LiuYanYGZ

linux C语言 用openssl进行签名验签 --- 亲测 sha256 sha512

1.签名

```
#include <string.h>
#include <openssl/rsa.h>
#include <openssl/pem.h>
#include <openssl/err.h>
#include <openssl/sha.h>
#include <openssl/crypto.h>

/*
 * 参考https://blog.csdn.net/zjf535214685/article/details/82182241
 */

#define PRIVATE_KEY_PATH ("./rsaprivatekey.pem")

#define SHA_WHICH        NID_sha256
#define WHICH_DIGEST_LENGTH    SHA256_DIGEST_LENGTH


void printHex(unsigned char *md, int len)
{

    int i = 0;
    for (i = 0; i < len; i++)
    {
        printf("%02x", md[i]);
    }

    printf("\n");
}

/*读取私钥*/
RSA* ReadPrivateKey(char* p_KeyPath)
{
    FILE *fp = NULL;
    RSA  *priRsa = NULL;

    printf("PrivateKeyPath[%s] \n", p_KeyPath);

    /*  打开密钥文件 */
    if(NULL == (fp = fopen(p_KeyPath, "r")))
    {
        printf( "fopen[%s] failed \n", p_KeyPath);
        return NULL;
    }
```

```
    /*  获取私钥 */
    priRsa = PEM_read_RSAPrivateKey(fp, NULL, NULL,NULL);
    if(NULL == priRsa)
    {
        ERR_print_errors_fp(stdout);
        printf( "PEM_read_RSAPrivateKey\n");
        fclose(fp);
        return NULL;
    }
    fclose(fp);

    return priRsa;
}

int test_RSA_sign(void)
{
    char *data = "china";
    char buf[128] = {0};
    RSA *privKey = NULL;
    int nOutLen = sizeof(buf);
    int nRet = 0;

    //对数据进行sha256算法摘要
    unsigned char md[WHICH_DIGEST_LENGTH];

    SHA256((unsigned char *)data, strlen(data), md);
    printHex(md, WHICH_DIGEST_LENGTH);

    privKey = ReadPrivateKey(PRIVATE_KEY_PATH);
    if (!privKey)
    {
        ERR_print_errors_fp (stderr);
        return -1;
    }


    /* 签名 */
    nRet = RSA_sign(SHA_WHICH, md, WHICH_DIGEST_LENGTH, buf, &nOutLen, privKey);
    if(nRet != 1)
    {
        printf("RSA_sign err !!! \n");
        goto quit;
    }
    printf("RSA_sign len = %d:", nOutLen);
    printHex(buf, nOutLen);


quit:
    RSA_free(privKey);

    return 0;
}


int main(int argc, char *argv[])
{
    test_RSA_sign();
    return 0;
}
```

