Talos Vulnerability Report

TALOS-2020-1068

Nitro Pro XRefTable Entry Missing Object Code Execution Vulnerability

SEPTEMBER 15, 2020

CVE NUMBER

CVE-2020-6115

SUMMARY

An exploitable vulnerability exists in the cross-reference table repairing functionality of Nitro Software, Inc.'s Nitro Pro 13.13.2.242. While searching for an object identifier in a malformed document that is missing from the cross-reference table, the application will save a reference to the object's cross-reference table entry inside a stack variable. If the referenced object identifier is not found, the application may resize the cross-reference table which can change the scope of its entry. Later when the application tries to reference cross-reference entry via the stack variable, the application will access memory belonging to the recently freed table causing a use-after-free condition. A specially crafted document can be delivered by an attacker and loaded by a victim in order to trigger this vulnerability.

CONFIRMED VULNERABLE VERSIONS

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

Nitro Pro 13.13.2.242
Nitro Pro 13.16.2.300

PRODUCT URLS

Nitro Pro - https://www.gonitro.com/nps/product-details/downloads

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 Software, Inc. includes their flagship product, Nitro Pro as part of their Nitro Productivity Suite. Nitro Pro is Nitro Software's PDF editor and flagship product. This product allows users to create and modify documents that follow the Portable Document Format (PDF) specification and other digital documents.

Nitro Software Inc. develops commercial software used to create, edit, sign, and secure Portable Document Format files and digital documents. This is supported by their Nitro Pro application as part of their Nitro Productivity Suite. The Nitro Pro application allows users to read, modify, and create documents that follow the Portable Document Format standard and all of its capabilities. At the end of each PDF document is what is known as the trailer which contains a file offset that points to a table that is known as the cross-reference table. This cross-reference table contains a list of items representing each object within the document. Each item describes a slot with information such as whether the object is being used or is free for use, and also includes the file offset for where the object is located. The cross-reference table is required by the PDF file format for the application to locate the different objects associated with a document so that it may be read, or modified. If this table is corrupt or malformed in some way, many PDF readers will go through a process of repairing or rebuilding the cross-reference table so that the application may properly read each object associated with the document.

When first opening the file, the following code will be executed. This will first open the file at [1], in order to create a file stream at [2]. After the file stream has been created, at [3] the application will begin to parse the document's header.

```
npdf!CosDosWriteToStream+0xa7b:
5a68fbdb 51              push    ecx
5a68fbdc 6a01            push    1
5a68fbde 50              push    eax
5a68fbdf e80c3afdff      call    npdf!ASFileSysOpenFile (5a6635f0)         // [1] open the file
5a68fbe4 83c40c          add     esp,0Ch
5a68fbe7 85c0            test    eax,eax
5a68fbe9 7435            jne     npdf!CosDosWriteToStream+0xac0 (5a68fc20)
..
npdf!CosDosWriteToStream+0xac0:
5a68fc20 6a01            push    1
5a68fc22 ff75f0          push    dword ptr [ebp-10h]
5a68fc25 8bce            mov     ecx,esi
5a68fc27 eb05            jmp     npdf!CosDosWriteToStream+0xace (5a68fc2e)
...
npdf!CosDosWriteToStream+0xace:
5a68fc2e e85d060000      call    npdf!CosDosWriteToStream+0x1130 (5a690290) // [2] construct a stream object
5a68fc33 8bce            mov     ecx,esi
5a68fc35 e876030000      call    npdf!CosDosWriteToStream+0xe50 (5a68ffb0)  // [3] parse the document header (version information)
5a68fc3a 8bd8            mov     ebx,eax
```

After extracting the version information from the document header, the following code will be executed by the application. At [4], this code will first allocate an object of 0x40 bytes and then pass the allocation to a constructor at [5]. This object is used to contain the information relevant to the cross-reference table. This table will be later used by the application in order to locate the different objects that are within the PDF document stream. After constructing the cross-reference table object, it will be stored into one of the properties belonging to a `CNxCosDoc` object.

```
npdf!CosDosWriteToStream+0xb8f:
5a68fcef 6a40            push    40h
5a68fcf1 e831e94e00      call    npdf!CAPContent::Wrap+0x27ce37 (5ab7e627)  // [4] allocation
5a68fcf6 83c404          add     esp,4
5a68fcf9 8945e8          mov     dword ptr [ebp-18h],eax
5a68fcfc c745fc01000000  mov     dword ptr [ebp-4],1
5a68fd03 85c0            test    eax,eax
5a68fd05 740e            je      npdf!CosDosWriteToStream+0xbb5 (5a68fd15)
...
5a68fd07 ff75ec          push    dword ptr [ebp-14h]
5a68fd0a 8bc8            mov     ecx,eax                                    // [5] cross-reference table object from allocation
5a68fd0c 53              push    ebx                                        // high 32-bits of offset for PDF header
5a68fd0d 56              push    esi                                        // low 32-bits of offset for PDF header
5a68fd0e e87d510000      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x46e0 (5a694e90)
...
npdf!CosDosWriteToStream+0xbbe:
5a68fd1e 894614          mov     dword ptr [esi+14h],eax                    // [6] store into property of CNxCosDoc
5a68fd21 85c0            test    eax,eax
5a68fd23 755e            jne     npdf!CosDosWriteToStream+0xc23 (5a68fd83)
```

After assigning the cross-reference table into the `CNxCosDoc` object, the application will assign a function into another one of its properties. At [7], the function that is assigned is responsible for scanning the file containing the document for a particular object identifier in order to repair the cross-reference table. The function from this property will be fetched and called later when the application needs to repair an incorrect or malformed cross-reference for a particular object id. After assigning this function to the `CNxCosDoc` property, the application will pass the object for file-reading to a method belonging to the cross-reference table object at [8]. This method will start with parsing the trailer, and continue to parse the rest of the document.

```
npdf!CosDosWriteToStream+0xc2f:
5a68fd8f c7462430e1685a  mov     dword ptr [esi+24h],offset npdf!CosDocCreate+0x80 (5a68e130)   // [7]
5a68fd96 eb07            jmp     npdf!CosDosWriteToStream+0xc3f (5a68fd9f)
...
npdf!CosDosWriteToStream+0xc3f:
5a68fd9f ff7618          push    dword ptr [esi+18h]                                    // object for reading from file
5a68fda2 8bc8            mov     ecx,eax                                                // [8] cross-reference table object
5a68fda4 e827620000      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x5820 (5a695fd0)
5a68fda9 84c0            test    al,al
5a68fdab 757c            jne     npdf!CosDosWriteToStream+0xcc9 (5a68fe29)
```

