Talos Vulnerability Report

TALOS-2021-1266

Nitro Pro PDF JavaScript TimeOutObject double free vulnerability

OCTOBER 13, 2021

CVE NUMBER

CVE-2021-21797

Summary

An exploitable double-free vulnerability exists in the JavaScript implementation of Nitro Pro PDF. A specially crafted document can cause a reference to a timeout object to be stored in two different places. When closed, the document will result in the reference being released twice. This can lead to code execution under the context of the application. An attacker can convince a user to open a document to trigger this vulnerability.

Tested Versions

Nitro Pro 13.31.0.605
Nitro Pro 13.33.2.645

Product URLs

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-415 - Double Free

Details

Nitro Software Inc. develops a variety of feature-rich PDF software that allows users to read and manipulate the files. This includes their flagship product, Nitro Pro, as part of their Nitro Productivity Suite. This product allows users to create and modify PDFs and other digital documents. This software includes support for several capabilities via third-party libraries to parse the PDF specification. This includes software produced by Kakadu Software for providing JPEG2000 image file format support, the LibTIFF library for providing support for TIFF image files, and Mozilla's SpiderMonkey for providing JavaScript support within their software.

In order to support many of the features of Nitro PDF, the application implements a local dispatching interface for its various components to be able to communicate with it. Specifically, this dispatching interface is an array of functions that are collectively referred to as the HFT Extension Manager and is of the type `HFTServer`. Upon the application populating this array, various plugins will then be loaded by the application and then register their capabilities with the `HFTServer` host. The Sixth Edition of the Portable Document Format (PDF) specification includes a javascript extension in order to allow document creators to improve the interactivity of their documents. Thus, the application implements this capability by loading a javascript plugin and registering it via the HFT Extension Manager interface. The javascript implementation that is used by the application is based on Mozilla's SpiderMonkey, and includes a number of bindings that allow a content developer to automate various aspects of the document.

The SpiderMonkey library allows a developer to develop their own custom classes and objects via its API in order to enable a document creator to script various parts of the application. When the application initializes its javascript plugin, the application needs to create a number of objects and classes at in order to expose various PDF automation points to the document creator. The following code represents a closure (or an anonymous function) and is responsible for initializing all of the available javascript classes. Firstly at [1], a number of objects that were captured from the encompassing function are extracted and then written to globals that are accessible by the plugin. This includes the parent object, the global root object for garbage collection, and the `pdf_java_script_actions` object. After assigning the captured variables, the function call at [2] is executed to define the "app" object by attaching its private data and any necessary methods or properties.

```
np_java_script+0x511f0:
2adb11f0 55              push    ebp
2adb11f1 8bec            mov     ebp,esp
2adb11f3 83ec40          sub     esp,40h
2adb11f6 a17441ec2a      mov     eax,dword ptr [np_java_script!CxImageJPG::`vftable'+0x4f8b8 (2aec4174)]
2adb11fb 33c5            xor     eax,ebp
2adb11fd 8945fc          mov     dword ptr [ebp-4],eax
2adb1200 53              push    ebx
2adb1201 8b5d08          mov     ebx,dword ptr [ebp+8]
2adb1204 894dc4          mov     dword ptr [ebp-3Ch],ecx                                        ; this
...
2adb120d e8de9e0000      call    np_java_script!nitro::java_script::create_plugin+0x9890 (2adbb0f0)
2adb1212 8bcb            mov     ecx,ebx
2adb1214 a3dce7ec2a      mov     dword ptr [np_java_script!CxImageJPG::`vftable'+0x59f20 (2aece7dc)],eax    ; [1] parent plugin object
containing global state
2adb1219 e8c29e0000      call    np_java_script!nitro::java_script::create_plugin+0x9880 (2adbb0e0)
2adb121e 8bcb            mov     ecx,ebx
2adb1220 a3d8e7ec2a      mov     dword ptr [np_java_script!CxImageJPG::`vftable'+0x59f1c (2aece7d8)],eax    ; root JSContext
2adb1225 e8d69e0000      call    np_java_script!nitro::java_script::create_plugin+0x98a0 (2adbb100)
2adb122a 8bcb            mov     ecx,ebx
2adb122c a3e4e7ec2a      mov     dword ptr [np_java_script!CxImageJPG::`vftable'+0x59f28 (2aece7e4)],eax    ; pdf_java_script_actions
2adb1231 e8da9e0000      call    np_java_script!nitro::java_script::create_plugin+0x98b0 (2adbb110)
2adb1236 a3e0e7ec2a      mov     dword ptr [np_java_script!CxImageJPG::`vftable'+0x59f24 (2aece7e0)],eax    ; array of objects
2adb123b e88059fbff      call    np_java_script+0x6bc0 (2ad66bc0)                                           ; [2] define app object and
attach its private data
2adb1240 85c0            test    eax,eax
2adb1242 0f8450020000    je      np_java_script+0x51498 (2adb1498)
```

