Windows Error Reporting Service CWerService::SvcCollectMemoryInfo Race Condition Arbitrary File Deletion Elevation of Privilege Vulnerability

# Environment

Microsoft Windows Version 10.0.18362.719 64-bits

# Description

There is a race condition vulnerability in wersvc.dll! CWerService::SvcCollectMemoryInfo, which can leads to delete arbitrary file with system privilege.

# Root Cause Analysis

In SvcCollectMemoryInfo:

\_\_int64 \_\_fastcall CWerService::SvcCollectMemoryInfo(CWerService \*this, struct \_WERSVC\_MSG \*a2, struct \_WERSVC\_MSG \*a3)

{

void \*hTempFile; // [sp+40h] [bp-448h]@2

…

if ( v4 < 0

|| (hTempFile = (void \*)-1, v4 = UtilGetTempFile(&hTempFile, &v9, L".csv", &FileName, v6, 0i64, 0, 0), v4 < 0)

|| ((**v4 = UtilCollectMemoryInfoHandle(hTempFile), v4 < 0) ? DeleteFileW(&FileName)** : (v4 = StringCchCopyW(

(unsigned \_\_int16 \*)v3

+ 24,

0x104ui64,

&FileName),

\*((\_DWORD \*)v3 + 11) = 0,

\*((\_DWORD \*)v3 + 10) = -268238848),

v4 < 0) )

{

\*((\_DWORD \*)v3 + 10) = 0xF0030001;

\*((\_DWORD \*)v3 + 11) = (unsigned \_\_int16)v4 | 0x80070000;

}

return (unsigned int)v4;

}

The function first creates a temporary file under WER temp folder (%ProgramData%\Microsoft\Windows\WER\TEMP), the temp file name has the format “WERXXXX.tmp.csv”.

The function UtilGetTempFile is responds for creating the temp file and ensure the file is safe (no symbolic link, reparse point, etc…), then returns a file handle.

As long as the file handle is not closed, it is safe to operate on the file because UtilGetTempFile already guarantees the file path is secure.

Then the function calls UtilCollectMemoryInfoHandle to write system memory info to the file. The problem here is, if UtilCollectMemoryInfoHandle fails, it will call DeleteFileW to delete the temporary file:

**v4 = UtilCollectMemoryInfoHandle(hTempFile), v4 < 0) ? DeleteFileW(&FileName)**

And at the point when DeleteFileW is being called, the file handle to the temporary file has already been closed, so we can change the file path to a symbolic link combined with junction to delete arbitrary file on the system with system privilege.

To make UtilCollectMemoryInfoHandle fail, we will exhaust the free space on the disk, so when UtilCollectMemoryInfoHandle tries to write to the temporary file, it will fail due to insufficient disk space.

We have developed a reliable poc to win the race in a fairly short time.

# The PoC Exploit:

The poc exploit works like this:

1. Try to exhaust the free disk space:
   1. Create a folder named “storage” under %ProgramData%\Microsoft\Windows\WER\ReportQueue
   2. Query the free space of the C: driver by calling GetDiskFreeSpaceExW, create a large file named “freespace” whose size is the FreeBytesAvailableToCaller returned by GetDiskFreeSpaceExW.
   3. Keeps creating small file of 32KB under the storage folder, until fails due to no disk space
2. Race in a loop
   1. Call SvcCollectMemoryInfo in a loop until it fails for 3 times in series.
   2. Start a race thread, the race thread will monitor WER temp folder, if it finds that a new temporary file is created under the temp folder whose name is WER1234.tmp.csv for example, then it will

2.2.1 Continuously try to delete the temp file until success (successfully deleting the file means the temporary file handle created by UtilGetTempFile has been closed)

2.2.2 Then mount the WER temp folder to “\RPC Control”, and create a symbolic from “\RPC Control\WER1234.tmp.csv” to the target file we want to delete. So if the DeleteFileW is called after we have setup the mount point and symbolic link, it will delete the target file.