Once inside the cross-reference table's method, the application will pass the object for reading from the file and a pointer to a local variable at [9] to a function that scans for the end-of-the file. At the end of the file is the trailer which contains a file offset that points to the cross-reference table for the document. The local variable at [9] will then be initialized with the information that was inferred from the trailer. The information that was returned from that function will then be passed to another method belonging to the cross-reference table object. The size of this structure is 32-bytes and is passed in two 16-byte parameters at [10].

```
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x584c:
5a695ffc 8b7508          mov     esi,dword ptr [ebp+8]          ; object for reading from file
5a695fff 8d45d0          lea     eax,[ebp-30h]                  ; structure containing result from function
5a696002 68ffffff7f      push    7FFFFFFFh
5a696007 6aff            push    0FFFFFFFFh
5a696009 56              push    esi                            ; object for reading from file
5a69600a 50              push    eax                            ; [9] where to store information read from trailer
5a69600b e8e0f1ffff      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x4a40 (5a6951f0)
5a696010 83c410          add     esp,10h
5a696013 807dd000        cmp     byte ptr [ebp-30h],0
5a696017 c745fc00000000  mov     dword ptr [ebp-4],0
5a69601e 7420            je      npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x5890 (5a696040)
...
5a696020 0f1045d8        movups  xmm0,xmmword ptr [ebp-28h]     ; 64-bit offset to table read from trailer
5a696024 83ec18          sub     esp,18h
5a696027 8bcf            mov     ecx,edi                        ; cross-reference table object
5a696029 8bc4            mov     eax,esp
5a69602b 56              push    esi
5a69602c 0f1100          movups  xmmword ptr [eax],xmm0         ; [10] store low 16-bytes of structure to first parameter
5a69602f f30f7e45e8      movq    xmm0,mmword ptr [ebp-18h]
5a696034 660fd64010      movq    mmword ptr [eax+10h],xmm0      ; [10] store high 16-bytes of structure to second parameter
5a696039 e822000000      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x58b0 (5a696060)
5a69603e eb02            jmp     npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x5892 (5a696042)
```

The method that was called will then scan the document for the correct location of the cross-reference table, and then eventually encounter the following block of code. At [11], a method belonging to the cross-reference table object will be called. This method will take the object for reading from the file as well as the 64-bit offset to the beginning of the document as per the table described by the trailer. This is where the application will actually begin parsing and validating the objects from the document trailer. Eventually, the application will actually begin to tokenize the PDF file using the function call at [12]. This function will write the trailer dictionary containing information about how to render the document to its first parameter.

```
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x5b78:
5a696328 51              push    ecx                                    ; high 32-bits of 64-bit parameter
5a696329 50              push    eax                                    ; low 32-bits of 64-bit parameter
5a69632a 53              push    ebx                                    ; object for reading from file
5a69632b 8d45a8          lea     eax,[ebp-58h]
5a69632e 8bcf            mov     ecx,edi                                ; cross-reference table object
5a696330 50              push    eax                                    ; result
5a696331 e80a1e0000      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7990 (5a698140)  ; [11] \
\
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7b04:
5a6982b4 6a00            push    0
5a6982b6 6a00            push    0
5a6982b8 6a00            push    0
5a6982ba 56              push    esi                                    ; object for reading from file
5a6982bb ff7738          push    dword ptr [edi+38h]
5a6982be 8bd8            mov     ebx,eax
5a6982c0 8955ec          mov     dword ptr [ebp-14h],edx
5a6982c3 8d45f0          lea     eax,[ebp-10h]
5a6982c6 50              push    eax                                    ; trailer dictionary from parsing
5a6982c7 e8a49bfeff      call    npdf!CNxVector::CNxVector+0xd0 (5a681e70)  ; [12] tokenize file
5a6982cc 83c418          add     esp,18h
5a6982cf a90000ffff      test    eax,0FFFF0000h
5a6982d4 7412            je      npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7b38 (5a6982e8)
```

As was mentioned, this function is responsible for tokenizing the different parts of a PDF file and processing each object required by the application to read the document. Later in the function when processing an object stream, the application will execute the following code. After processing any dictionary or object belonging to the stream, the application will scan for the "stream" identifier that begins the content of the object stream. At [13], the application will check that the next byte read from the file matches the first character 's', and then at [14] will continue to read from the file to verify that the rest of the string for "stream" can be read. After successfully reading this, the application will seek to the end of the stream, and then skip past the newline character that should terminate it at [15]. At [16], the application will get the current file position and store the offset into a local variable. Afterwards, at [17], the application will call the `GetAtomFromString` to convert the string "Length" into an atom. This method will return the 64-bit atom, and then pass it onto the function at [18]. This function will attempt to fetch the item identified by the "/Length" atom from a dictionary and store it to its third parameter which is at `-20(%ebp)` on the stack.

```
npdf!CNxVector::CNxVector+0x8dd:
5a68267d 8d45e8          lea     eax,[ebp-18h]                                       ; byte that is read from file
5a682680 c745e8ffffffff  mov     dword ptr [ebp-18h],0FFFFFFFFh
5a682687 50              push    eax                                                 ; result byte
5a682688 56              push    esi                                                 ; file reading object
5a682689 e85285feff      call    npdf!ASStmWrite+0x2230 (5a66abe0)                   ; read a single character from file into second
parameter
5a68268e 83c408          add     esp,8
5a682691 837de873        cmp     dword ptr [ebp-18h],73h                             ; [13] check byte against 's'
5a682695 0f8563060000    jne     npdf!CNxVector::CNxVector+0xf5e (5a682cfe)
...
5a68269b 68fc3ebb5a      push    offset npdf!local_file_handle::`vftable'+0x8d0 (5abb3efc)  ; "tream"
5a6826a0 8bce            mov     ecx,esi                                             ; file reading object
5a6826a2 e8b97bfeff      call    npdf!ASStmWrite+0x18b0 (5a66a260)                   ; [14] read from file matching against its
parameter
5a6826a7 84c0            test    al,al
5a6826a9 0f844f060000    je      npdf!CNxVector::CNxVector+0xf5e (5a682cfe)
...
npdf!CNxVector::CNxVector+0x9a2:
5a682742 83f80a          cmp     eax,0Ah                                             ; [15] newline
5a682745 743f            je      npdf!CNxVector::CNxVector+0x9e6 (5a682786)
...
npdf!CNxVector::CNxVector+0x9e6:
5a682786 8b06            mov     eax,dword ptr [esi]
5a682788 8bce            mov     ecx,esi                                             ; file reading object
5a68278a 8b4010          mov     eax,dword ptr [eax+10h]                             ; [16] GetPosition
5a68278d ffd0            call    eax
5a68278f 8bf8            mov     edi,eax
...
5a682791 89550c          mov     dword ptr [ebp+0Ch],edx
5a682794 8b4508          mov     eax,dword ptr [ebp+8]                               ; pointer to dictionary of object stream
5a682797 b90062d05a      mov     ecx,offset npdf!CAPContent::`vftable'+0x139b10 (5ad06200)  ; string "Length"
5a68279c 897dd4          mov     dword ptr [ebp-2Ch],edi
5a68279f 8b00            mov     eax,dword ptr [eax]                                 ; dereference pointer to dictionary
5a6827a1 8945dc          mov     dword ptr [ebp-24h],eax                             ; store dictionary into local variable
5a6827a4 8d45e0          lea     eax,[ebp-20h]                                       ; result object for function call at [18]
5a6827a7 50              push    eax
5a6827a8 e853d9ffff      call    npdf!local_file_handle::write+0x1000 (5a680100)     ; [17] GetAtomFromString
...
5a6827ad 52              push    edx                                                 ; high 32-bits of atom
5a6827ae 50              push    eax                                                 ; low 32-bits of atom
5a6827af ff75dc          push    dword ptr [ebp-24h]                                 ; dictionary
5a6827b2 e839540000      call    npdf!CosNewDict+0x420 (5a687bf0)                    ; [18] get value from dictionary key
5a6827b7 83c410          add     esp,10h
5a6827ba 84c0            test    al,al
5a6827bc 751f            jne     npdf!CNxVector::CNxVector+0xa3d (5a6827dd)
```

