Talos Vulnerability Report

TALOS-2020-0997

Nitro PRO PDF nested pages remote code execution vulnerability

MAY 18, 2020

CVE NUMBER

CVE-2020-6074

Summary

An exploitable code execution vulnerability exists in the PDF parser of Nitro Pro 13.9.1.155. A specially crafted PDF document can cause a use-after-free which can lead to remote code execution. An attacker can provide a malicious file to trigger this vulnerability.

Tested Versions

Nitro Pro 13.9.1.155

Product URLs

https://www.gonitro.com/

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

Nitro PDF allows users to save, read, sign and edit PDF files on their machines.

The modules analyzed for this vulnerability are:

```
Mapped memory image file: C:\Program Files\Nitro\Pro\13\NitroPDF.exe
Image path: C:\Program Files\Nitro\Pro\13\NitroPDF.exe
Image name: NitroPDF.exe
Browse all global symbols  functions  data
Timestamp:        Thu Dec 19 05:46:43 2019 (5DFB6323)
CheckSum:         00932421
ImageSize:        00934000
File version:     13.9.1.155
Product version:  13.9.1.155
```

During the parsing of a PDF, a buffer is allocated for each page in the PDF. These pages are stored in a larger PDF object.

```
NitroPDF+0x98a5a
        curr_page_number = 0;
        result = PDDocGetNumPages(v2);
        num_pages = result;
        if ( !result )
        {
            num_pages = 1;
            v3 = 1;
        }
        if ( num_pages > 0 )
        {
            do
            {
                v7 = operator new(0xB8ui64);
                memset(v7, 0, 0xB8ui64);
                *((_DWORD *)v7 + 0x27) = curr_page_number;
                *((_BYTE *)v7 + 0x98) = v3;
                sub_98850(v1, (__int64)v7, curr_page_number, 0);

                // Saving the allocated page in the larger PDF object
                result = (*(__int64 (__fastcall **)(__int64, void **))(*(_QWORD *)(v1 + 0x5A8) + 72i64))(v1 + 0x5A8, &v7);
                ++curr_page_number;
            } while ( curr_page_number < num_pages );
        }
```

In the case of nested pages, it is possible to free this page in the overarching PDF object.

```
NitroPDF+0x9c13d
    if ( (_DWORD)curr_pages == 1 )
    {
        page_struct = (_BYTE *)sub_96E30(v10, 0i64, 1u);
        if ( page_struct )
        {
        if ( page_struct[0x98] )
        {
            LODWORD(curr_pages) = 0;

            // Freeing the original page allocated for the PDF
            ((void (__cdecl *)(void *, int))j_j_free)(page_struct, 0xB8);
            *(_QWORD *)(v10 + 0x5B8) = *(_QWORD *)(v10 + 0x5B0);
        }
        }
    }
```

With this page freed, a crafted PDF can cause an exception to occur. One way is to claim to have more pages in the PDF than given via the /Kids tag. An example of what causes this exception is below:

```
3 0 obj
<<
  /Count 2
  /Kids [ ]
>>
```

During the handling of this exception, a set of cleanup code is called for the main PDF object. In this code, various buffers are cleared, including the original buffer allocation for our page in the PDF.

```
NitroPDF+3154b6

.text:00000000003154B6                mov     [rsp+38h+var_28], rdx
.text:00000000003154BB                push    rbp
.text:00000000003154BC                sub     rsp, 30h
.text:00000000003154C0                mov     rbp, rdx
.text:00000000003154C3                mov     rax, [rbp+78h]
.text:00000000003154C7                mov     qword ptr [rax], 0
.text:00000000003154CE                mov     byte ptr [rax+98h], 1
.text:00000000003154D5                lea     rax, unk_988FB
.text:00000000003154DC                add     rsp, 30h
.text:00000000003154E0                pop     rbp
.text:00000000003154E1                retn
```

While the page was freed, it was never cleared from the original PDF object. A carefully crafted PDF could possibly setup the heap in a way to write an arbitrary null byte out of bounds which can lead to further memory corruption and possibly arbitrary code execution.

Timeline

2020-02-17 - Vendor Disclosure
2020-05-18 - Public Release

CREDIT

Discovered by Cory Duplantis of Cisco Talos.