* 1. The main thread calls SvcCollectMemoryInfo and let the child thread to race
  2. After SvcCollectMemoryInfo call in main thread returns, check if the race is success, if failed, go to 2.1 to race again

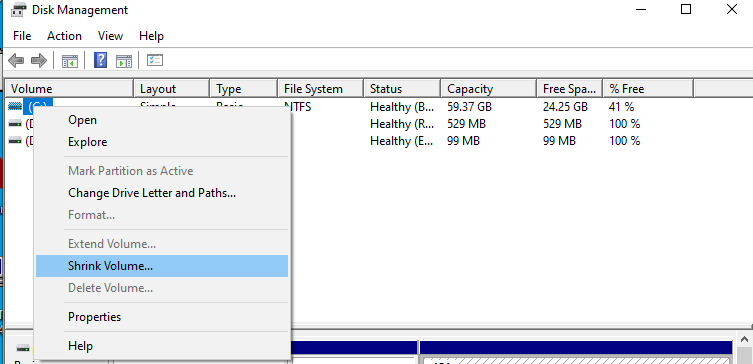
# How to reproduce:

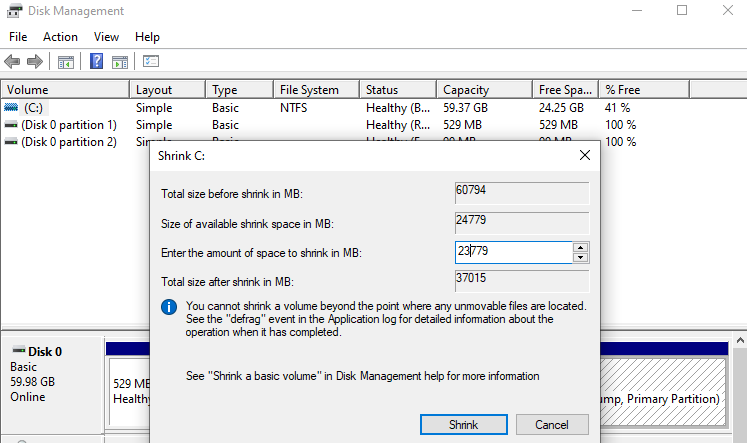
First, we recommend reproducing the poc in a virtual machine, because the poc will exhaust all free disk space of the system driver C.

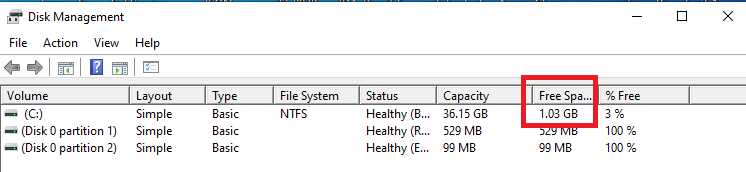
1. Prepare a virtual machine

We prefer to use a VM with small free disk space, it is not a must but will let the poc finish faster because it takes less time to exhaust the disk free space.

Also, it would be better to shrink the free space of the C driver to let is has less free space (e.g. 1 G free space) to let the poc run faster (not a must but better to do this), below picture demonstrates the setting:







You can see, in the above example, the original free space of C is 24779 MB, we shrink 23779 MB on it, so the left free space after the shrink is about 1G.

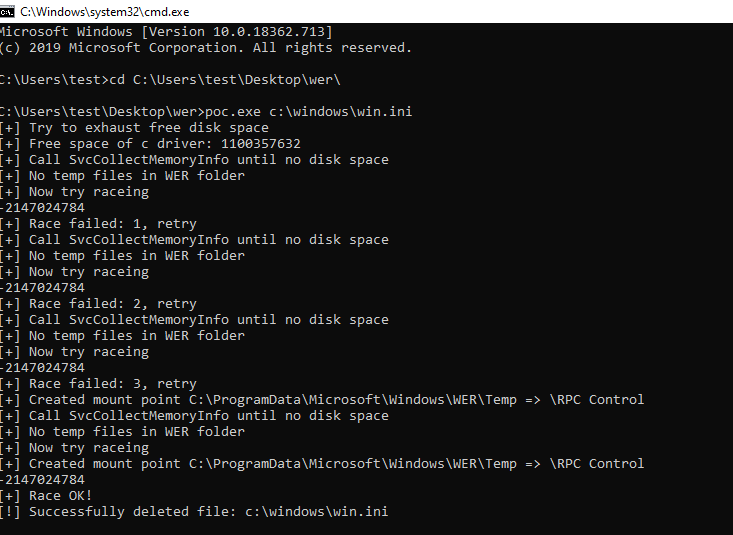
1. Execute

poc.exe target\_file\_to\_delete

For example,

poc.exe c:\windows\win.ini

the exploit will continuous to race the vulnerability, if it successes, it will have output like this:



Upon success, you can check whether the target file is really deleted.