After fetching the value of the "/Length" key for the object stream's dictionary, the application will first load it into the `%eax` register at [19]. This register will then be passed to the `CosIntegerValue` function at [20] in order to convert the integer into a 32-bit integer length that can be used. The `CosIntegerValue` function is simply a wrapper around the `CosInteger64Value` function and will branch to its entry-point at [21]. The PDF file format specification allows integers to be references to other objects which can contain the actual integer. In order to support this, the beginning of the `CosInteger64Value` function must check the type of the object that is passed to it. If the type is a `PDFReference`, then the application must recursively dereference objects until an integer length is discovered. This is performed by the function call at [22].

```
npdf!CNxVector::CNxVector+0xa3d:
5a6827dd 8b45e0          mov     eax,dword ptr [ebp-20h]              ; [19] integer fetched from "/Length" atom of dictionary
5a6827e0 85c0            test    eax,eax
5a6827e2 7574            jne     npdf!CNxVector::CNxVector+0xab8 (5a682858)
...
npdf!CNxVector::CNxVector+0xab8:
5a682858 c745fc01000000  mov     dword ptr [ebp-4],1
5a68285f 50              push    eax                                  ; integer object fetched from "/Length"
5a682860 c645fc02        mov     byte ptr [ebp-4],2
5a682864 e8f7600000      call    npdf!CosIntegerValue (5a688960)      ; [20] convert integer object into a 32-bit integer
\
npdf!CosIntegerValue:
5a688960 55              push    ebp
5a688961 8bec            mov     ebp,esp
5a688963 5d              pop     ebp
5a688964 e967000000      jmp     npdf!CosInteger64Value (5a6889d0)    ; [21] chain to CosInteger64Value
...
npdf!CosInteger64Value:
5a6889d0 55              push    ebp
5a6889d1 8bec            mov     ebp,esp
5a6889d3 83ec08          sub     esp,8
5a6889d6 8d45f8          lea     eax,[ebp-8]                          ; result to store integer
5a6889d9 50              push    eax
5a6889da ff7508          push    dword ptr [ebp+8]                    ; integer object
5a6889dd e8de010000      call    npdf!CosNewInteger64+0x1b0 (5a688bc0)  ; [22] check if object is a reference in order to dereference it
5a6889e2 83c408          add     esp,8
5a6889e5 84c0            test    al,al
5a6889e7 740a            je      npdf!CosInteger64Value+0x23 (5a6889f3)
```

This function will first validate that the object is associated with an actual `CNxCosDoc` object. Once that has been verified, the function will then check that the type of the object is a `PDFReference` at [23]. Once this is confirmed, the application then knows it must dereference the object in order to determine its value. When dereferencing the object, the application will need to consult the cross-reference table object in order to locate where the object containing the value for the "/Length" parameter is located at. This is done on an "as-needed" basis in order to be able to deal with malformed documents. At [24], the application will call the function that is necessary to dereference the object that is a `PDFReference`.

```
npdf!CosNewInteger64+0x1b0:
5a688bc0 55              push    ebp
5a688bc1 8bec            mov     ebp,esp
5a688bc3 8b4508          mov     eax,dword ptr [ebp+8]                   ; CNxCosObj
5a688bc6 85c0            test    eax,eax
5a688bc8 0f84b0000000    je      npdf!CosNewInteger64+0x26e (5a688c7e)
5a688bce 83780400        cmp     dword ptr [eax+4],0                     ; verify CNxCosDoc of object is valid
5a688bd2 0f84a6000000    je      npdf!CosNewInteger64+0x26e (5a688c7e)
5a688bd8 8a08            mov     cl,byte ptr [eax]
5a688bda 80f909          cmp     cl,9                                    ; [23] check type of object is PDFReference
5a688bdd 7541            jne     npdf!CosNewInteger64+0x210 (5a688c20)
...
npdf!CosNewInteger64+0x1cf:
5a688bdf 8d4d08          lea     ecx,[ebp+8]                             ; result from dereference
5a688be2 51              push    ecx
5a688be3 50              push    eax                                     ; CNxCosObj
5a688be4 e8070f0000      call    npdf!CosReferenceDirect+0x310 (5a689af0)  ; [24] dereference object
5a688be9 83c408          add     esp,8
5a688bec 84c0            test    al,al
5a688bee 7520            jne     npdf!CosNewInteger64+0x200 (5a688c10)
```

