Talos Vulnerability Report

TALOS-2021-1386

Hancom Office 2020 Hword HwordApp.dll SectorLoc heap-based buffer overflow

FEBRUARY 15, 2022

CVE NUMBER

CVE-2021-21958

Summary

A heap-based buffer overflow vulnerability exists in the Hword HwordApp.dll functionality of Hancom Office 2020 11.0.0.2353. A specially-crafted malformed file can lead to memory corruption and potential arbitrary code execution. An attacker can provide a malicious file to trigger this vulnerability.

Tested Versions

Hancom Office 2020 11.0.0.2353

Product URLs

Hancom Office 2020 - https://office.hancom.com/

CVSSv3 Score

7.8 - CVSS:3.0/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H

CWE

CWE-122 - Heap-based Buffer Overflow

Details

Hancom Office is considered one of the more popular Office suites used within South Korea.

To be able to reproduce the vulnerability, we need to turn on PageHeap for Hword.exe app and open our malicious file.

The debugger stops with the following information

```
(1294.1198): 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: 26E63B2:0
eax=7f169054 ebx=00000053 ecx=00000014 edx=00000053 esi=7f169040 edi=7f167000
eip=69c9f26d esp=011cd768 ebp=011cd788 iopl=0         nv up ei ng nz na pe cy
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00200287
MSVCR120!memcpy+0x2a:
69c9f26d f3a4            rep movs byte ptr es:[edi],byte ptr [esi]
```

As we can see a heap-based buffer overflow occured. Let's go back to a few functions before :

```
Line 1  struct_this *__thiscall read_CFB_directory_entry_data(struct_this *this, int a2, unsigned int stream_size, int StartSecLoc)
Line 2  {
Line 3    unsigned int vector_size; // edi
Line 4    void *dstBuffer; // eax
Line 5    unsigned __int16 sectorSize; // [esp-18h] [ebp-38h]
Line 6    int size_to_read; // [esp-14h] [ebp-34h]
Line 7
Line 8    sub_10BBE230(this, (_WORD *)a2, stream_size);
Line 9    this->vftable0 = &tfo_olefs::OleEntryMiniInputStream::`vftable';
Line 10   this->vftable8 = &tfo_olefs::OleEntryMiniInputStream::`vftable';
Line 11   vector_size = (stream_size >> 6) + 1;
Line 12   if ( (this->currentEnd - this->secStartPtr) >> 2 < vector_size )
Line 13   {
Line 14         if ( vector_size > 0x3FFFFFFF )
Line 15           std::_Xlength_error("vector<T> too long");
Line 16         vector_resize((struct_this_1 *)&this->secStartPtr, vector_size);
Line 17   }
Line 18   if ( get_mini_fat_sectors(this->gapC, &this->secStartPtr, StartSecLoc) )
Line 19   {
Line 20         sub_10B7CE20(*(_DWORD **)this->gapC, (char **)&this[1]);

Line 21         if ( (unsigned int)((this->secEndPtr - this->secStartPtr) >> 2) <= 1 )
Line 22           size_to_read = this->stream_size;
Line 23         else
Line 24           size_to_read = 0x40;
Line 25
Line 26         dstBuffer = operator new[](
Line 27                                     0x40u,
Line 28                                     1,
Line 29                                     "..\\..\\Common\\Engine\\TfoCore\\tfo_olefs\\tfo_olefs\\OleEntryMiniInputStream.cpp",
Line 30                                     29);
Line 32         this->dstBuffer = dstBuffer;
Line 33         this->word24 = 0;
Line 34         sectorSize = this->sectorSize;
Line 35         this->dword18 = 0;
Line 36         fread_wrapper(this->dword10, (int)dstBuffer, &this[1].vftable0, StartSecLoc, sectorSize, size_to_read);
Line 37         this->byte4 = 1;
Line 38   }
Line 39   else
Line 40   {
Line 41         this->byte4 = 0;
Line 42   }
Line 43   return this;
Line 44 }
```

