Story of a dumb patch

Author:
Cesar Cerrudo
(cesar>.at.<argeniss>.dot.<com)

Abstract:

This paper is an advisory but mostly it describes a mistake made by Microsoft on patch MS05-018 where Microsoft failed to properly fix a vulnerability having to release a new patch MS05-049. Hopefully this paper will open the eyes to software vendors to not repeat this kind of mistakes.

Introduction:

On April 12, 2005 Microsoft released a patch to address a stack overflow vulnerability on Client/Server Runtime Server Subsystem (CSRSS), while I was reverse engineering the bug in order to build an exploit I found that the bug was improperly patched and it could be still exploited. The problem was that Microsoft didn't patch the vulnerable function they just added some validation code before the call to the vulnerable function, but what Microsoft missed was that the vulnerable function can be reached from different paths and the validation code was added on just one of them.

Vulnerability details:

Console windows are managed by csrss.exe process, specifically by code inside winsrv.dll, this module handles the creation of console windows and the properties associated with the windows such as size, font, color, etc. Console windows properties can be set by selecting Properties on window system menu, setting the values you want and then saving the changes. Lets see how this is done:

When you select Properties item on console window menu:
1. A window message is sent and it is handled by csrss.exe by ConsoleWindowProc() function of winsrv.dll
2. Csrss.exe creates a shared section (for more info on shared sections see 1 on References section)
3. Shared section is mapped inside csrss.exe
4. The shared section is filled with current console window properties.
5. The shared section is unmapped from csrss.exe
6. Csrss.exe duplicates shared section handle inside console window process.
7. Csrss.exe creates a new thread inside console window process.
8. The new thread loads console.dll inside console window process.
9. Console.dll maps the section and creates the properties window filling it with the properties that are read from the section.

When you save the properties changes:
1. Console window process creates a shared section and maps it.
2. The properties values are copied to the mapped shared section.
3. The shared section is unmapped.
4. A window message is sent with the handle of the shared section.
   (All the previous steps are done by WriteStateValues() function of console.dll)
5. The window message is handled by csrss.exe by ConsoleWindowProc() function of winsrv.dll
6. Csrss.exe duplicates the handle of the shared section to have a local handle and maps the shared section.
7. Csrss.exe reads the properties values from the shared section.
8. Csrss.exe sets the new properties values to the console window.

The vulnerability is on step 7 when csrss.exe reads the new properties values, these values are read from the next structure stored on the shared section:

```
typedef struct _CONSOLE_STATE_INFO
{
  /* 0x00 */  DWORD cbSize;
  /* 0x04 */  COORD ScreenBufferSize;
  /* 0x08 */  COORD WindowSize;
  /* 0x0c */  POINT WindowPosition;
  /* 0x14 */  COORD FontSize;
  /* 0x18 */  DWORD FontFamily;
  /* 0x1c */  DWORD FontWeight;
  /* 0x20 */  WCHAR FaceName[32];
  /* 0x60 */  DWORD CursorSize;
  /* 0x64 */  BOOL  FullScreen;
  /* 0x68 */  BOOL  QuickEdit;
  /* 0x6c */  BOOL  DefaultWindowPos;
  /* 0x70 */  BOOL  InsertMode;
  /* 0x74 */  WORD  ScreenColors;
  /* 0x76 */  WORD  PopupColors;
  /* 0x78 */  BOOL  HistoryNoDup;
```

```
    /* 0x7c */  DWORD HistoryBufferSize;
    /* 0x80 */  DWORD NumberOfHistoryBuffers;
    /* 0x84 */  COLORREF ColorTable[16];
    /* 0xc4 */  DWORD CodePage;
    /* 0xc8 */  DWORD hwnd;
    /* 0xcc */  WCHAR ConsoleTitle[2];
  } CONSOLE_STATE_INFO, *PCONSOLE_STATE_INFO;
```

The problem occurs when Unicode string FaceName is copied to a buffer without validating the size so the buffer is overflowed, this is done by DoFontEnum() function of winsrv.dll.