Inside the function call, the methods and properties for the “app” object are assigned and stored into an array of `JSFunctionSpec` and `JSPropertySpec` elements on the stack. Prior to assigning these methods and properties, the function will first push the arguments for the function call to `JS_DefineObject` at [3]. These parameters include the `JSClass` definition for the “App” class, and the object’s name “app”. After pushing its parameter, the function will continue to assign the `JSFunctionSpec` for the methods to attach. At [4], the function will assign the `JSNative` implementations that will be dispatched when the “setInterval” or “setTimeOut” methods are called. Once the array of `JSFunctionSpec` and `JSPropertySpec` elements have been assigned, the function will proceed to use them by first calling the `JS_DefineObject` function at [5], followed by the registration of the properties at [6] with the `JS_DefineProperties` function, and then the registration of the object’s functions at [7] with the `JS_DefineFunctions` function.

```
np_java_script+0x6bc0:
2ad66bc0 55              push    ebp
2ad66bc1 8bec            mov     ebp,esp
2ad66bc3 81ec94010000    sub     esp,194h
2ad66bc9 a17441ec2a      mov     eax,dword ptr [np_java_script!CxImageJPG::`vftable'+0x4f8b8 (2aec4174)]
2ad66bce 33c5            xor     eax,ebp
2ad66bd0 8945fc          mov     dword ptr [ebp-4],eax
...
np_java_script+0x6e97:
2ad66e97 6a06            push    6                                                         ; flags
2ad66e99 50              push    eax                                                       ; class prototype
2ad66e9a 680010ec2a      push    offset np_java_script!CxImageJPG::`vftable'+0x4c744 (2aec1000)         ; [3] JSClass for "App"
2ad66e9f 686409e62a      push    offset np_java_script!CxImageJPG::Encode+0x9e9e4 (2ae60964)           ; [3] "app"
2ad66ea4 ff35dce7ec2a    push    dword ptr [np_java_script!CxImageJPG::`vftable'+0x59f20 (2aece7dc)]    ; parent object containing global
state
2ad66eaa c745b0e07fd62a  mov     dword ptr [ebp-50h],offset np_java_script+0x7fe0 (2ad67fe0)
2ad66eb1 ff35d8e7ec2a    push    dword ptr [np_java_script!CxImageJPG::`vftable'+0x59f1c (2aece7d8)]    ; root JSContext
...
np_java_script+0x6ed9:
2ad66ed9 c745cc4c09e62a  mov     dword ptr [ebp-34h],offset np_java_script!CxImageJPG::Encode+0x9e9cc (2ae6094c)    ; [4]
JSFunctionSpec.name = "setInterval"
2ad66ee0 c745d03085d62a  mov     dword ptr [ebp-30h],offset np_java_script+0x8530 (2ad68530)                        ; JSFunctionSpec.call =
JSAppSetInterval
2ad66ee7 c745d402000000  mov     dword ptr [ebp-2Ch],2                                                              ; JSFunctionSpec.nargs =
2
2ad66eee 8945d8          mov     dword ptr [ebp-28h],eax
2ad66ef1 c745dc5809e62a  mov     dword ptr [ebp-24h],offset np_java_script!CxImageJPG::Encode+0x9e9d8 (2ae60958)    ; [4]
JSFunctionSpec.name = "setTimeOut"
2ad66ef8 c745e02089d62a  mov     dword ptr [ebp-20h],offset np_java_script+0x8920 (2ad68920)                        ; JSFunctionSpec.call =
JSAppSetTimeOut
2ad66eff c745e402000000  mov     dword ptr [ebp-1Ch],2                                                              ; JSFunctionSpec.nargs =
2
2ad66f06 8945e8          mov     dword ptr [ebp-18h],eax
2ad66f09 0f1145ec        movups  xmmword ptr [ebp-14h],xmm0
2ad66f0d ff1514e7e52a    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9c794 (2ae5e714)]                   ; [5] call
JS_DefineObject
2ad66f13 8bf0            mov     esi,eax                                                                            ; save new JSObject
2ad66f15 83c418          add     esp,18h
2ad66f18 85f6            test    esi,esi
2ad66f1a 743f            je      np_java_script+0x6f5b (2ad66f5b)
...
2ad66f1c 8d856cfeffff    lea     eax,[ebp-194h]                                                                     ; address of
JSPropertySpec array
2ad66f22 50              push    eax                                                                                ; ps
2ad66f23 56              push    esi                                                                                ; obj
2ad66f24 ff35d8e7ec2a    push    dword ptr [np_java_script!CxImageJPG::`vftable'+0x59f1c (2aece7d8)]                ; root JSContext
2ad66f2a ff1518e7e52a    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9c798 (2ae5e718)]                   ; [6]
JS_DefineProperties
2ad66f30 83c40c          add     esp,0Ch
2ad66f33 85c0            test    eax,eax
2ad66f35 7446            je      np_java_script+0x6f7d (2ad66f7d)
2ad66f37 8d85ecfeffff    lea     eax,[ebp-114h]                                                                     ; address of
JSFunctionSpec array
2ad66f3d 50              push    eax                                                                                ; fs
2ad66f3e 56              push    esi                                                                                ; obj
2ad66f3f ff35d8e7ec2a    push    dword ptr [np_java_script!CxImageJPG::`vftable'+0x59f1c (2aece7d8)]                ; root JSContext
2ad66f45 ff1538e7e52a    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9c7b8 (2ae5e738)]                   ; [7] JS_DefineFunctions
2ad66f4b 83c40c          add     esp,0Ch
2ad66f4e 85c0            test    eax,eax
2ad66f50 742b            je      np_java_script+0x6f7d (2ad66f7d)
```