The heap overflow occured after a call to `fread_wrapper` in `line 36` where the size of the bytes to read is controled by the variable `size_to_read`. The `dstBuffer` will always have a constant size equal to `0x40` bytes (`line 26`). Further analysis revealed that the structure being parsed here is `Compound File Directory Entry` [https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-cfb/60fe8611-66c3-496b-b70d-a504c94c9ace] (OLESSDirectoryEntry) located at offset : 0x18180

```
1:8180h: 01 00 43 00 6F 00 6D 00 70 00 4F 00 62 00 6A 00  ..C.o.m.p.O.b.j.
1:8190h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
1:81A0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
1:81B0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
1:81C0h: 12 00 02 00 20 00 00 00 21 00 00 00 FF FF FF FF  .... ...!...ÿÿÿÿ
1:81D0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
1:81E0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
1:81F0h: 00 00 00 E4 F2 00 00 00 54 00 00 00 00 00 00 00  ...äò...T.......
```

The `size_to_read` value was assigned at `line 22`. It equals 0x54, which corresponds to `Stream Size` located at `offset : 1:81F8h`. But why was `line 22` executed instead of `line 24`? It turned out that at the beginning, right after the call to `vector_resize` (`line 16`) both `this->secEndPtr` and `this->secStartPtr` point to the same memory space address.

Later, inside `get_mini_fat_sectors`, the address in `this->secEndPtr` is increased by 4 each time a new sector is added to the vector, related with a parsed `directory entry` until the `FAT_ENDOFCHAIN` marker is found. In our case only one sector (`0xF2` is added at the beginning) because: Starting Sector Location (4 bytes) offset : 1:81F8h - F2 00 00 00 MiniFat entries in memory + (0xF2 * 4) == ffffffe (FAT_ENDOFCHAIN) - possible offset in the file 0x123C8

Which in consequence give us a difference of 4 between `this->secEndPtr` and `this->secStartPtr`. Next, divided by 4 (`line 21 ( >> 2)`) this passes the constraint and to `size_to_read` is assigned `0x54` which leads to a heap-based buffer overflow.