When dereferencing the object, the application will perform the standard validations to ensure that the object is owned by a document and is of the correct type. At [25], the application will load the object from the function's parameter and proceed to validate that it is attached to a document and it is of type `PDFReference`. Once these properties have been checked, the application will load the object identifier that the reference points to and store it into a local variable at [26]. The object identifier directly corresponds to the object id and generation number as per the PDF file format specification. This identifier is later used to identify the target object in the cross-reference table in order for the application to read its value from the file. At [27], the application will then check to ensure the reference is not cyclic and then load the cross-reference table for the document at [27]. If the cross-reference table is already initialized, at [28] the object's identifier will be used to fetch the object using a method belonging to the cross-reference table object.

```
npdf!CosReferenceDirect+0x310:
5a689af0 55              push    ebp
5a689af1 8bec            mov     ebp,esp
5a689af3 6aff            push    0FFFFFFFFh
5a689af5 68cb2bb85a      push    offset npdf!CAPContent::Wrap+0x2813db (5ab82bcb)
5a689afa 64a100000000    mov     eax,dword ptr fs:[00000000h]
5a689b00 50              push    eax
5a689b01 81ec20020000    sub     esp,220h
...
5a689b1e 8b7508          mov     esi,dword ptr [ebp+8]                   ; [25] CNxCosObj
5a689b21 8b7d0c          mov     edi,dword ptr [ebp+0Ch]
5a689b24 85f6            test    esi,esi
5a689b26 0f8425010000    je      npdf!CosReferenceDirect+0x471 (5a689c51)
...
5a689b2c 8b5e04          mov     ebx,dword ptr [esi+4]                   ; verify CNxCosDoc of object is valid
5a689b2f 85db            test    ebx,ebx
5a689b31 0f841a010000    je      npdf!CosReferenceDirect+0x471 (5a689c51)
5a689b37 803e09          cmp     byte ptr [esi],9                        ; check type of object is PDFReference
5a689b3a 7418            je      npdf!CosReferenceDirect+0x374 (5a689b54)
...
npdf!CosReferenceDirect+0x374:
5a689b54 8b4610          mov     eax,dword ptr [esi+10h]                 ; CNxCosObj object identifier
5a689b57 8bcb            mov     ecx,ebx                                 ; CNxCosDoc
5a689b59 8985dcfdffff    mov     dword ptr [ebp-224h],eax                ; [26] store object identifier for object
...
5a689b72 ffb5d8fdffff    push    dword ptr [ebp-228h]
5a689b78 ffb5d4fdffff    push    dword ptr [ebp-22Ch]
5a689b7e e84dfbffff      call    npdf!CosFloatValue+0x620 (5a6896d0)     ; [27] check if reference is cyclic
5a689b83 83c408          add     esp,8
5a689b86 83f801          cmp     eax,1
5a689b89 7551            jne     npdf!CosReferenceDirect+0x3fc (5a689bdc)
...
npdf!CosReferenceDirect+0x3fc:
5a689bdc ffb5dcfdffff    push    dword ptr [ebp-224h]                    ; object identifier
5a689be2 8d85dcfdffff    lea     eax,[ebp-224h]
5a689be8 8bcb            mov     ecx,ebx                                 ; CNxCosDoc
5a689bea 50              push    eax
5a689beb 6a00            push    0
5a689bed 8d85d4fdffff    lea     eax,[ebp-22Ch]
5a689bf3 50              push    eax
5a689bf4 e877f9ffff      call    npdf!CosFloatValue+0x4c0 (5a689570)
...
5a689bf9 8b4604          mov     eax,dword ptr [esi+4]                   ; CNxCosDoc of object
5a689bfc 8b4814          mov     ecx,dword ptr [eax+14h]                 ; [27] fetch cross reference table
5a689bff 85c9            test    ecx,ecx
5a689c01 7435            je      npdf!CosReferenceDirect+0x458 (5a689c38)
...
5a689c03 57              push    edi                                     ; result
5a689c04 ff7610          push    dword ptr [esi+10h]                     ; [28] object identifier
5a689c07 e8d4b70000      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x4c30 (5a6953e0)
5a689c0c 84c0            test    al,al
5a689c0e 752e            jne     npdf!CosReferenceDirect+0x45e (5a689c3e)
```

As prior mentioned, the references for objects can be calculated on demand. This capability is performed by the following method which is responsible for using the document's cross-reference table to locate objects by an id that is passed to it as a parameter. If the document is malformed or the cross-reference table is incomplete in any way, this method can scan the document for the object and update its view of the cross-reference table. This allows the application to be resilient when the cross-reference table is either missing or incomplete. However, the vulnerability highlighted by this advisory is due to a side-effect of the implementation of this methodology. At [29] the application will load some of the properties of the cross-reference table object belonging to the document and cache them into some local variables on the stack. Once this is done, the application will calculate how many entries are currently stored within the cross-reference segment described by the table. Each entry is 0x28 bytes in size. At [30], the application will check if the object id that was passed as a parameter is located within the table's segment. If it is not, the application will then seek to the next object id at [31] by adding 1 in order to allocate space for the new object id. At [32], the application will then pass the current cross-reference table segment, and the new identifier to a method that is responsible for re-allocating the table for the new identifier.

```
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x4c30:
5a6953e0 55              push    ebp
5a6953e1 8bec            mov     ebp,esp
5a6953e3 6aff            push    0FFFFFFFFh
5a6953e5 68813eb85a      push    offset npdf!CAPContent::Wrap+0x282691 (5ab83e81)
5a6953ea 64a100000000    mov     eax,dword ptr fs:[00000000h]
5a6953f0 50              push    eax
5a6953f1 81ec6c040000    sub     esp,46Ch
...
5a69540e 8bf9            mov     edi,ecx                         ; cross-reference table object
5a695410 8b450c          mov     eax,dword ptr [ebp+0Ch]         ; result to store object that will be found
5a695413 8b570c          mov     edx,dword ptr [edi+0Ch]         ; [29] load cross-reference table start
5a695416 8b5d08          mov     ebx,dword ptr [ebp+8]           ; object identifier to search for
5a695419 898598fbffff    mov     dword ptr [ebp-468h],eax
5a69541f 8b4710          mov     eax,dword ptr [edi+10h]         ; [29] load current slot in cross-reference table
5a695422 8bc8            mov     ecx,eax
5a695424 8985a0fbffff    mov     dword ptr [ebp-460h],eax        ; [29] store current slot in cross-reference table to local variable
5a69542a 2bca            sub     ecx,edx
5a69542c 8995a8fbffff    mov     dword ptr [ebp-458h],edx        ; [29] store cross-reference etable start to local variable
5a695432 b867666666      mov     eax,66666667h
5a695437 f7e9            imul    ecx
5a695439 899d90fbffff    mov     dword ptr [ebp-470h],ebx        ; store object identifier to search for
5a69543f c1fa04          sar     edx,4
5a695442 8bf2            mov     esi,edx
5a695444 c1ee1f          shr     esi,1Fh
5a695447 03f2            add     esi,edx
5a695449 3bde            cmp     ebx,esi                         ; [30] check if object id is within cross-reference table length
5a69544b 0f82a6010000    jb      npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x4e47 (5a6955f7)
...
5a695451 8b85a0fbffff    mov     eax,dword ptr [ebp-460h]
5a695457 85db            test    ebx,ebx
5a695459 0f8863010000    js      npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x4e12 (5a6955c2)
...
5a69545f 8d4b01          lea     ecx,[ebx+1]                     ; [31] seek to next object id
5a695462 3bf1            cmp     esi,ecx
5a695464 730c            jae     npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x4cc2 (5a695472)
5a695466 8d470c          lea     eax,[edi+0Ch]                   ; [32] cross reference table start
5a695469 50              push    eax
5a69546a 51              push    ecx                             ; [32] next object id
5a69546b 8bc8            mov     ecx,eax                         ; [32] object containing member for cross reference table variables
5a69546d e81eefffff      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x3be0 (5a694390)
```