Once the “app” object has been defined and had its namespace populated, when the “app.setInterval” method is called by the javascript interpreter the following function will be executed. The first thing this function will do is to check the number of parameters and their types. After confirming both the parameters are the correct type, at [8] the function will convert its first parameter to a string and store it to the `%ebx` register. At [9], the function will then fetch the second parameter as a number and then convert it to an unsigned integer to store in the `%edi` register. Once storing the parameters to their respective registers, the function will allocate space on the heap at [10] in order to construct an object representing the private data of the callback and initialize it at [11]. This private data is later registered by the plugin as it maintains the javascript callback that is to be executed when a timer notification is broadcasted. When the application broadcasts a timer notification, the application will then fetch a reference to this object and then use it to determine which javascript to execute. It is specifically the way the application manages the scope of this object that is directly responsible for the vulnerability described within this document.

```
np_java_script+0x8530:
2ad68530 55              push    ebp
2ad68531 8bec            mov     ebp,esp
2ad68533 6aff            push    0FFFFFFFFh
2ad68535 682035e52a      push    offset np_java_script!CxImageJPG::Encode+0x915a0 (2ae53520)
2ad6853a 64a100000000    mov     eax,dword ptr fs:[00000000h]
2ad68540 50              push    eax
2ad68541 83ec10          sub     esp,10h
...
np_java_script+0x85fc:
2ad685fc 8b7508          mov     esi,dword ptr [ebp+8]                                           ; %esi := JSContext
2ad685ff 51              push    ecx                                                             ; callback parameter
2ad68600 56              push    esi
2ad68601 ff15f0e7e52a    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9c870 (2ae5e7f0)]       ; JS_ValueToString
2ad68607 50              push    eax                                                             ; callback parameter value
2ad68608 ff1540e7e52a    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9c7c0 (2ae5e740)]       ; JS_GetStringChars
2ad6860e 8bd8            mov     ebx,eax                                                         ; [8] store callback string
...
2ad68610 8d45e4          lea     eax,[ebp-1Ch]
2ad68613 50              push    eax                                                             ; result
2ad68614 ff7704          push    dword ptr [edi+4]                                               ; timeout parameter
2ad68617 56              push    esi                                                             ; JSContext (%esi)
2ad68618 ff15ece7e52a    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9c86c (2ae5e7ec)]       ; [9] JS_ValueToNumber
2ad6861e f20f1045e4      movsd   xmm0,mmword ptr [ebp-1Ch]                                       ; timeout parameter
2ad68623 83c418          add     esp,18h
2ad68626 e846a60500      call    np_java_script!CxImageJPG::Encode+0xcf1 (2adc2c71)              ; __dotui3
2ad6862b 8bf8            mov     edi,eax
...
np_java_script+0x8646:
2ad68646 6a0c            push    0Ch
2ad68648 e898a00500      call    np_java_script!CxImageJPG::Encode+0x765 (2adc26e5)              ; [10] allocate private data for
callback
2ad6864d 8bf0            mov     esi,eax
2ad6864f 0f57c0          xorps   xmm0,xmm0
2ad68652 53              push    ebx                                                             ; callback string
2ad68653 8975ec          mov     dword ptr [ebp-14h],esi                                         ; store private data within stack
frame
2ad68656 660fd606        movq    mmword ptr [esi],xmm0
2ad6865a c7460800000000  mov     dword ptr [esi+8],0                                             ; [11] initialize object for private
data
2ad68661 e85a5d0400      call    np_java_script+0x4e3c0 (2adae3c0)                               ; duplicate callback string
2ad68666 8906            mov     dword ptr [esi],eax                                             ; [11] initialize object for private
data
2ad68668 83c408          add     esp,8
...
2ad6866b c6460400        mov     byte ptr [esi+4],0                                              ; [11] initialize object for private
data
2ad6866f a17ce3ec2a      mov     eax,dword ptr [np_java_script!CxImageJPG::`vftable'+0x59ac0 (2aece37c)]
2ad68674 85c0            test    eax,eax
2ad68676 7458            je      np_java_script+0x86d0 (2ad686d0)
```

