openssl AES加密算法API的使用示例

openssl为用户提供了丰富的指令，同时也提供了供编程调用的API，本文以使用128位aes算法的ecb模式进行加密和解密验证，如下所示

第一种方法，直接使用aes算法提供的api进行调用，代码如下

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <openssl/aes.h>

int main(void)
{
    char userkey[AES_BLOCK_SIZE];
    unsigned char *date = malloc(AES_BLOCK_SIZE*3);
    unsigned char *encrypt = malloc(AES_BLOCK_SIZE*3 + 4);
    unsigned char *plain = malloc(AES_BLOCK_SIZE*3);
    AES_KEY key;

    memset((void*)userkey, 'k', AES_BLOCK_SIZE);
    memset((void*)date, 'p', AES_BLOCK_SIZE*3);
    memset((void*)encrypt, 0, AES_BLOCK_SIZE*6);
    memset((void*)plain, 0, AES_BLOCK_SIZE*3);

    /*设置加密key及密钥长度*/
    AES_set_encrypt_key(userkey, AES_BLOCK_SIZE*8, &key);

    int len = 0;
    /*循环加密，每次只能加密AES_BLOCK_SIZE长度的数据*/
while(len < AES_BLOCK_SIZE*3) {
        AES_encrypt(date+len, encrypt+len, &key);
        len += AES_BLOCK_SIZE;
    }
    /*设置解密key及密钥长度*/
    AES_set_decrypt_key(userkey, AES_BLOCK_SIZE*8, &key);

    len = 0;
    /*循环解密*/
while(len < AES_BLOCK_SIZE*3) {
        AES_decrypt(encrypt+len, plain+len, &key);
        len += AES_BLOCK_SIZE;
    }
    /*解密后与原数据是否一致*/
    if(!memcmp(plain, date, AES_BLOCK_SIZE*3)){
        printf("test success\n");
    }else{
        printf("test failed\n");
    }

    printf("encrypt: ");
    int i = 0;
    for(i = 0; i < AES_BLOCK_SIZE*3 + 4; i++){
        printf("%.2x ", encrypt[i]);
        if((i+1) % 32 == 0){
            printf("\n");
        }
    }
    printf("\n");

    return 0;
}
```


访问人数：
UV
访问总量：
PV
昵称： Gordon0918
园龄： 6年8个月
粉丝： 21
关注： 1
+加关注

< 2020年4月 >

日	一	二	三	四	五	六
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2
3	4	5	6	7	8	9

搜索

找找看
谷歌搜索

常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签

我的标签

android(6)
逆向(4)
Hook(2)
ida(2)
genrsa(2)
RSA(2)
smali(2)
so(1)
sphinx(1)
substrate(1)
更多

随笔分类

android(6)
android安全(11)
C/C++(2)
git(1)
Linux(3)
openssl(8)
Scracpy(2)
Windows(1)
渗透测试
协议(2)

编译执行结果如下

```
xlzh@cmos:~/cmos/openssl-code/aes$ gcc aes.c -o aes.out -lssl -lcrypto
xlzh@cmos:~/cmos/openssl-code/aes$ ./aes.out
test success
encrypt: 08 a9 74 4d b0 66 57 1b 57 fe 60 3d 91 e4 ed 53 08 a9 74 4d b0 66 57 1b 57 fe 60
3d 91 e4 ed 53
08 a9 74 4d b0 66 57 1b 57 fe 60 3d 91 e4 ed 53 00 00 00 00
xlzh@cmos:~/cmos/openssl-code/aes$
```

第二种方法，使用EVP框架，示例如下

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <openssl/evp.h>
#include <openssl/aes.h>

int main(void)
{
    char userkey[EVP_MAX_KEY_LENGTH];
    char iv[EVP_MAX_IV_LENGTH];
    unsigned char *date = malloc(AES_BLOCK_SIZE*3);
    unsigned char *encrypt = malloc(AES_BLOCK_SIZE*6);
    unsigned char *plain = malloc(AES_BLOCK_SIZE*6);
    EVP_CIPHER_CTX ctx;
    int ret;
    int tlen = 0;
    int mlen = 0;
    int flen = 0;

    memset((void*)userkey, 'k', EVP_MAX_KEY_LENGTH);
    memset((void*)iv, 'i', EVP_MAX_IV_LENGTH);
    memset((void*)date, 'p', AES_BLOCK_SIZE*3);
    memset((void*)encrypt, 0, AES_BLOCK_SIZE*6);
    memset((void*)plain, 0, AES_BLOCK_SIZE*6);

    /*初始化ctx*/
    EVP_CIPHER_CTX_init(&ctx);

    /*指定加密算法及key和iv(此处IV没有用)*/
    ret = EVP_EncryptInit_ex(&ctx, EVP_aes_128_ecb(), NULL, userkey, iv);
    if(ret != 1) {
        printf("EVP_EncryptInit_ex failed\n");
        exit(-1);
    }

    /*禁用padding功能*/
    EVP_CIPHER_CTX_set_padding(&ctx, 0);
    /*进行加密操作*/
    ret = EVP_EncryptUpdate(&ctx, encrypt, &mlen, date, AES_BLOCK_SIZE*3);
    if(ret != 1) {
        printf("EVP_EncryptUpdate failed\n");
        exit(-1);
    }
    /*结束加密操作*/
    ret = EVP_EncryptFinal_ex(&ctx, encrypt+mlen, &flen);
    if(ret != 1) {
        printf("EVP_EncryptFinal_ex failed\n");
        exit(-1);
    }

    tlen = mlen + flen;

    tlen = 0;
    mlen = 0;
    flen = 0;

    EVP_CIPHER_CTX_cleanup(&ctx);
    EVP_CIPHER_CTX_init(&ctx);

    ret = EVP_DecryptInit_ex(&ctx, EVP_aes_128_ecb(), NULL, userkey, iv);
    if(ret != 1) {
        printf("EVP_DecryptInit_ex failed\n");
        exit(-1);
```