The following method is responsible for reallocating the cross-reference table in order to accommodate the new object id that was passed to it as a parameter. Due to the parent function having cached the cross-reference table inside local variables, as this function changes the scope of the cross-reference tables the values in the parent function's local variables will hence become invalid. The first thing this method does is calculate the number of cross-reference table entries for the current segment at [34]. This is later stored into a local variable at `-0xc(%ebp)`. The total number of cross-reference tables entries are then calculated at [35] and then stored into the local variable at `-0x8(%ebp)`. This method will then perform a check if the currently chosen object identifier is larger than the current table length. If this is the case, the method will then proceed to re-allocate the table out from underneath the caller of the method. Prior to doing this, however, the application will calculate the number of cross-reference table entries and begin to check for integer overflows. Afterwards prior to reallocating the table, at [37] will assign the new object identifier from the parameter into the `%ebx` register.

```
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x3be0:
5a694390 55              push    ebp
5a694391 8bec            mov     ebp,esp
5a694393 83ec0c          sub     esp,0Ch
5a694396 53              push    ebx
5a694397 56              push    esi
5a694398 8bf1            mov     esi,ecx
5a69439a b867666666      mov     eax,66666667h
5a69439f 57              push    edi
5a6943a0 8b7d08          mov     edi,dword ptr [ebp+8]      ; object identifier passed as parameter
5a6943a3 8975fc          mov     dword ptr [ebp-4],esi      ; this
5a6943a6 8b5604          mov     edx,dword ptr [esi+4]      ; [34] current cross-reference table position
5a6943a9 2b16            sub     edx,dword ptr [esi]        ; [34] cross-reference table start
5a6943ab 8b4e08          mov     ecx,dword ptr [esi+8]      ; [35] cross-reference table end
5a6943ae 2b0e            sub     ecx,dword ptr [esi]        ; [35] cross-reference table start
5a6943b0 f7ea            imul    edx
5a6943b2 b867666666      mov     eax,66666667h
5a6943b7 c1fa04          sar     edx,4
5a6943ba 8bda            mov     ebx,edx
5a6943bc c1eb1f          shr     ebx,1Fh
5a6943bf 03da            add     ebx,edx
5a6943c1 f7e9            imul    ecx
5a6943c3 895df4          mov     dword ptr [ebp-0Ch],ebx    ; [34] number of cross-reference table entries up to the current position
5a6943c6 c1fa04          sar     edx,4
5a6943c9 8bc2            mov     eax,edx
5a6943cb c1e81f          shr     eax,1Fh
5a6943ce 03c2            add     eax,edx
5a6943d0 8945f8          mov     dword ptr [ebp-8],eax      ; [35] total number of cross-reference table entries
5a6943d3 3bf8            cmp     edi,eax
5a6943d5 766f            jbe     npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x3c96 (5a694446)
...
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x3c27:
5a6943d7 81ff66666606    cmp     edi,6666666h               ; check for integer overflow
5a6943dd 0f8797000000    ja      npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x3cca (5a69447a)
5a6943e3 8b55f8          mov     edx,dword ptr [ebp-8]      ; [36] total number of cross-reference table entries
5a6943e6 8bc8            mov     ecx,eax
5a6943e8 d1e9            shr     ecx,1
5a6943ea b866666606      mov     eax,6666666h
5a6943ef 2bc1            sub     eax,ecx
5a6943f1 3bd0            cmp     edx,eax
5a6943f3 7604            jbe     npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x3c49 (5a6943f9)
5a6943f5 8bdf            mov     ebx,edi                    ; [37] load next object identifier into %ebx
5a6943f7 eb08            jmp     npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x3c51 (5a694401)
```

The changing of the scope of the cross-reference table is done by the application using the following code. At [38], the application will take the total number of xref table entries that were calculated and pass them to the method at [38]. This method perform an allocation using the total number of cross-reference table entries and then store the resulting pointer into the `%esi` register. After allocating space for the new number of entries, the application will calculate the number of elements up to the new object identifier and include it with the pointer from the allocation as the parameters to the method at [39]. This method will zero out all of the elements in the newly allocated table. Afterwards at [40], the application will copy the entries from the previous cross-reference table directly into this new array. Finally at [41], the application will pass the total number of cross-reference table entries, the next object identifier from the method's parameter, and the new array to a method that will release the memory used by the previous cross-reference table array.

```
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x3c49:
5a6943f9 8d1c11          lea     ebx,[ecx+edx]
5a6943fc 3bdf            cmp     ebx,edi
5a6943fe 0f42df          cmovb   ebx,edi
5a694401 53              push    ebx                     ; [38] number of xref table entries to allocate
5a694402 8bce            mov     ecx,esi
5a694404 e837430000      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7f90 (5a698740)
5a694409 8bf0            mov     esi,eax                 ; [38] store into register for copying into later
...
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x3c5b:
5a69440b 8b45f4          mov     eax,dword ptr [ebp-0Ch]    ; [39] total number of xref table entries
5a69440e 8d0c80          lea     ecx,[eax+eax*4]
5a694411 8d14ce          lea     edx,[esi+ecx*8]
5a694414 8bcf            mov     ecx,edi                 ; next object identifier
5a694416 2bc8            sub     ecx,eax                 ; subtract from total number of xref table entries
5a694418 51              push    ecx                     ; [39] number of elements to clear
5a694419 52              push    edx                     ; [39] pointer to beginning of elements to clear
5a69441a 8d4d0c          lea     ecx,[ebp+0Ch]
5a69441d e83e0d0000      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x49b0 (5a695160)
...
5a694422 8b45fc          mov     eax,dword ptr [ebp-4]      ; xref table entries object
5a694425 8bc8            mov     ecx,eax
5a694427 56              push    esi                     ; [40] destination of xref table entries to copy
5a694428 ff7004          push    dword ptr [eax+4]       ; [40] source xref table entries end
5a69442b ff30            push    dword ptr [eax]         ; [40] source xref table entries start
5a69442d e8ce420000      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7f50 (5a698700)
...
5a694432 8b4dfc          mov     ecx,dword ptr [ebp-4]      ; xref table entries object
5a694435 53              push    ebx                     ; [41] number of xref elements
5a694436 57              push    edi                     ; [41] next object identifier from parameter
5a694437 56              push    esi                     ; [41] new xref table entries
5a694438 e843420000      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7ed0 (5a698680)
```