After constructing the private data object, the function will proceed to use the parameter containing the timeout that was converted to an integer and pass it along with the private data to the function call at [12]. This function call dispatches to the `HFTServer` function at offset +0x84 and will register the private data that was passed to it so that it will be called upon receiving a timer notification. The implementation of this `HFTServer` function will then allocate 0x10 bytes of space for an object, store the object containing the private data into it. Afterwards, the 0x10-sized object will be appended to an array of type `CAPArray` that is stored globally. Once the object was stored to the `CAPArray`, the `HFTServer` function will return and then append the same object to another array. As this same reference is being stored twice, the same object can be referenced in more than one place.

Afterwards, a `JSObject` will be constructed using the function call at [13]. The function call at [13] is part of the javascript implementation's API and has the name `JS_NewObject`. This API call will create a new `JSObject` using the `JSClass` definition that is passed as its second parameter. When defining a `JSClass` to be passed to `JS_NewObject`, a number of fields are populated in order to tell the implementation how it should manage the `JSObject`. In the definition that is passed as the parameter to the API call at [13], the name of this class is "TimeOut". Some of the other necessary fields of a `JSClass` allow an implementer to provide various operations for managing their newly created object. One of these operations is the `JSClass.finalize` property. This property is to help manage the scope of the object when the javascript implementation needs to destroy the object. The `JSClass` definition that is used for the call to `JS_NewObject` contains only an implementation of this operation and will thus be called when SpiderMonkey needs to destroy the newly instantiated "TimeOut" object. After creating the new `JSObject` with this "TimeOut" class, the function will store it in the `%esi` register. Afterwards, the new "TimeOut" object along with the private data for this object will then be passed to the `JS_SetPrivate` API call at [14]. As the `JSClass.finalize` properly was defined, upon destruction of the `JSObject` the javascript implementation will call its `JSClass.finalize` function.

```
np_java_script+0x86d0:
2ad686d0 a128ebec2a      mov     eax,dword ptr [np_java_script!CxImageJPG::`vftable'+0x5a26c (2aeceb28)]    ; HFT Extension Manager
2ad686d5 57              push    edi                                                                 ; timeout integer
2ad686d6 56              push    esi                                                                 ; private data object
2ad686d7 c745fc00000000  mov     dword ptr [ebp-4],0
2ad686de 8b8084000000    mov     eax,dword ptr [eax+84h]                                             ; HFTServer+0x84
2ad686e4 68a06fd62a      push    offset np_java_script+0x6fa0 (2ad66fa0)
2ad686e9 c645fc01        mov     byte ptr [ebp-4],1
2ad686ed ffd0            call    eax                                                                 ; [12] register private data
object with extension manager
2ad686ef 83c40c          add     esp,0Ch
2ad686f2 c6460501        mov     byte ptr [esi+5],1                                                  ; activate private data object
2ad686f6 8d45ec          lea     eax,[ebp-14h]                                                       ; fetch private data object from
stack frame
2ad686f9 b920e0ec2a      mov     ecx,offset np_java_script!CxImageJPG::`vftable'+0x59764 (2aece020)         ; global array of JSObject
2ad686fe 50              push    eax                                                                 ; private data
2ad686ff e8fc0e0000      call    np_java_script+0x9600 (2ad69600)                                    ; [12] append to global array of
private data
2ad68704 8b7dec          mov     edi,dword ptr [ebp-14h]
...
np_java_script+0x8712:
2ad68712 6a00            push    0
2ad68714 6a00            push    0
2ad68716 684810ec2a      push    offset np_java_script!CxImageJPG::`vftable'+0x4c78c (2aec1048)             ; JSClass for "TimeOut"
2ad6871b ff35d8e7ec2a    push    dword ptr [np_java_script!CxImageJPG::`vftable'+0x59f1c (2aece7d8)]        ; JSContext
2ad68721 ff1510e7e52a    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9c790 (2ae5e710)]           ; [13] JS_NewObject
2ad68727 8bf0            mov     esi,eax                                                             ; store object
...
2ad68729 57              push    edi                                                                 ; private data
2ad6872a 56              push    esi                                                                 ; JSObject
2ad6872b ff35d8e7ec2a    push    dword ptr [np_java_script!CxImageJPG::`vftable'+0x59f1c (2aece7d8)]        ; JSContext
2ad68731 ff150ce7e52a    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9c78c (2ae5e70c)]           ; [14] JS_SetPrivate
2ad68737 8b4518          mov     eax,dword ptr [ebp+18h]
```

