSA_new();
    pubrsa->n = BN_bin2bn(n_b, n_size, NULL);
    pubrsa->e = BN_bin2bn(e_b, b_size, NULL);

    printf("PUBKEY: \n");
    PEM_write_RSAPublicKey(stdout, pubrsa);

    RSA_free(rsa);
    RSA_free(pubrsa);

    return 0;
}
```

归类：C/OOC编程, 编程开发 ，标签：PEM, RSA, RSAPublicKey, 公钥, 私钥

文章导航

上一篇：OpenSSL编程-非对称加密及RSA简介

下一篇：OpenSSL编程-Python实现公钥RSAPublicKey格式与大数字符串转换

发表评论

称呼：请填写 邮箱：请填写

链接：可以不填

写完 提交