Crash Information

```
0:000> !analyze -v
*******************************************************************************
*                                                                             *
*                        Exception Analysis                                   *
*                                                                             *
*******************************************************************************

*** WARNING: Unable to verify checksum for PresentationFramework.ni.dll
*** WARNING: Unable to verify checksum for WindowsBase.ni.dll
Failed to request MethodData, not in JIT code range
MethodDesc:   2f0380f0
Method Name:  HwordAppModule.HwordDocProxy.Open(HwordAppModule.HwordFrameProxy, System.String, System.String, Int32, Int32)
Class:        2ea08dfc
MethodTable:  2f0382a0
mdToken:      060002dd
Module:       2df82848
IsJitted:     yes
CodeAddr:     00344368
Transparency: Safe critical
Unable to load image c:\Program Files (x86)\Hnc\Office 2020\HOffice110\Bin\HwordAppModule.dll, Win32 error 0n2
MethodDesc:   2df8e340
Method Name:  Hword.HwordFrame._OpenDocument(System.String, HwordDefine.OpenAttr, Boolean, Boolean, Boolean, System.String)
Class:        2a39ace0
MethodTable:  2df8e8a4
mdToken:      06001258
Module:       161e4044
IsJitted:     yes
CodeAddr:     003431e8
Transparency: Critical
MethodDesc:   2df8e334
Method Name:  Hword.HwordFrame.OpenDocument(System.String, HwordDefine.OpenType, HwordDefine.OpenAttr, Boolean)
Class:        2a39ace0
MethodTable:  2df8e8a4
mdToken:      06001257
Module:       161e4044
IsJitted:     yes
CodeAddr:     00341be8
Transparency: Critical
MethodDesc:   2df8cee8
Method Name:  Hword.HwordApp.ProcessShellCommand(Hnc.Static.CommandParser)
Class:        2a391574
MethodTable:  2df8d058
mdToken:      0600116e
Module:       161e4044
IsJitted:     yes
CodeAddr:     2f06ee40
Transparency: Critical
MethodDesc:   161eb214
Method Name:  Hword.HwordAppMain.StartApp(Hnc.Static.CommandParser)
Class:        1f837b24
MethodTable:  161eb290
mdToken:      0600075b
Module:       161e4044
IsJitted:     yes
CodeAddr:     2e7410c8
Transparency: Critical
MethodDesc:   161eb1d8
Method Name:  Hword.HwordAppMain.OnApplicationStartup(System.Object, System.Windows.StartupEventArgs)
Class:        1f837b24
MethodTable:  161eb290
mdToken:      06000756
Module:       161e4044
IsJitted:     yes
CodeAddr:     1d75abb8
Transparency: Critical

KEY_VALUES_STRING: 1

    Key  : AV.Fault
    Value: Read

    Key  : Analysis.CPU.mSec
    Value: 38139

    Key  : Analysis.DebugAnalysisManager
    Value: Create

    Key  : Analysis.Elapsed.mSec
    Value: 268034

    Key  : Analysis.Init.CPU.mSec
    Value: 285562

    Key  : Analysis.Init.Elapsed.mSec
    Value: 86923456

    Key  : Analysis.Memory.CommitPeak.Mb
    Value: 1927

    Key  : CLR.BuiltBy
    Value: NET48REL1LAST_C

    Key  : CLR.Engine
    Value: CLR

    Key  : CLR.Version
    Value: 4.8.4400.0

    Key  : Timeline.OS.Boot.DeltaSec
    Value: 420319

    Key  : WER.Process.Version
    Value: 11.0.0.2353


NTGLOBALFLAG:  2000000

PROCESS_BAM_CURRENT_THROTTLED: 0

PROCESS_BAM_PREVIOUS_THROTTLED: 0

APPLICATION_VERIFIER_FLAGS:  0
```

```
APPLICATION_VERIFIER_LOADED: 1

EXCEPTION_RECORD:  (.exr -1)
ExceptionAddress: 69c9f26d (MSVCR120!memcpy+0x0000002a)
   ExceptionCode: c0000005 (Access violation)
  ExceptionFlags: 00000000
NumberParameters: 2
   Parameter[0]: 00000000
   Parameter[1]: 7f167000
Attempt to read from address 7f167000

FAULTING_THREAD:  00001198

PROCESS_NAME:  Hword.exe

READ_ADDRESS:  7f167000