The name of the `JSClass.finalize` property for the “TimeOut” `JSObject` is suspected by the author to be “JSTimeOutDestructor”. The implementation of the “JSTimeOutDestructor” function is listed in the following disassembly. When destroying the “TimeOut” object, this function will first grab the `JSObject` from its parameter in order to extract its private object using the `JS_GetPrivate` API at [15]. Once fetching the private data, the function will proceed to destroy it by checking its first property and then passing it to the `nitro::very_unsafe::ASfree` call at [16]. Once freeing its first property, the entire private data object is then passed to the `free` call at [17]. It is prudent to note that the private data that is released within this function had been previously copied into the `CAPArray` when the object was registered with the HFT Extension Manager. This implies that there are two references to the same object without there being any kind of reference counting to keep track of the private data.

```
np_java_script+0x8b90:
2ad68b90 55              push    ebp
2ad68b91 8bec            mov     ebp,esp
2ad68b93 56              push    esi
2ad68b94 57              push    edi
2ad68b95 8b7d0c          mov     edi,dword ptr [ebp+0Ch]                                   ; "TimeOut" JSObject
2ad68b98 85ff            test    edi,edi
2ad68b9a 751c            jne     np_java_script+0x8bb8 (2ad68bb8)
np_java_script+0x8bf9:
2ad68bf9 57              push    edi                                                       ; "TimeOut" JSObject
2ad68bfa ff7508          push    dword ptr [ebp+8]                                         ; JSContext
2ad68bfd ff1508e7e52a    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9c788 (2ae5e708)]   ; [15] JS_GetPrivate
2ad68c03 8bf0            mov     esi,eax                                                   ; store private data to register
2ad68c05 83c408          add     esp,8
2ad68c08 85f6            test    esi,esi
2ad68c0a 7421            je      np_java_script+0x8c2d (2ad68c2d)
...
2ad68c0c 8b06            mov     eax,dword ptr [esi]                                       ; check first member of private object
2ad68c0e 85c0            test    eax,eax
2ad68c10 7410            je      np_java_script+0x8c22 (2ad68c22)
...
2ad68c12 50              push    eax                                                       ; first member of private object
2ad68c13 ff153cf0e52a    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9d0bc (2ae5f03c)]   ; [16] nitro::very_unsafe::ASfree
2ad68c19 83c404          add     esp,4
...
2ad68c1c c70600000000    mov     dword ptr [esi],0
2ad68c22 6a0c            push    0Ch
2ad68c24 56              push    esi                                                       ; private object
2ad68c25 e8f09a0500      call    np_java_script!CxImageJPG::Encode+0x79a (2adc271a)                 ; [17] free private object
2ad68c2a 83c408          add     esp,8
```