Prior to releasing the old cross-reference table entry array back to the operating system, this method will first perform a few calculations in order to determine how to reallocate it. At [42], the application will take the start of the cross-reference table and subtract it from its end in order to determine the size of the array. At [43], the application will divide the difference by the cross-reference table entry size (0x28) in order to determine the number of elements that can be stored within the array. This length will again be multiplied to determine the size of the cross-reference table to pass as a parameter to `free` at [44]. As mentioned in the prior function which caches the cross-reference table in the function's frame, this call to `free` will invalidate any pointers that were cached. After releasing the memory for the old cross-reference table, the application will then update the new table with the old entries at [45]. At [46], the application will then take the current object identifier that was passed as a parameter, multiply it by the cross-reference table size (0x28), and store it into the current segment's slot. At [47], the total number of cross-reference table entries will be multiplied by the cross-reference table size (0x28) and then store it into the property of the cross-reference table containing the end of the cross-reference table entries array.

```
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7ed0:
5a698680 55              push    ebp
5a698681 8bec            mov     ebp,esp
5a698683 56              push    esi
5a698684 57              push    edi
5a698685 8bf9            mov     edi,ecx
5a698687 8b37            mov     esi,dword ptr [edi]                     ; [42] cross-reference table array start
5a698689 85f6            test    esi,esi
5a69868b 7440            je      npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7f1d (5a6986cd)
...
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7edd:
5a69868d 8b5708          mov     edx,dword ptr [edi+8]                   ; [42] cross-reference table array end
5a698690 b867666666      mov     eax,66666667h
5a698695 2bd6            sub     edx,esi                                 ; [43] get the difference of both pointers
5a698697 f7ea            imul    edx
5a698699 c1fa04          sar     edx,4                                   ; [43] divide by 0x28
5a69869c 8bc2            mov     eax,edx
5a69869e c1e81f          shr     eax,1Fh
5a6986a1 03c2            add     eax,edx
5a6986a3 8d0c80          lea     ecx,[eax+eax*4]
5a6986a6 c1e103          shl     ecx,3
5a6986a9 81f900100000    cmp     ecx,1000h
5a6986af 7212            jb      npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7f13 (5a6986c3)
...
5a6986c3 51              push    ecx                                     ; [44] size
5a6986c4 56              push    esi                                     ; [44] cross-reference table
5a6986c5 e80d5f4e00      call    npdf!CAPContent::Wrap+0x27cde7 (5ab7e5d7)  ; [44] call to free()
5a6986ca 83c408          add     esp,8
...
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7f1d:
5a6986cd 8b4d08          mov     ecx,dword ptr [ebp+8]                   ; [45] cross-reference table start from parameter
5a6986d0 8b450c          mov     eax,dword ptr [ebp+0Ch]                 ; [46] next object identifier
5a6986d3 890f            mov     dword ptr [edi],ecx                     ; [45] update cross-reference table
5a6986d5 8d0480          lea     eax,[eax+eax*4]
5a6986d8 8d04c1          lea     eax,[ecx+eax*8]
5a6986db 894704          mov     dword ptr [edi+4],eax                   ; [46] update current cross-reference table segment
5a6986de 8b4510          mov     eax,dword ptr [ebp+10h]                 ; [47] number of cross-reference table entries
5a6986e1 8d0480          lea     eax,[eax+eax*4]
5a6986e4 8d04c1          lea     eax,[ecx+eax*8]
5a6986e7 894708          mov     dword ptr [edi+8],eax                   ; [47] update end of current cross-reference table
5a6986ea 5f              pop     edi
5a6986eb 5e              pop     esi
5a6986ec 5d              pop     ebp
5a6986ed c20c00          ret     0Ch
```

After the function has reallocated the old cross-reference table and updated it, the application can then return back to the function which cached the cross-reference table entries inside its frame as local variables. Continuing where the method had left off, the following code will extract the 64-bit file offset from the current cross-reference table entry for the referenced object identifier. As the file offset is 64-bits, this is done in two 32-bit parts by the instructions at [48]. Once combining the 64-bit offset from its two 32-bit parts, at [48] the file offset will be used to seek the file reader object to the object that needs to be parsed.

```
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x4ef0:
5a6956a0 8b85a4fbffff    mov     eax,dword ptr [ebp-45Ch]          ; [48] cross-reference table position for the current object identifier
5a6956a6 8b9da8fbffff    mov     ebx,dword ptr [ebp-458h]          ; cross-reference table start
5a6956ac 8b4f30          mov     ecx,dword ptr [edi+30h]           ; low 32-bits of file offset of cross-reference table
5a6956af 8b95a8fbffff    mov     edx,dword ptr [ebp-458h]          ; cross-reference table start
5a6956b5 6a01            push    1
5a6956b7 034c0310        add     ecx,dword ptr [ebx+eax+10h]       ; [48] low 32-bits of file offset in entry
5a6956bb 8b9da4fbffff    mov     ebx,dword ptr [ebp-45Ch]          ; cross-reference table position for the current object identifier
5a6956c1 8b4734          mov     eax,dword ptr [edi+34h]           ; high 32-bits of file offset of cross-reference table
5a6956c4 13441a14        adc     eax,dword ptr [edx+ebx+14h]       ; high 32-bits of file offset in entry
...
5a6956c8 8b16            mov     edx,dword ptr [esi]               ; file reader object
5a6956ca 50              push    eax                               ; high 32-bits of file offset for object
5a6956cb 51              push    ecx                               ; low 32-bits of file offset for object
5a6956cc 8bce            mov     ecx,esi
5a6956ce 8b4214          mov     eax,dword ptr [edx+14h]           ; [49] SetPosition
5a6956d1 ffd0            call    eax
5a6956d3 8b9d90fbffff    mov     ebx,dword ptr [ebp-470h]          ; load object identifier
5a6956d9 84c0            test    al,al
5a6956db 7526            jne     npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x4f53 (5a695703)
```