ERROR_CODE: (NTSTATUS) 0xc0000005 - The instruction at 0x%p referenced memory at 0x%p. The memory could not be %s.

EXCEPTION_CODE_STR:  c0000005

EXCEPTION_PARAMETER1:  00000000

EXCEPTION_PARAMETER2:  7f167000

STACK_TEXT:
011cd76c 69c9f71f     7f166fc1 7f169001 00000053 MSVCR120!memcpy+0x2a
011cd788 69ca744f     7f166fc1 fffffffe 7f169001 MSVCR120!memcpy_s+0x3e
011cd7bc 69ca74bd     7f166fc0 ffffffff 00000001 MSVCR120!_fread_nolock_s+0xd5
011cd804 69ca7512     7f166fc0 ffffffff 00000001 MSVCR120!fread_s+0x6b
011cd820 66112be4     7f166fc0 00000001 00000054 MSVCR120!fread+0x16
WARNING: Stack unwind information not available. Following frames may be wrong.
011cd838 66af18e9     7f166fc0 00000054 000000f2 HwordApp!HwordDeletePropertyArray+0x188a64
011cd854 66af15ad     7f162f28 7f166fc0 787f6ff4 HwordApp!HwordDeletePropertyArray+0xb67769
011cd8a4 66abeb4b     51520f80 00000054 000000f2 HwordApp!HwordDeletePropertyArray+0xb6742d
011cd8d8 664b7eda     787f6fc0 2e77cd93 6269ef90 HwordApp!HwordDeletePropertyArray+0xb349cb
011cd90c 664b7fc2     51520f80 807f6fa8 011cd974 HwordApp!HwordDeletePropertyArray+0x52dd5a
011cd94c 6643d959     51520f80 807f0f90 011cd974 HwordApp!HwordDeletePropertyArray+0x52de42
011cda28 66443a47     51520f80 00001a00 502e2f48 HwordApp!HwordDeletePropertyArray+0x4b37d9
011cdcb0 66446aa3     37a0adf8 82e3ef90 62e0ffa8 HwordApp!HwordDeletePropertyArray+0x4b98c7
011cdf08 66446289     37a0adf8 82e3ef90 62e0ffa8 HwordApp!HwordDeletePropertyArray+0x4bc923
011ce4c8 6645474a     37a0adf8 82e3ef90 7a120fc0 HwordApp!HwordDeletePropertyArray+0x4bc109
011ce574 66441a45     7a120fc0 82e3ef90 62748a78 HwordApp!HwordDeletePropertyArray+0x4ca5ca
011ce5c8 6677fe29     53706fd8 00000000 53706fd8 HwordApp!HwordDeletePropertyArray+0x4b78c5
011ce5fc 660eceb0     2800cfd8 011ce65c 27d2cfdc HwordApp!HwordDeletePropertyArray+0x7f5ca9
011ce624 660ea988     53702fe8 011ce65c 27d2cfdc HwordApp!HwordDeletePropertyArray+0x162d30
011ce6f8 65fa2d43     00000000 83348fb0 011ce828 HwordApp!HwordDeletePropertyArray+0x160808
011cea78 00344404     4ecc8fa0 6823ef6c 69c75094 HwordApp!HwordDeletePropertyArray+0x18bc3
011ceae0 003434c9     00000000 00000000 200c1228 0x344404
011ceba8 00341f67     2014a1e4 00000000 00000000 0x3434c9
011cebf4 2f06f1b9     00000000 00000000 00000001 0x341f67
011cec54 2e7412bf     2016699c 201669ac 20149a84 0x2f06f1b9
011cec68 1d75ac14     2016699c 201662c8 20166594 0x2e7412bf
011cec7c 6e0cfd43     2016699c 00000000 20149e8c 0x1d75ac14
011cec94 6e095ef2     20166458 00000000 011cecbc PresentationFramework_ni+0x2ffd43
011ceca4 6feeee42     00000001 20166458 200d4628 PresentationFramework_ni+0x2c5ef2
011cecbc 6feeed85     00000001 00000000 00000000 WindowsBase_ni+0xdee42
011cecf8 6fef10cd     00000000 00000001 00000000 WindowsBase_ni+0xded85
011ced40 6feef56f     201664e4 73e68604 200c3b4c WindowsBase_ni+0xe10cd
011cedac 73e68537     00000000 20149b58 00000000 WindowsBase_ni+0xdf56f
011cedc0 73e684f4     00000000 20149b58 00000000 mscorlib_ni!System.Threading.ExecutionContext.Run(System.Threading.ExecutionContext,
System.Threading.ContextCallback, System.Object, Boolean)$##6003AEF+0x17
011ceddc 6fef0f83     20149b58 201664c4 00000000 mscorlib_ni!System.Threading.ExecutionContext.Run(System.Threading.ExecutionContext,
System.Threading.ContextCallback, System.Object)$##6003AEE+0x44
011cee0c 6fef0d80     20149b58 00000000 00000000 WindowsBase_ni+0xe0f83
011cee44 6feed346     00000000 200d46fc 00000000 WindowsBase_ni+0xe0d80
011cee84 6feec57c     00000000 00000000 200d4380 WindowsBase_ni+0xdd346