Upon the application closing the document, a notification of type 51 will be broadcast in order to notify the different components of the application that any resources associated with the document are to be released. If the javascript embedded within the document proceeds to close the document, the following “CloseDocIdleProc” anonymous function will be executed in response. At [18], the application will dispatch to the function at offset +0x9c of the `HFTServer`. This function will destroy of the document is done via the implementation of the `CNxAvDoc::DeleteContents` method which will call the `npdf.dll!PDDocClose` function. Calling this function will result in a notification being broadcasted to notify each of the application’s components that the current document is being destroyed.

```
np_java_script+0x1e380:
2ad7e380 55              push    ebp
2ad7e381 8bec            mov     ebp,esp
2ad7e383 6aff            push    0FFFFFFFFh
2ad7e385 68c047e52a      push    offset np_java_script!CxImageJPG::Encode+0x92840 (2ae547c0)
...
np_java_script+0x1e3eb:
2ad7e3eb a128ebec2a      mov     eax,dword ptr [np_java_script!CxImageJPG::`vftable'+0x5a26c (2aeceb28)]    ; HFT Extension Manager
2ad7e3f0 8b889c000000    mov     ecx,dword ptr [eax+9Ch]                                            ; HFTServer+0x9c
2ad7e3f6 8a4604          mov     al,byte ptr [esi+4]
2ad7e3f9 50              push    eax
2ad7e3fa 52              push    edx
2ad7e3fb ffd1            call    ecx                                                                ; [18] broadcast notification
```

The handler for notification 51 will execute the following method when the function call at offset +0x9c of the `HFTServer` is made. This method is responsible for fetching the `pd_doc` and the `App` object when the notification was constructed and to pass them as parameters to the handler associated with the notification at [19]. The handler for the notification will do a number of things to clean up the document, however, the handler will first pass the `pd_doc` that was passed as a parameter to the function call at [20].

```
np_java_script+0x9630:
2ad69630 55              push    ebp
2ad69631 8bec            mov     ebp,esp
2ad69633 8b4508          mov     eax,dword ptr [ebp+8]
2ad69636 ff7114          push    dword ptr [ecx+14h]                ; struct App
2ad69639 ff7008          push    dword ptr [eax+8]                  ; pd_doc
2ad6963c 8b4110          mov     eax,dword ptr [ecx+10h]            ; handler
2ad6963f ffd0            call    eax                                ; [19] \
2ad69641 83c408          add     esp,8
2ad69644 5d              pop     ebp
2ad69645 c20400          ret     4
\
np_java_script+0x8d60:
2ad68d60 55              push    ebp
2ad68d61 8bec            mov     ebp,esp
2ad68d63 56              push    esi
2ad68d64 57              push    edi
2ad68d65 6a00            push    0
2ad68d67 ff7508          push    dword ptr [ebp+8]                  ; pd_doc
2ad68d6a e8d1c6ffff      call    np_java_script+0x5440 (2ad65440)   ; [20] iterate through global array of objects
```

The following function is responsible for iterating through the array of objects that were previously added by the javascript plugin when they were constructed, and proceed to destroy each of them individually. Firstly, the function will calculate the boundaries of the global array and use it to set the number of iterations for the loop at [22]. For each iteration of this loop, similar to the `JSClass.finalize` operation defined by the “TimeOut” class the function will proceed to delete each element in the array by checking its first property. If it is defined, it will be passed to the `nitro::very_unsafe::ASfree` function at [23]. Once the callback string for the element is destroyed, the element itself will then be passed to the `free` call at [24]. If the private data associated with the “TimeOut” object can be made to be destroyed by the javascript embedded within the document while the notification of type 51 is dispatched to the plugin, then a double-free condition can be made to occur with the private data from the “TimeOut” object. This can lead to code execution under the context of the application.

```
np_java_script+0x5440:
2ad65440 55              push    ebp
2ad65441 8bec            mov     ebp,esp
2ad65443 8b0d20e0ec2a    mov     ecx,dword ptr [np_java_script!CxImageJPG::`vftable'+0x59764 (2aece020)]    ; [21] linked list of private
data objects
2ad65449 57              push    edi
2ad6544a 8b3d24e0ec2a    mov     edi,dword ptr [np_java_script!CxImageJPG::`vftable'+0x59768 (2aece024)]    ; [21] linked list of private
data objects
2ad65450 2bf9            sub     edi,ecx
2ad65452 c1ff02          sar     edi,2
2ad65455 83ef01          sub     edi,1                                                                      ; index for loop
2ad65458 0f8882000000    js      np_java_script+0x54e0 (2ad654e0)
...
np_java_script+0x5463:
2ad65463 8b34b9          mov     esi,dword ptr [ecx+edi*4]                                                  ; [22] loop through each element
of array
2ad65466 8b4608          mov     eax,dword ptr [esi+8]
2ad65469 85c0            test    eax,eax
2ad6546b 7405            je      np_java_script+0x5472 (2ad65472)
2ad6546d 394508          cmp     dword ptr [ebp+8],eax
2ad65470 7567            jne     np_java_script+0x54d9 (2ad654d9)
..
2ad65496 8b06            mov     eax,dword ptr [esi]                                                        ; check first property of array
element
2ad65498 85c0            test    eax,eax
2ad6549a 740a            je      np_java_script+0x54a6 (2ad654a6)
...
2ad6549c 50              push    eax
2ad6549d ff153cf0e52a    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9d0bc (2ae5f03c)]           ; [23]
nitro::very_unsafe::ASfree
2ad654a3 83c404          add     esp,4
...
2ad654a6 6a0c            push    0Ch
2ad654a8 56              push    esi
2ad654a9 e86cd20500      call    np_java_script!CxImageJPG::Encode+0x79a (2adc271a)                         ; [24] free array element
...
2ad654c9 8b0d20e0ec2a    mov     ecx,dword ptr [np_java_script!CxImageJPG::`vftable'+0x59764 (2aece020)]
2ad654cf 83c414          add     esp,14h
2ad654d2 832d24e0ec2a04  sub     dword ptr [np_java_script!CxImageJPG::`vftable'+0x59768 (2aece024)],4      ; reduce number of elements in
array
2ad654d9 83ef01          sub     edi,1                                                                      ; next index
2ad654dc 7985            jns     np_java_script+0x5463 (2ad65463)                                           ; [22] continue to next
iteration of loop
```