As the referenced object identifier does not exist, the application will begin to enter a path where the repair procedure from the method of the `CNxCosDoc` is called. This repair procedure will simply scan a range from the file for a string or signature for an object, and return the 64-bit file offset. This done as a last-ditch effort by the application when an object identifier could not be found through the regular means of the cross-reference table. At [50], the application will search through the cross-reference table using the regular means of the object identifier and its generation number. If the cross-reference table entry was not found, the application will begin to generate a signature for the desired object identifier in order to pass to the repair procedure for the document.. This is done by generating a signature with the `sprintf` function. At [51], the format string “%ld %d obj” is used with the object identifier and its generation number. These parameters are then passed to `sprintf` in order to write a signature into a buffer on the stack. Afterwards, the signature on the stack is then passed to the document’s repair procedure at [52] which will unconditionally return a file offset.

```
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x4f53:
5a695703 8b85a4fbffff    mov     eax,dword ptr [ebp-45Ch]                                    ; cross-reference table offset for current
object identifier
5a695709 8b8da8fbffff    mov     ecx,dword ptr [ebp-458h]                                    ; start of cross-reference table
5a69570f ff740118        push    dword ptr [ecx+eax+18h]                                     ; generation number for object
5a695713 8bce            mov     ecx,esi                                                     ; file reader object
5a695715 53              push    ebx                                                         ; object identifier to search
5a695716 e8c549fdff      call    npdf!ASStmWrite+0x1730 (5a66a0e0)                           ; [50] find CNxCosObj by identifier
5a69571b 84c0            test    al,al
5a69571d 0f8565010000    jne     npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x50d8 (5a695888)
...
5a695723 8b85a4fbffff    mov     eax,dword ptr [ebp-45Ch]                                    ; cross-reference table offset for current
object identifier
5a695729 8b8da8fbffff    mov     ecx,dword ptr [ebp-458h]                                    ; start of cross-reference table
5a69572f ff740118        push    dword ptr [ecx+eax+18h]                                     ; generation number for object
5a695733 8d45d0          lea     eax,[ebp-30h]
5a695736 53              push    ebx                                                         ; object identifier to search
5a695737 68cca4bb5a      push    offset npdf!local_file_handle::`vftable'+0x6ea0 (5abba4cc)  ; "%ld %d obj"
5a69573c 50              push    eax                                                         ; destination buffer on stack
5a69573d e8be76feff      call    npdf!nitro::as_layer::cabinet::new_cabinet+0x15d0 (5a67ce00)  ; [51] call to sprintf
...
5a69574e 8b8da8fbffff    mov     ecx,dword ptr [ebp-458h]
5a695754 52              push    edx                                                         ; high 32-bits of when to stop reading
5a695755 50              push    eax                                                         ; low 32-bits of when to stop reading
5a695756 8b85a4fbffff    mov     eax,dword ptr [ebp-45Ch]                                    ; offset into cross-reference table entry
for object id
5a69575c 6a00            push    0
5a69575e 6a00            push    0
5a695760 ff740114        push    dword ptr [ecx+eax+14h]                                     ; high 32-bits of file offset
5a695764 ff740110        push    dword ptr [ecx+eax+10h]                                     ; low 32-bits of file offset
5a695768 8b4f38          mov     ecx,dword ptr [edi+38h]                                     ; CNxCosDoc
5a69576b 8d45d0          lea     eax,[ebp-30h]                                               ; object identifier string from sprintf
5a69576e 50              push    eax                                                         ; string to search for
5a69576f e8fca9ffff      call    npdf!CosDosWriteToStream+0x1010 (5a690170)                  ; [52] CallFileRepairProc
```

After getting the offset to the object using the document repair scanner, the application will update the cross-reference table entry for the object with the 64-bit file offset. This is done by passing the discovered 64-bit file offset for the object to the function call at [53] which will update the cross-reference table entry with the offset. Next at [54], the application will recurse back into the tokenisation function for parsing the PDF file format. Regardless of whether it succeeds or fails, execution will continue onto [55]. At [55], the application will load the current cross-reference table entry offset for the missing object identifier into the `%esi` register, load the cross-reference table from the document into the `%eax` register, and then add them together in order to locate the “in-use” flag for the object. Due to the current value of this stack variable being changed due to the reallocation of the cross-reference table entry array, when dereferencing this flag the application will access memory that has already been released back to the operating system. With the provided proof-of-concept, this will be the first time the application will touch the recently freed memory.

```
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x50b3:
5a695863 ffb59cfbffff    push    dword ptr [ebp-464h]                    ; high 32-bits of file offset for object
5a695869 8bcf            mov     ecx,edi                                 ; cross-reference table for document
5a69586b ffb58cfbffff    push    dword ptr [ebp-474h]                    ; low 32-bits of file offset for object
5a695871 53              push    ebx                                     ; [53] object id to update
5a695872 e8b92d0000      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x7e80 (5a698630)
...
5a695877 8b4f38          mov     ecx,dword ptr [edi+38h]                 ; CNxCosDoc
5a69587a c685affbffff01  mov     byte ptr [ebp-451h],1                   ; unused flag
5a695881 81490c01000080  or      dword ptr [ecx+0Ch],80000001h           ; update CNxCosDoc's flags
...
5a695888 8b95a8fbffff    mov     edx,dword ptr [ebp-458h]                ; xref table start
5a69588e 8b85a4fbffff    mov     eax,dword ptr [ebp-45Ch]                ; xref table entry offset for object id
5a695894 8b4f38          mov     ecx,dword ptr [edi+38h]                 ; CNxCosDoc
5a695897 0fb7440218      movzx   eax,word ptr [edx+eax+18h]              ; generation number for object
5a69589c 50              push    eax                                     ; generation number
5a69589d 53              push    ebx                                     ; object id
5a69589e ff714c          push    dword ptr [ecx+4Ch]                     ; cross-reference table for document
5a6958a1 8d85a0fbffff    lea     eax,[ebp-460h]                          ;
5a6958a7 56              push    esi                                     ; file reader object
5a6958a8 51              push    ecx                                     ; CNxCosDoc
5a6958a9 50              push    eax                                     ; result of parsing
5a6958aa e8c1c5feff      call    npdf!CNxVector::CNxVector+0xd0 (5a681e70)  ; [54] pdf tokenization and parser
5a6958af 83c418          add     esp,18h
5a6958b2 898594fbffff    mov     dword ptr [ebp-46Ch],eax
5a6958b8 e911010000      jmp     npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x521e (5a6959ce)
...
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x521e:
5a6959ce 8bb5a4fbffff    mov     esi,dword ptr [ebp-45Ch]                ; [55] cross-reference table offset for object identifier
5a6959d4 a90000ffff      test    eax,0FFFF0000h
5a6959d9 0f848a000000    je      npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x52b9 (5a695a69)
...
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x52b9:
5a695a69 8b85a8fbffff    mov     eax,dword ptr [ebp-458h]                ; [55] cross-reference table start
5a695a6f f644300804      test    byte ptr [eax+esi+8],4                  ; [55] check flag in cross-reference table for object id
5a695a74 7438            je      npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x52fe (5a695aae)
5a695a76 ffb5a0fbffff    push    dword ptr [ebp-460h]
```

