CVE-2018-12794类型混淆漏洞详细分析

kqsdy / 2019-03-07 08:35:00 / 浏览数 1549 安全技术 漏洞分析

漏洞概述

通过构造数据可以使程序在处理Template对象时使用Form对象的函数进行处理，从而造成越界数据读取，该漏洞为类型混淆型漏洞。通过构建XML数据包（XDP）模板并对

漏洞细节

POC代码分析
分析漏洞的POC文件，通过PDF流解析工具PdfStreamDumper可以看到pdf文件里面的objects流。其中第5个object流为XML Data Package结构。
XML Data Package（XDP）是Adobe Systems创建的XML 文件格式。该格式允许将PDF内容或Adobe XML Forms Architecture（XFA）资源打包在XML容器中。XDP符合XML
1.0的规范，可以作为独立文档，也可以在PDF文档中携带。XDP提供了一种在XML容器中打包表单组件的机制，XDP还可以打包PDF文件以及XML表单和模板数据。

```
10 Objects
1 HLen: 0x179
2 HLen: 0x32
3 HLen: 0xC0
4 HLen: 0x21
5 0x326-0x50C
6 HLen: 0x1029
7 HLen: 0x128D
8 HLen: 0x12B1
9 HLen: 0x12B2
0 HLen: 0x11A
```

```
      <pdf>
        <interactive>
          1
        </interactive>
        <scriptModel>
          XFA
        </scriptModel>
        <encryption>
          <permissions>
          </permissions>
        </encryption>
      </pdf>
    </present>
    <acrobat>
      <acrobat7>
        <dynamicRender>
          required
        </dynamicRender>
      </acrobat7>
    </acrobat>
  </config>
```

```
<template>

  <subform layout='tb' name='outerform'>

        <pageSet>
             <pageArea id='Page2' name='Page2'>
                  <contentArea h='100mm' w='200mm' x='0.25in' y='0.25in'/>
                  <medium long='297mm' short='210mm' stock='a4'/>
             </pageArea>
        </pageSet>


        <subform name="sub1"></subform>

        <subform name="sub2">
             <calculate>
                  <script contentType="application/x-javascript">
                       app.alert("crash...!");
                  </script>
             </calculate>
        </subform>

   </subform>

</template>
```

而在第1个object流对象里面的XFA（XML Forms
Architecture）对象会执行JavaScript代码，该代码会操作sub1和sub2，先将sub1添加为xfa.template对象，sub2添加为xfa.from对象，然后将sub2附加到sub1。
XFA为XML Forms Architecture，是一系列专有
XML规范，用于增强Web表单的处理。XFA提供基于模板的语法和处理规则集，允许用户构建交互式表单。基于模板的语法将定义用户在其中提供数据的字段，XFA的开放特性将提供描述交互式表单的通用 XML 语法。
最后执行JavaScript代码将o2的presence属性设置为inactive ，该属性的含义为隐藏对象并将其从事件处理中排除。在执行该操作的时候将触发crash。