2. 验签

```
#include <string.h>
#include <openssl/rsa.h>
#include <openssl/pem.h>
#include <openssl/err.h>
#include <openssl/sha.h>
#include <openssl/crypto.h>

/*
 * 参考https://blog.csdn.net/zjf535214685/article/details/82182241
```

```
 */

#define PUBLIC_KEY_PATH  ("./rsapubkey.pem")

#define SHA_WHICH        NID_sha256
#define WHICH_DIGEST_LENGTH    SHA256_DIGEST_LENGTH


void printHex(unsigned char *md, int len)
{

    int i = 0;
    for (i = 0; i < len; i++)
    {
        printf("%02x", md[i]);
    }

    printf("\n");
}

/*读取公匙*/
RSA* ReadPublicKey(char* p_KeyPath)
{
    FILE *fp = NULL;
    RSA *pubRsa = NULL;

    printf("PublicKeyPath[%s]\n", p_KeyPath);

    /*  打开密钥文件 */
    if(NULL == (fp = fopen(p_KeyPath, "r")))
    {
        printf( "fopen[%s] \n", p_KeyPath);
        return NULL;
    }
    /*  获取公钥 */
    if(NULL == (pubRsa = PEM_read_RSA_PUBKEY(fp, NULL, NULL,NULL)))
    {
        printf( "PEM_read_RSAPrivateKey error\n");
        fclose(fp);
        return NULL;
    }
    fclose(fp);

    return pubRsa;
}

int test_RSA_verify(void)
{
    char *data = "china";
    char buf[128] = {
        0x06,0x62,0x0b,0xb4,0x16,0xdf,0x52,0xb9,
        0x42,0x53,0x05,0x95,0x12,0xbe,0x3e,0x4f,
        0x9e,0x4d,0xed,0x20,0xf8,0x3a,0x07,0xad,
        0xc4,0xe0,0x6d,0xb9,0xd5,0x35,0xe8,0xae,
        0xf3,0x84,0xdb,0xd5,0x33,0x6f,0x10,0x9b,
        0x47,0x8d,0x26,0x7a,0x50,0x9f,0xf9,0x57,
        0xec,0xba,0xa3,0xc1,0x50,0xae,0x47,0xbb,
        0xcb,0x6c,0x87,0x78,0x19,0xb3,0x1f,0x1f,
        0x68,0x9a,0xc2,0x9e,0xde,0x3c,0xdd,0x97,
        0x17,0x17,0xaf,0xd1,0xc9,0xfb,0x68,0x58,
        0x19,0xbb,0xa4,0xf4,0x18,0x4d,0xe3,0xf3,
        0xb0,0x8d,0x30,0xe6,0x5b,0x6d,0x5e,0x2f,
        0xf5,0xe7,0x6b,0x30,0xf0,0x70,0xa4,0x69,
        0xfa,0xb9,0xa8,0xdd,0xf0,0x71,0x99,0x6c,
        0x7a,0xc2,0xce,0xe8,0x13,0x46,0x0c,0x85,
        0x8e,0x3f,0x55,0xe3,0xe7,0x30,0xd1,0x7d,
    };


    RSA *pubKey = NULL;
    int nOutLen = sizeof(buf);
    int nRet = 0;

    //对数据进行sha256算法摘要
    unsigned char md[WHICH_DIGEST_LENGTH];
```

```
    SHA256((unsigned char *)data, strlen(data), md);
    printHex(md, WHICH_DIGEST_LENGTH);


    pubKey = ReadPublicKey(PUBLIC_KEY_PATH);
    if (!pubKey)
    {
        printf("Error: can't load public key");
        return -1;
    }

    /* 验签 */
    nRet = RSA_verify(SHA_WHICH, md, WHICH_DIGEST_LENGTH, buf, nOutLen, pubKey);
    printf("RSA_verify %s(ret=%d).\r\n", (1 == nRet) ? "Success" : "Failed", nRet);

    RSA_free(pubKey);

    return 0;
}

int main(int argc, char *argv[])
{
    test_RSA_verify();
    return 0;
}
```

```
[root@localhost PCSCTest]#
[root@localhost PCSCTest]#
[root@localhost PCSCTest]# gcc RSA_sign.c  -lcrypto ; ./a.out
50c0152c2952082aeaf427885a2f617d67cf6de183a8816c0955ea5b875a216b
PrivateKeyPath[./rsaprivatekey.pem]
RSA_sign len = 128:06620bb416df52b94253059512be3e4f9e4ded20f83a07adc4e06db9d535e8aef384dbd5336f109b478d267a509ff957ecbaa3c150ae47bbcb6c877819b3
1f1f689ac29ede3cdd971717afd1c9fb685819bba4f4184de3f3b08d30e65b6d5e2ff5e76b30f070a469fab9a8ddf071996c7ac2cee813460c858e3f55e3e730d17d
[root@localhost PCSCTest]#
[root@localhost PCSCTest]#
[root@localhost PCSCTest]#
[root@localhost PCSCTest]# gcc RSA_verify.c  -lcrypto ; ./a.out
50c0152c2952082aeaf427885a2f617d67cf6de183a8816c0955ea5b875a216b
PublicKeyPath[./rsapubkey.pem]
RSA_verify Success(ret=1).
[root@localhost PCSCTest]#
```

标签: 加密

LiuYanYGZ
关注 - 0
粉丝 - 44
+加关注

0 推荐 0 反对

« 上一篇: photoshop调整图片的 色相/饱和度
» 下一篇: linux C语言 用openssl进行签名验签 --- 亲测2 sha256 sha512

posted @ 2020-03-21 17:47 LiuYanYGZ 阅读(1847) 评论(0) 编辑 收藏 举报

编辑推荐：
· 聊聊我在微软外服的工作经历及一些个人见解
· 死磕 NIO — Reactor 模式就一定意味着高性能吗？
· 消息队列那么多，为什么建议深入了解下RabbitMQ？
· 技术管理进阶——管人还是管事？
· 以终为始：如何让你的开发符合预期

最新新闻：
· LSTM一败涂地！男生发表4页最离谱论文，用时序模型预测女友情绪(2021-10-25 17:09)
· AI杀手终成现实？美国陆军「杀人机器狗」引发恐慌(2021-10-25 17:02)
· AI学会灌水和造假！Google新研究揭露了AI现实应用的陷阱(2021-10-25 16:50)
· 可口可乐宣布推出由100%植物性塑料制成的瓶子(2021-10-25 16:40)
· 抢先Win11！华为移动应用引擎第二批众测开启：在PC上玩安卓App(2021-10-25 16:30)
» 更多新闻...

历史上的今天：
2018-03-21 选择Netty的理由