The disassembly within this advisory is for version 13.31.0.605 and is using the modules loaded at the following addresses.

```
Browse full module list
start    end        module name
2ad60000 2af77000   np_java_script   (export symbols)       C:\Program Files\Nitro\Pro\13\np_java_script.dll
00370000 00dd6000   NitroPDF   (export symbols)       C:\Program Files\Nitro\Pro\13\NitroPDF.exe
2b090000 2b11b000   js32u      (export symbols)       C:\Program Files\Nitro\Pro\13\js32u.dl
```

Crash Information

When first opening up the application in a debugger, set a breakpoint at the following address, and then resume execution. Afterwards, open up the provided proof-of-concept in which execution should break at the address of the breakpoint that was just set.

```
0:000> bp np_java_script+864d
0:000> g
```

At this breakpoint, the value in the `%eax` register is the private data that was allocated for the “TimeOut” object. Set the next breakpoint to the given address and continue execution.

```
0:000> r
eax=44b66ff0 ebx=44b54fb0 ecx=0000000c edx=00000000 esi=2f70ae88 edi=00000064
eip=2bf0864d esp=021fd72c ebp=021fd75c iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000206
np_java_script+0x864d:
2bf0864d 8bf0            mov     esi,eax

0:000> r @eax
eax=44b66ff0

0:000> bp np_java_script+8727
0:000> g
```

This address is where the `JSObject` using the “TimeOut” `JSClass` is instantiated. The `%eax` register is the `JSObject`, and the `%edi` register is the private data that was just allocated. In the following output, our private data is at address 0x44b66ff0 and our “TimeOut” object was instantiated at address 0xffe2540.

```
0:000> r
eax=0ffe2540 ebx=44b54fb0 ecx=00000004 edx=1a470fd8 esi=44b66ff0 edi=44b66ff0
eip=2bf08727 esp=021fd720 ebp=021fd75c iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000206
np_java_script+0x8727:
2bf08727 8bf0            mov     esi,eax

0:000> r @eax,@edi
eax=0ffe2540 edi=44b66ff0
```

Before we continue execution, we will first set breakpoints on both destructors in order to capture the private data being released the first time when traversing the array. The first breakpoint is the address of the `JSTimeOutDestructor` function which is where our private data is used, and the second breakpoint is the address of the call to `free` that is called by the handler for notification 51.

```
0:000> bp np_java_script+8b90
0:000> bp np_java_script+54a9

0:000> g
```