漏洞调试

设置windbg为默认调试器，运行POC文件，windbg将暂停到发生crach的地方。

```
Executable search path is:
ModLoad: 009a0000 00bd0000   C:\Program Files\Adobe\Acrobat Reader DC\Reader\AcroRd32.exe
(e8c.e88): Access violation - code c0000005 (!!! second chance !!!)
eax=0062006f ebx=00000000 ecx=0062006f edx=002bc63c esi=00000000 edi=0706c8b0
eip=5c667421 esp=002bc4a4 ebp=002bc4a4 iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00210206
*** WARNING: Unable to verify checksum for C:\Program Files\Adobe\Acrobat Reader DC\Reader\plug_ins\AcroForm.api
*** ERROR: Symbol file could not be found.  Defaulted to export symbols for C:\Program Files\Adobe\Acrobat Reader DC\Reader\plug_ins\AcroForm.api
AcroForm!PlugInMain+0x27081:
5c667421 83790400        cmp     dword ptr [ecx+4],0  ds:0023:00620073=????????
0:000> kb L3
ChildEBP RetAddr  Args to Child
WARNING: Stack unwind information not available. Following frames may be wrong.
002bc4a4 5c6d7d91 5ccb41a8 0706c8b0 05a8ba08 AcroForm!PlugInMain+0x27081
002bc50c 5c6cdd14 05a8ba08 01000001 002bc63c AcroForm!PlugInMain+0x979f1
002bc5a8 5c68cc57 00000001 002bc63c 07325178 AcroForm!PlugInMain+0x8d974
0:000> ub AcroForm!PlugInMain+0x979f1
AcroForm!PlugInMain+0x979d3:
5c6d7d73 8bf9            mov     edi,ecx
5c6d7d75 33f6            xor     esi,esi
5c6d7d77 39b7d0010000    cmp     dword ptr [edi+1D0h],esi
5c6d7d7d 761f            jbe     AcroForm!PlugInMain+0x979fe (5c6d7d9e)
5c6d7d7f 8b87d4010000    mov     eax,dword ptr [edi+1D4h]
5c6d7d85 ff74240c        push    dword ptr [esp+0Ch]
5c6d7d89 8d0cf0          lea     ecx,[eax+esi*8]
5c6d7d8c e889f6f8ff      call    AcroForm!PlugInMain+0x2707a (5c66741a)
0706ca00  6f 00 3d 00 41 00 64 00-6f 00 62 00 65 00 20 00  o.=.A.d.o.b.e. .
0706ca10  53 00 79 00 73 00 74 00-65 00 6d 00 73 00 20 00  S.y.s.t.e.m.s. .
0706ca20  49 00 6e 00 63 00 6f 00-72 00 70 00 6f 00 72 00  I.n.c.o.r.p.o.r.
0706ca30  61 00 74 00 65 00 64 00-2c 00 20 00 63 00 3d 00  a.t.e.d.,. .c.=.
0706ca40  55 00 53 00 00 00 00 00-6d 1d fb 04 5c bd 1c 0e  U.S.....m...\...
0706ca50  43 00 3a 00 5c 00 50 00-72 00 6f 00 67 00 72 00  C.:.\.P.r.o.g.r.
0706ca60  61 00 6d 00 20 00 46 00-69 00 6c 00 65 00 73 00  a.m. .F.i.l.e.s.
0706ca70  5c 00 41 00 64 00 6f 00-62 00 65 00 5c 00 41 00  \.A.d.o.b.e.\.A.
0706ca80  63 00 72 00 6f 00 62 00                          c.r.o.b.
0:000> r
eax=0062006f ebx=00000000 ecx=0062006f edx=002bc63c esi=00000000 edi=0706c8b0
eip=5c667421 esp=002bc4a4 ebp=002bc4a4 iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00210206
AcroForm!PlugInMain+0x27081:
5c667421 83790400        cmp     dword ptr [ecx+4],0  ds:0023:00620073=????????
068e5de0  9e 8f 7b 03 83 d6 14 0a-00 04 10 96 00 00 01 36  ..{............6
068e5df0  01 00 00 07 00 36 00 7e-01 ff 02 20 02 59 02 ad  .....6.~... .Y..
068e5e00  03 23 03 4f 03 6f 03 75-03 7e 03 8a 03 8c 03 a1  .#.O.o.u.~......
068e5e10  03 ce 03 f6 04 0d 04 4f-05 13 05 1d 05 c7 05 ea  .......O........
068e5e20  05 f4 06 03 06 15 06 1b-06 1f 06 3a 06 5e 06 ff  ...........:.^..
068e5e30  07 6d 1d 6a 1d 6b 1d c3-1d ca 1d ff 1e 7f 1e 9b  .m.j.k..........
068e5e40  1e 9e 1e f1 1e f9 1f 15-1f 1d 1f 45 1f 4d 1f 57  ...........E.M.W
068e5e50  1f 59 1f 5b 1f 5d 1f 7d-1f b4 1f c4 1f d3 1f db  .Y.[.].}........
068e5e60  1f ef 1f f4 1f fe 20 0f-20 22 20 26 20 30 20 34  ...... . " & 0 4
068e5e70  20 3a 20 3c 20 3e 20 44                          : < > D
0:000> r
eax=44203e20 ebx=00000000 ecx=44203e20 edx=001fc340 esi=00000000 edi=068e5ca0
eip=5af37421 esp=001fc1a8 ebp=001fc1a8 iopl=0         nv up ei pl nz na po nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00210202
AcroForm!PlugInMain+0x27081:
5af37421 83790400        cmp     dword ptr [ecx+4],0  ds:0023:44203e24=????????
```

