CVE-2018-6318

DLL Hijacking in Sophos Tester Tool

By

Valthek

The Sophos company tool called Sophos Tester tool in its versions 3.2.0.7 and 3.2.0.12 (possibly in some previous or later that I could not obtain or check also keep in mind the exploit).

Version 3.2.0.7 has a compilation date of May 29, 2017 and 3.2.0.12 of October 10, 2017.

Both versions can be found in VirusTotal and possibly in other more public sites over the Internet.

3.2.0.7 -> https://www.virustotal.com/#/file/a739c3b087e04bb5beddf08ce9a056b1af989e4b4d1add8eea290cbec475da0f/detection

3.2.0.12 -> https://www.virustotal.com/#/file/6aafa205568e9ce6ac6d56ebf55beb254e835691c66f9b794d3cc860166274f4/detection

In its first execution, the tool creates a signed driver (with a valid signature at the current date for all operating systems) and a dynamic library in the "System32" directory, the library or in the WoW folder in the case of a 64bit operating system with a 32bit tool.

The controller has the name "tester86.sys" if the operating system is 32 bits or "tester64.sys" in the case of 64 bits. The dynamic library has the name "tester86.dll".

While the driver and library are encrypted in the tool, they can be obtained simply by running and installing the application and get them from the folders.

- ## USE OF THE TOOL

The tool is used to test existing exploits, emulated ransomware, code integrity, etc.

For this you can use the application itself or another application installed on the system, for example, a PDF viewer.

After carrying out the relevant verification, the application returns whether the test could be performed or not.

- ## CVE-2018-6318

The controller loads the dynamic library that is responsible for performing the test chosen by the user by searching the library "ntdll.dll" in the newly loaded process and after obtaining it install a hook in a function to execute a shellcode that the controller has previously created in the memory of the application.

The shellcode has the function to desistalar the hook, and to load by means of "LdrLoadDll" the dynamic library "tester86.dll" from the directory of "System32".

After that, it returns control to the Windows loader.

The vulnerability is present when the controller does not have any type of verification that the library that exists in the system directory is the official one or the shellcode checks anything of it.

Knowing this and with admin permissions you can change the library for another library with

the same malicious name, and when a user runs the application, although the application test would give an error, the malicious DLL would have been loaded under the context of the user and the application used to make the test.

```
000A0100   90              nop
000A0101   FF35 00000A00   push    dword ptr [A0000]                ntdll.ZwTestAlert
000A0107   60              pushad
000A0108   9C              pushfd
000A0109   68 40000A00     push    0A0040
000A010E   68 40000000     push    40
000A0113   68 38000A00     push    0A0038
000A0118   68 30000A00     push    0A0030
000A011D   68 FFFFFFFF     push    -1
000A0122   FF15 10000A00   call    [A0010]                          ntdll.ZwProtectVirtualMemory
000A0128   BE 28000A00     mov     esi, 0A0028
000A012D   8B3D 00000A00   mov     edi, [A0000]                     ntdll.ZwTestAlert
000A0133   B9 06000000     mov     ecx, 6
000A0138   FC              cld
000A0139   F3:A4           rep     movs byte ptr es:[edi], byte ptr [esi]
000A013B   68 40000A00     push    0A0040
000A0140   68 20000000     push    20
000A0145   68 38000A00     push    0A0038
000A014A   68 30000A00     push    0A0030
000A014F   68 FFFFFFFF     push    -1
000A0154   FF15 10000A00   call    [A0010]                          ntdll.ZwProtectVirtualMemory
000A015A   68 50000A00     push    0A0050
000A015F   68 58000A00     push    0A0058                           UNICODE "@@`",LF,"C:\WINDOWS\System32\tester86.dll"
000A0164   6A 00           push    0
000A0166   6A 00           push    0
000A0168   FF15 08000A00   call    [A0008]                          ntdll.LdrLoadDll
000A016E   9D              popfd
000A016F   61              popad
000A0170   C3              retn
```

- **POSSIBLE USES OF THE EXPLOIT**

Any action that an attacker wants to make by changing the DLL could be done or even modify a small part of the code of the original DLL, making the use of the application fail and executing a small malicious code.

With this PDF a simple POC is provided but it shows how in a system with the running application that attack can be made.