After testing a flag for the current cross-reference table entry, the application will finish execution of the function by updating the cross-reference table with the actual `CNxCosObj` that was constructed for the referenced object. The application will load the recently freed variables referencing cross-reference table start into the `%ecx` register. At [56], this pointer will then be used to write the `CNxCosObj` that was constructed, update the entry's in-use flag, and then write the object that references it into the cross-reference table. Due to the cross-reference table array in the local variables having already having been freed due to the prior-mentioned reallocation, these stores to the cross-reference table entry will write to memory that it shouldn't which is a use-after-free vulnerability. Under the correct conditions this can lead to code execution under the context of the application.

```
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x541d:
5a695bcd 8b8da8fbffff    mov     ecx,dword ptr [ebp-458h]   ; cross-reference table start
5a695bd3 8b85a0fbffff    mov     eax,dword ptr [ebp-460h]   ; cross-reference table entry member for current object
5a695bd9 890431          mov     dword ptr [ecx+esi],eax    ; [56] pointer to member of cross-reference table entry containing the resolved
CNxCosObj
5a695bdc 8b8588fbffff    mov     eax,dword ptr [ebp-478h]   ; CNxCosObj that owns the desired object identifier
5a695be2 804c310801      or      byte ptr [ecx+esi+8],1     ; [56] update the in-use flag for the cross-reference table entry
5a695be7 89443104        mov     dword ptr [ecx+esi+4],eax  ; [56] update the cross-reference table entry with the owner of the object
..
5a695beb 8b85a0fbffff    mov     eax,dword ptr [ebp-460h]   ; current object from cross-reference table entry
5a695bf1 8b8d98fbffff    mov     ecx,dword ptr [ebp-468h]   ; result for caller
5a695bf7 8901            mov     dword ptr [ecx],eax        ; store CNxCosObj for caller
```

Crash Information

To visualize the change in scope, you can first set a breakpoint on the function responsible for re-allocating the cross-reference table.

```
0:000> bp npdf+c546d
0:000> g
Breakpoint 0 hit
eax=379c7fcc ebx=00000004 ecx=379c7fcc edx=00000004 esi=00000004 edi=379c7fc0
eip=5a69546d esp=0195cbc0 ebp=0195d050 iopl=0         nv up ei ng nz ac pe cy
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00200297
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x4cbd:
5a69546d e81eefffff      call    npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x3be0 (5a694390)
```

Dumping the value of the variable on the stack shows each cross-reference table entry is presently allocated.

```
0:000> dc @ebp-458 L4
0195cbf8  313b2f60 155d3fa4 0195cc1c 77914eac  `/;1.?]......N.w
0:000> dc @$p
313b2f60  00000000 00000000 c0c0c000 c0c0c0c0  ................
313b2f70  00000000 00000000 00000000 00000000  ................
313b2f80  00000000 c0c0c0c0 37daeda0 37daee18  ............7...7
313b2f90  c0c0c003 c0c0c0c0 00000009 00000000  ................
313b2fa0  00000000 00000000 00000000 c0c0c0c0  ................
313b2fb0  37daee30 37daeef0 c0c0c003 c0c0c0c0  0..7...7........
313b2fc0  00000042 00000000 00000000 00000000  B...............
313b2fd0  00000000 c0c0c0c0 00000000 00000000  ................
```

Stepping over the function will re-allocate the cross-reference table entry.

```
0:000> p
eax=2c797000 ebx=00000004 ecx=2c796f10 edx=02961078 esi=00000004 edi=379c7fc0
eip=5a695472 esp=0195cbc8 ebp=0195d050 iopl=0         nv up ei pl nz ac pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00200216
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x4cc2:
5a695472 8b4738          mov     eax,dword ptr [edi+38h] ds:0023:379c7ff8=3177cf90
```

Dumping it shows that it has just been freed.

```
0:000> dc @$exp
313b2f60  ???????? ???????? ???????? ????????  ????????????????
313b2f70  ???????? ???????? ???????? ????????  ????????????????
313b2f80  ???????? ???????? ???????? ????????  ????????????????
313b2f90  ???????? ???????? ???????? ????????  ????????????????
313b2fa0  ???????? ???????? ???????? ????????  ????????????????
313b2fb0  ???????? ???????? ???????? ????????  ????????????????
313b2fc0  ???????? ???????? ???????? ????????  ????????????????
313b2fd0  ???????? ???????? ???????? ????????  ????????????????
```

Dumping out the cross-reference table shows that the values at `0x0(%edi)` (table start) and `0x4(%edi)` (table end) have changed.

```
0:000> dc @edi L(40/4)
379c7fc0  316a0fe0 316a1000 316a1000 2c796f10  ..j1..j1..j1.oy,
379c7fd0  2c796fd8 2c797000 00000000 00000000  .oy,.py,........
379c7fe0  00000000 c0c0c0c0 00000000 00000000  ................
379c7ff0  00000000 00000000 3177cf90 c0c0c000  .........w1....
```

Resuming execution, the following crash will occur.

```
(9fc.684): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=313b2f60 ebx=00000003 ecx=1b58ef4f edx=02961078 esi=00000078 edi=379c7fc0
eip=5a695a6f esp=0195d36c ebp=0195d7f4 iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00210206
npdf!nitro::digital_signature::signature_verifier::GetSignerCertificate+0x52bf:
5a695a6f f644300804      test    byte ptr [eax+esi+8],4     ds:0023:313b2fe0=??
```

The base addresses of the libraries in this report.

```
0:000> lm m npdf
Browse full module list
start    end        module name
5a5d0000 5b017000   npdf       (export symbols)       npdf.dll
010c0000 01941000   NitroPDF   (deferred)
```

Exploit Proof of Concept

In the provided proof-of-concept, the stream for object 3 (“3 0 obj”) references a length that should be resolved by object 4 (“/Length 4 0 R”) which doesn’t exist. As the cross-reference table pointed to by the trailer is corrupted, this results in Nitro Pro only parsing 3 elements of the cross-reference table. When dereferencing the object for the length, this results in Nitro Pro attempting to resize the cross-reference table in order to accommodate a slot for the 4th object. It is this resize that pushes the previous cross-reference table out of scope thus invalidating the cross-reference table entry. Although there are likely multiple ways of reaching this particular code path as the relevant function is used by many parts of Nitro PDF’s parser, the easiest that was found by the author is by triggering a dereference for an object that is larger than the currently existing cross-reference table.

TIMELINE

2020-05-07 - Vendor Disclosure
2020-09-01 - Vendor Patched
2020-09-15 - Public Release

CREDIT

Discovered by a member of Cisco Talos.