可以发现程序异常在AcroForm.api模块，ecx

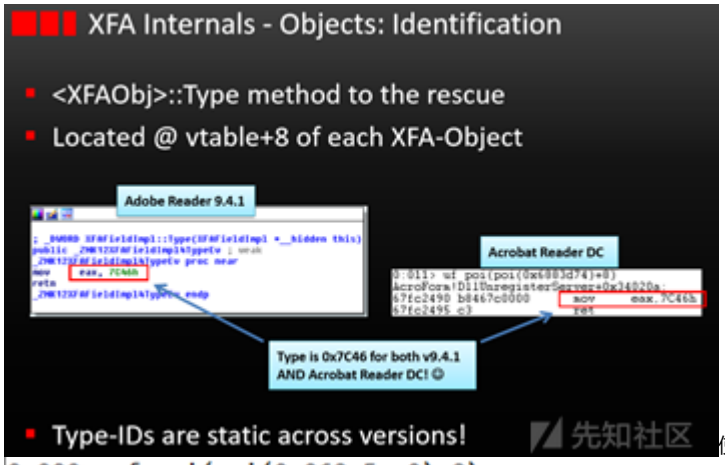
通过反汇编发现ecx的值为[eax+esi*8]，而esi只是一个偏移并且为0，所以ecx的值受e

当重新运行程序，可以发现[edi+1d4]每次的值都差别很大，说明该地址的值是未知区

在通过heap命令来看该区域堆栈的分配情况可以发现edi所在的空间大小为0x140，edi

```
068e5140: 00298 . 00518 [101] - busy (510)
068e5658: 00518 . 00220 [101] - busy (211)
068e5878: 00220 . 00420 [101] - busy (411)
068e5c98: 00420 . 00148 [101] - busy (140)
068e5de0: 00148 . 010a0 [101] - busy (1096)
068e6e80: 010a0 . 09818 [101] - busy (9810)
068f0698: 09818 . 00820 [101] - busy (811)
068f0eb8: 00820 . 02660 [101] - busy (2654)
068f3518: 02660 . 00410 [101] - busy (408)
068f3928: 00410 . 00298 [101] - busy (290)
068f3bc0: 00298 . 00600 [101] - busy (5f4)
```

根据后面的一些指针操作可以发现该堆块保存的都是一些地址，猜测该堆块应该保存的是一个对象。从poi(poi(对象地址)+8)的命令可以显示出Type-IDs。



使用uf poi(poi(0x068e5ca0)+8)命令，可以看到类型为0x7C00，说明了该堆块保存的是一

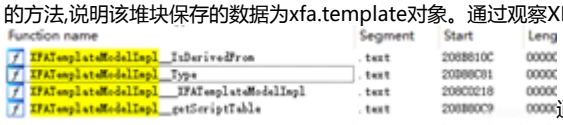
```
0:000> uf poi(poi(0x068e5ca0)+8)
AcroForm!DllUnregisterServer+0x3791fe:
5b46041b b8007c0000      mov     eax,7C00h
5b460420 c3              ret
```