随笔档案

2017年12月(1)
2017年4月(6)
2017年3月(4)
2016年6月(4)
2016年5月(1)
2016年4月(5)
2016年3月(6)
2016年1月(5)
2015年7月(1)
2015年1月(3)
2014年7月(1)

最新评论

1. Re:openssl 对称加密算法enc命令详解
fedora 29 x86 workstation OpenSSL 1.1.1d FIPS 10 Sep 2019 没有 aes-256-gcm. openssl enc -ciphers 何解...
--NickD
2. Re:openssl 对称加密算法enc命令详解
-pass env:passwd 的passwd的前面不需要加$?
--creazyloser
3. Re:Android AccessibilityService(辅助服务) 使用示例
他是返回的整个activity 的view ，所以会包含三个Fragment 的
--伍歌歌
4. Re:PPTP协议握手流程分析
大佬 自己能用软件模拟vpn 并建立通道 进行数据传输吗
--54辉哥
5. Re:Https协议简析及中间人攻击原理
写的不错
--aqu415

阅读排行榜

1. openssl 对称加密算法enc命令详解(25875)
2. openssl 证书请求和自签名命令req详解(23152)
3. 如何把java代码转换成smali代码(20907)
4. openssl 非对称加密算法RSA命令详解(18579)
5. openssl 摘要和签名验证指令dgst使用详解(18023)

评论排行榜

1. 如何把java代码转换成smali代码(4)
2. android调试系列--使用ida pro调试原生程序(3)
3. openssl 对称加密算法enc命令详解(2)
4. openssl 非对称加密算法RSA命令详解(1)
5. openssl 非对称加密算法DSA命令详解(1)

推荐排行榜

1. openssl 证书请求和自签名命令req详解(5)
2. Android调试系列—使用android studio调试smali代码(3)
3. openssl 非对称加密算法RSA命令详解(2)
4. openssl CA服务器模拟指令CA详解(1)
5. Https协议简析及中间人攻击原理(1)

```
    }

    EVP_CIPHER_CTX_set_padding(&ctx, 0);
    ret = EVP_DecryptUpdate(&ctx, plain, &mlen, encrypt, AES_BLOCK_SIZE*3);
    if(ret != 1) {
        printf("EVP_DecryptUpdate failed\n");
        exit(-1);
    }

    ret = EVP_DecryptFinal_ex(&ctx, plain+mlen, &flen);
    if(ret != 1) {
        printf("EVP_DecryptFinal_ex failed\n");
        exit(-1);
    }
    /*对比解密后与原数据是否一致*/
    if(!memcmp(plain, date, AES_BLOCK_SIZE*3)) {
        printf("test success\n");
    } else {
        printf("test failed\n");
    }

    printf("encrypt: ");
    int i;
    for(i = 0; i < AES_BLOCK_SIZE*3+4; i ++){
        printf("%.2x ", encrypt[i]);
        if((i+1)%32 == 0){
            printf("\n");
        }
    }
    printf("\n");

    return 0;
}
```

编译执行结果如下：

```
xlzh@cmos:~/cmos/openssl-code/aes$ gcc evp.c -o evp.out -lssl -lcrypto
xlzh@cmos:~/cmos/openssl-code/aes$ ./evp.out
test success
encrypt: 08 a9 74 4d b0 66 57 1b 57 fe 60 3d 91 e4 ed 53 08 a9 74 4d b0 66 57 1b 57 fe 60
3d 91 e4 ed 53
08 a9 74 4d b0 66 57 1b 57 fe 60 3d 91 e4 ed 53 00 00 00 00
xlzh@cmos:~/cmos/openssl-code/aes$
```

EVP框架是对openssl提供的所有算法进行了封装，在使用工程中只需要修改少量的代码就可以选择不同的加密算法，在工作中通常采用这种方式。

在上述两个示例中，直接使用API提供的接口，没有使用padding，在EVP中同样需要声明不可以使用padding方式，否则即使要加密的数据长度是AES_BLOCK_SIZE的整数倍，EVP默认也会对原始数据进行追加，导致结果不同，所以在试验中通过EVP_CIPHER_CTX_set_padding(&ctx, 0)函数关闭的EVP的padding功能，同样在解密的时候也需要进行关闭。

分类: openssl

标签: openssl 加密 aes cbc EVP API

Gordon0918
关注 - 1
粉丝 - 21
+加关注

0 0

« 上一篇： Linux能力(capability)机制的继承
» 下一篇： openssl 非对称加密算法RSA命令详解

posted @ 2016-03-29 14:26 Gordon0918 阅读(14975) 评论(0) 编辑 收藏

相关博文：
· openssl evp 对称加密(AES_ecb,ccb)
· AES加密算法C++实现
· OpenSSL中AES加密的用法
· Openssl aes加解密例程 更进一步
· 使用openssl库实现RSA、AES数据加密
» 更多推荐...

最新 IT 新闻:
· 新MacBook Air评测：性价比提升，同价位最值得买的苹果笔记本
· 新冠疫情影响美科技产业 初创企业3月累计裁员近4000人
· 受新冠病毒疫情影响 施乐宣布放弃对惠普“蛇吞象”式敌意收购
· 文件显示特斯拉拖着不愿关厂 卫生官员称是“公共健康风险”
· 传iPhone 9或4月22日开卖 代码曝光具备车钥匙功能
» 更多新闻...