011ceec0 6feee661     200d52f0 00000000 00000000 WindowsBase_ni+0xdc57c
011ceefc 6feee94c     200d4f4c 00000000 00000000 WindowsBase_ni+0xde661
011cef1c 6feeee42     00000001 200c3b4c 200d4628 WindowsBase_ni+0xde94c
011cef34 6feeed85     00000001 200d4f34 00000000 WindowsBase_ni+0xdee42
011cef70 6feecf62     00000000 00000001 200d4f34 WindowsBase_ni+0xded85
011cefc8 6feee4b4     00000001 200d4f34 200d4f14 WindowsBase_ni+0xdcf62
011cf010 161ed16e     00000000 00000000 0000c1c0 WindowsBase_ni+0xde4b4
011cf044 775636db     0017052e 0000c1c0 00000000 0x161ed16e
011cf070 7755a66a     26ff9ff6 0017052e 0000c1c0 USER32!_InternalCallWinProc+0x2b
011cf154 775583da     26ff9ff6 00000000 0000c1c0 USER32!UserCallWinProcCheckWow+0x33a
011cf1c8 775581a0     011cf220 011cf210 6ff074f1 USER32!DispatchMessageWorker+0x22a
011cf1d4 6ff074f1     011cf220 29163fd7 74f5fda4 USER32!DispatchMessageW+0x10
011cf210 6feeb3d7     201475ec 200d4380 0017052e WindowsBase_ni+0xf74f1
011cf258 6feeb319     20166978 011cf274 6e095ebc WindowsBase_ni+0xdb3d7
011cf264 6e095ebc     20149ac4 00000000 011cf294 WindowsBase_ni+0xdb319
011cf274 6e095a7a     6dddd384 00000000 201662c8 PresentationFramework_ni+0x2c5ebc
011cf294 6e09586e     20149a84 201662c8 011cf2dc PresentationFramework_ni+0x2c5a7a
011cf2dc 1d75b67a     200c1fd4 200c6a00 200c1fec PresentationFramework_ni+0x2c586e
011cf2f0 1d750ede     200c67f4 011cf3b0 00000000 0x1d75b67a
011cf308 74f5f066     1f827ac0 011cf368 74f6230a 0x1d750ede
011cf314 74f6230a     011cf3b0 011cf358 75052440 clr!CallDescrWorkerInternal+0x34
011cf368 74f685eb     00000000 200c233c 011cf3c4 clr!CallDescrWorkerWithHandler+0x6b
011cf3dc 7510b28b     011cf4b8 3102b858 161eb1a8 clr!MethodDescCallSite::CallTargetWorker+0x16a
011cf500 7510b96a     011cf544 00000000 3102ba34 clr!RunMain+0x1b3
011cf76c 7510b897     00000000 3102b108 00d90000 clr!Assembly::ExecuteMainMethod+0xf7
011cfc50 7510ba18     3102b1f0 00000000 00000000 clr!SystemDomain::ExecuteMainMethod+0x5ef
011cfca8 7510bb3e     3102b1b0 00000000 75107420 clr!ExecuteEXE+0x4c
011cfce8 75107445     3102b07c 00000000 75107420 clr!_CorExeMainInternal+0xdc
011cfd24 7573fa84     7c0c16f9 757d4330 7573fa20 clr!_CorExeMain+0x4d
011cfd5c 757ce81e     757d4330 75730000 011cfd84 mscoreei!_CorExeMain+0xd6
011cfd6c 757d4338     757d4330 7682fa29 01354000 MSCOREE!ShellShim__CorExeMain+0x9e
011cfd74 7682fa29     01354000 7682fa10 011cfde0 MSCOREE!_CorExeMain_Exported+0x8
011cfd84 779d7a9e     01354000 09867502 00000000 KERNEL32!BaseThreadInitThunk+0x19
011cfde0 779d7a6e     ffffffff 779f8a41 00000000 ntdll!__RtlUserThreadStart+0x2f
011cfdf0 00000000     757d4330 01354000 00000000 ntdll!_RtlUserThreadStart+0x1b


STACK_COMMAND:  ~0s ; .cxr ; kb

FAULTING_SOURCE_LINE:  f:\dd\vctools\crt\crtw32\string\i386\memcpy.asm

FAULTING_SOURCE_FILE:  f:\dd\vctools\crt\crtw32\string\i386\memcpy.asm

FAULTING_SOURCE_LINE_NUMBER:  188

FAULTING_SOURCE_CODE:
No source found for 'f:\dd\vctools\crt\crtw32\string\i386\memcpy.asm'


SYMBOL_NAME:  MSVCR120!memcpy+2a
```

```
MODULE_NAME: MSVCR120

IMAGE_NAME:  MSVCR120.dll

FAILURE_BUCKET_ID:  INVALID_POINTER_READ_STRING_DEREFERENCE_AVRF_c0000005_MSVCR120.dll!memcpy

OSPLATFORM_TYPE:  x86

OSNAME:  Windows 8

IMAGE_VERSION:  12.0.40649.5

FAILURE_ID_HASH:  {9105dc67-e4c3-f9b0-d352-023f957ac60d}

Followup:     MachineOwner
---------
```

Timeline

2021-10-19 - Vendor Disclosure
2022-02-15 - Public Release

CREDIT

Discovered by Marcin 'Icewall' Noga of Cisco Talos.

VULNERABILITY REPORTS

PREVIOUS REPORT TALOS-2021-1387

NEXT REPORT TALOS-2021-1393