该文件官方没有pdb文件，在调试的时候很多地方很难定位到具体的结构，不过在网上能找

```
XFATemplateModelImpl__Type proc near
mov     eax, 7C00h
retn
XFATemplateModelImpl__Type endp
```

可以看到0x7C00表示的是XFATemplateModelImpl::Type

的方法,说明该堆块保存的数据为xfa.template对象。通过观察XFATemplateModelImpl相关的有如下函数，其中没有发现有关对象创建和内存申请的函数。



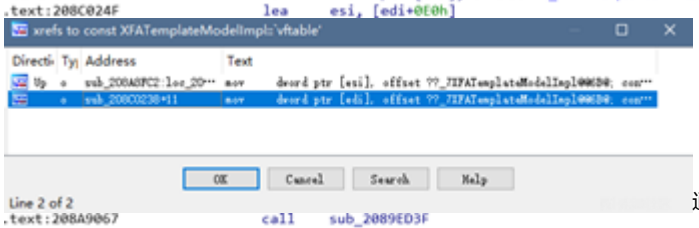
通过交叉引用或直接搜索XFATemplateModelImpl可以找到XFATemplateModelImpl类的构造函数XFATem

```
.rdata:20FED7CC ; const XFATemplateModelImpl::`vftable'
.rdata:20FED7CC ??_7XFATemplateModelImpl@@6B@ dd offset XFATemplateModelImpl___XFATemplateModelImpl
.rdata:20FED7CC                                         ; DATA XREF: sub_208A8FC2:loc_208A9081↑o
.rdata:20FED7CC                                         ; sub_208C0238+11↑o
.rdata:20FED7D0                 dd offset XFATemplateModelImpl__IsDerivedFrom
.rdata:20FED7D4                 dd offset XFATemplateModelImpl__Type
```

因为虚表是由构造函数进行初始化，可以通过

```
.text:208C0238 sub_208C0238    proc near               ; CODE XREF: XFATemplateModelImpl___XFATemplateModelImpl+3↑p
.text:208C0238
.text:208C0238 var_10          = dword ptr -10h
.text:208C0238 var_4           = dword ptr -4
.text:208C0238 arg_4           = dword ptr  0Ch
.text:208C0238
.text:208C0238 ; FUNCTION CHUNK AT .text:20E348AB SIZE 000000BD BYTES
.text:208C0238
.text:208C0238 ; __unwind { // loc_20E3494D
.text:208C0238                 push    4
.text:208C023A                 mov     eax, offset loc_20E3494D
.text:208C023F                 call    __EH_prolog3
.text:208C0244                 mov     edi, ecx
.text:208C0246                 mov     [ebp+var_10], edi
.text:208C0249                 mov     dword ptr [edi], offset ??_7XFATemplateModelImpl@@6B@ ; const XFATemplateModelImpl::`vftable'
.text:208C024F                 lea     esi, [edi+0E0h]
```

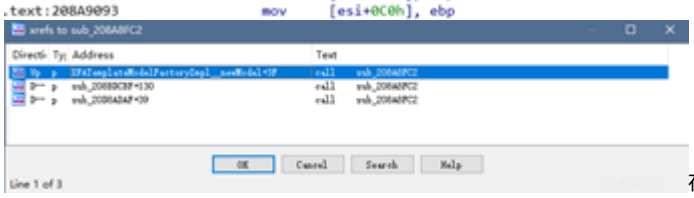
通过对该虚表指针进行交叉引用可以得到两处



通过交叉引用可以找到另一处引用函数sub_208A8FC2。

```
.text:208A9067                 call    sub_2089ED3F
.text:208A906C                 cmp     [esp+30h+var_10], 0
.text:208A9071                 mov     byte ptr [esp+30h+var_4], 4
.text:208A9076                 jz      short loc_208A9081
.text:208A9078                 lea     ecx, [esp+30h+var_14]
.text:208A907C                 call    sub_2086EA7F
.text:208A9081
.text:208A9081 loc_208A9081:                           ; CODE XREF: sub_208A8FC2+B4↑j
.text:208A9081                 mov     dword ptr [esi], offset ??_7XFATemplateModelImpl@@6B@ ; const XFATemplateModelImpl::`vftable'
.text:208A9087                 mov     [esi+088h], ebp
.text:208A908D                 mov     [esi+0BCh], ebp
.text:208A9093                 mov     [esi+0C0h], ebp
```