Resuming execution, we encounter the call to `free` at our second breakpoint. Our private data is stored in the `%esi` register and was just pushed as an argument. We can dump out the parameters to `free` to see that our private data will be released if execute this instruction. We can resume execution.

```
0:000> r
eax=00000001 ebx=00000000 ecx=f03dd875 edx=02000000 esi=44b66ff0 edi=00000000
eip=2bf054a9 esp=02fff998 ebp=02fff9ac iopl=0         nv up ei pl nz ac pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000216
np_java_script+0x54a9:
2be054a9 e86cd20500      call    np_java_script!CxImageJPG::Encode+0x79a (2bf6271a)

0:000> r @esi
esi=44b66ff0

0:000> dc @esp L2
01fff998  44b66ff0 0000000c                    .O......

0:000> g
```

We have now arrived at the `JSTimeOutDestructor` function. Our first parameter is a `JSContext`, and our second parameter should be the “TimeOut” `JSObject` which we can dump to confirm that it matches the address 0xffe2540. Set the next breakpoint which will be where our private data will be fetched from our `JSObject`.

```
0:000> r
eax=2bf08b90 ebx=2f70ae88 ecx=00000000 edx=02000000 esi=1a470fd8 edi=0ffe2540
eip=2bf08b90 esp=021ff6d0 ebp=021ff6e8 iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000206
np_java_script+0x8b90:
2bf08b90 55              push    ebp

0:000> r @edi
edi=0ffe2540

0:000> dc @esp+4 L2
021ff6d4  2f70ae88 0ffe2540                    ..p/@%..

0:000> bp np_java_script+8bfd
0:000> g
```

Now we’re at the call to `JS_GetPrivateData`. We can check our arguments to it by dumping two values from the stack. Our first parameter is the `JSContext`, and our second is the “TimeOut” `JSObject` that the private data was attached to. Use the `p` command to step over the function so we can examine the result that is returned.

```
0:000> r
eax=00000000 ebx=2f70ae88 ecx=2bfff3b4 edx=02000000 esi=1a470fd8 edi=0ffe2540
eip=2bf08bfd esp=021ff6bc ebp=021ff6cc iopl=0         nv up ei pl zr na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000246
np_java_script+0x8bfd:
2bf08bfd ff1508e7ff2b    call    dword ptr [np_java_script!CxImageJPG::Encode+0x9c788 (2bffe708)] ds:0023:2bffe708={js32u!JS_GetPrivate
(2c1342d0)}

0:000> dc @esp L2
021ff6bc  2f70ae88 0ffe2540                    ..p/@%..
```

Stepping over the function shows the address 0x44b66ff0 was returned. This corresponds to the address that was allocated for the private data earlier. Resume execution after confirming that the private data matches.

```
0:000> p
eax=44b66ff0 ebx=2f70ae88 ecx=2bfff3b4 edx=02000000 esi=1a470fd8 edi=0ffe2540
eip=2bf08c03 esp=021ff6bc ebp=021ff6cc iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000206
np_java_script+0x8c03:
2bf08c03 8bf0            mov     esi,eax

0:000> r @eax
eax=44b66ff0

0:000> g
```

After resuming execution, we can see our released private data was just dereferenced prior to the application freeing it.

```
(1798.1510): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=44b66ff0 ebx=2f70ae88 ecx=2bfff3b4 edx=02000000 esi=44b66ff0 edi=0ffe2540
eip=2bf08c0c esp=021ff6c4 ebp=021ff6cc iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00010206
np_java_script+0x8c0c:
2bf08c0c 8b06            mov     eax,dword ptr [esi]  ds:0023:44b66ff0=????????
```

In this analysis, the `np_java_script.dll` library was mapped at address 0x2bf00000.

```
0:000> lm m np_java_script
Browse full module list
start    end        module name
2bf00000 2c117000   np_java_script   (export symbols)       C:\Program Files\Nitro\Pro\13\np_java_script.dll
```

Exploit Proof of Concept

In the provided proof-of-concept, it is object 110 revision 0 that contains the javascript which triggers this vulnerability.

Mitigation

To mitigate this vulnerability, one can visit the “JavaScript” item in the preferences and click the “Disable JavaScript” checkbox. As this vulnerability depends on the execution of JavaScript, enabling this option will prevent the vulnerability from being triggered.

Timeline

2021-03-15 - Vendor Disclosure
2021-04-20 - 30 day follow up with vendor
2021-06-08 - Copies of advisories issued
2021-06-22 - Granted disclosure extension
2021-07-19 - Final disclosure extension granted
2021-10-14 - Public Release

CREDIT

Discovered by a member of Cisco Talos.

VULNERABILITY REPORTS

PREVIOUS REPORT
TALOS-2021-1265

NEXT REPORT
TALOS-2021-1313