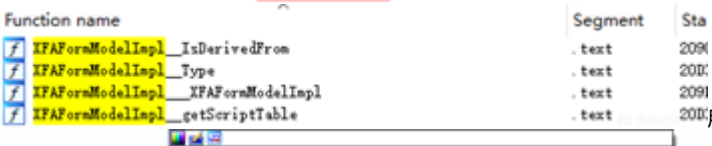
再对sub_208A8FC2进行交叉引用可以找到在



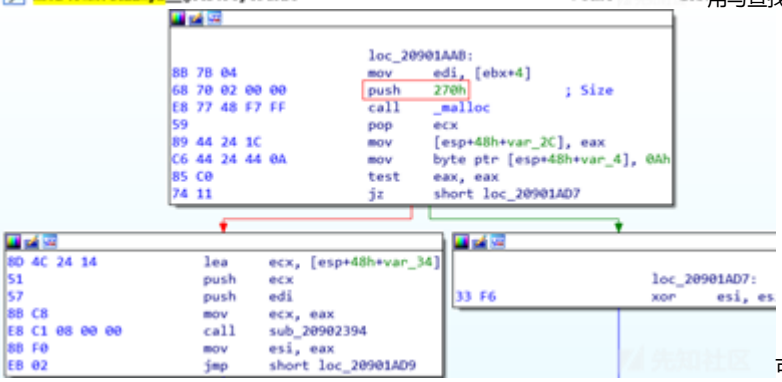
在XFATemplateModelFactoryImpl::newModel函数中可以看到在函数开始就申请了0x140字

```
.text:208A8F55 XFATemplateModelFactoryImpl__newModel proc near
.text:208A8F55                                         ; DATA XREF: .rdata:20FEEB80↓o
.text:208A8F55
.text:208A8F55 var_10          = dword ptr -10h
.text:208A8F55 var_4           = dword ptr -4
.text:208A8F55 arg_0           = dword ptr  8
.text:208A8F55 arg_4           = dword ptr  0Ch
.text:208A8F55
.text:208A8F55 ; FUNCTION CHUNK AT .text:20E31D63 SIZE 0000000A BYTES
.text:208A8F55 ; FUNCTION CHUNK AT .text:20E31D6D SIZE 0000001B BYTES
.text:208A8F55
.text:208A8F55 ; __unwind { // loc_20E31D6D
.text:208A8F55                 push    8
.text:208A8F57                 mov     eax, offset loc_20E31D6D
.text:208A8F5C                 call    __EH_prolog3
.text:208A8F61                 mov     edi, ecx
.text:208A8F63                 push    140h            ; Size
.text:208A8F68                 call    _malloc
```

Template对应的类是XFATemplateModelImpl，推测Form对应的类为XFAFormModelIm



用与查找XFATemplateModelImpl类的办法可以找到XFAFormModelImpl的虚表。在对虚



可以看到在创建Form对象的时候申请的空间大小为0x270，说明Form对象的大小为

参考资料

https://github.com/siberas/arpwn
SyScan3602016-_Pwning_Adobe_Reader_with_XFA.pdf
http://blogs.adobe.com/formfeed/2009/03/xfa_30_presenceinactive.html

上一篇：某decms v5.7 sp2 后... 下一篇：MIME嗅探，Encoding嗅探...

1. 1 条回复

fresh 2019-03-07 10:41:18

优秀呀

0 回复Ta

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板