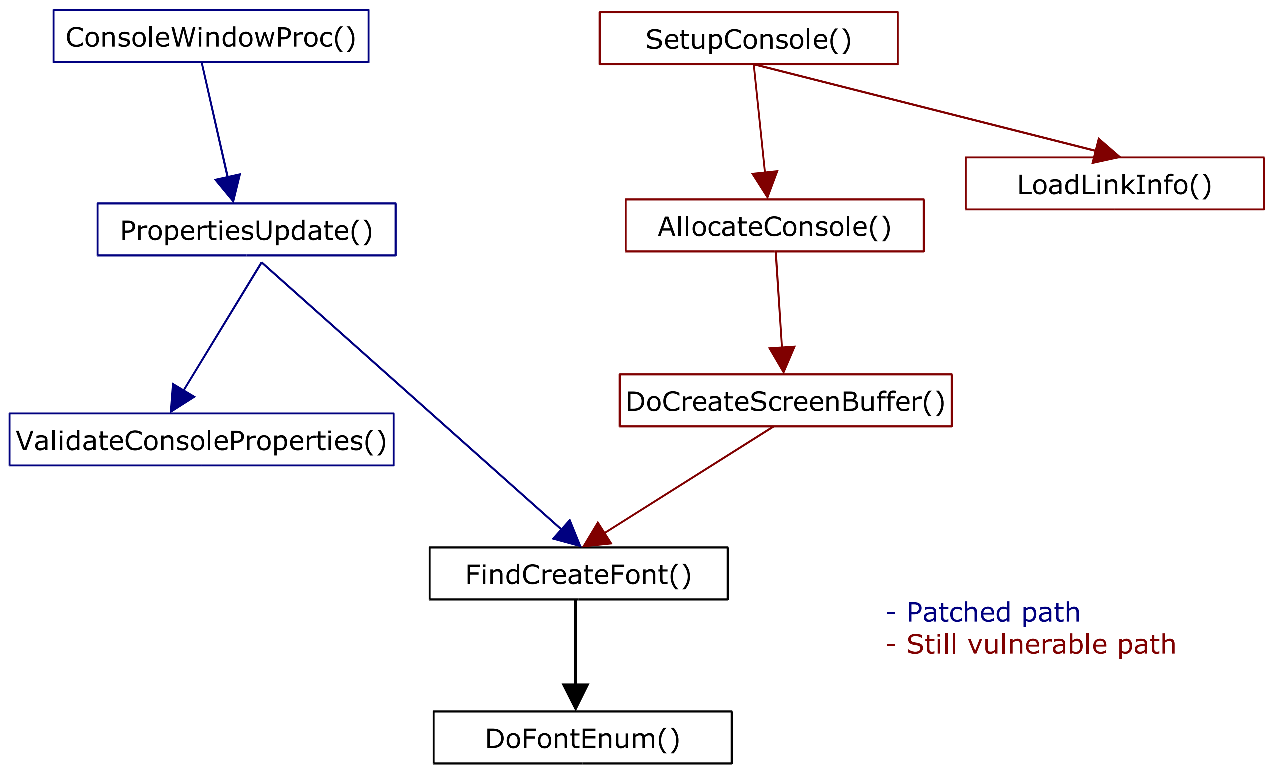
Snip of vulnerable code of Windows 2000 sp4:

```
push    [ebp+arg_4]          //pointer to  FaceName Unicode string.
lea     eax, [ebp+var_70.lfFaceName]
push    eax                  //pointer to buffer were Unicode string will be copied.
call    wcscpy               //crash!! no length check
```

The Dumb Fix:

In order to fix the vulnerability Microsoft decided to not do it on DoFontEnum() function but to add the validation function ValidateConsoleProperties() before DoFontEnum() is called, I think Microsoft did this because none of the data on the structure were validated which also caused a local DOS when setting some values on the structure, so the validation function also validates other data of the structure besides the Unicode string.
The problem with this fix is that Microsoft only patched one path to the vulnerable function but they forgot to do proper research to identify all the paths since these properties can be set not only from the console properties window.
When a console application is ran properties are read from registry or if they were ran by a shortcut the properties are read from the shortcut file. It seems that it's properly validated when csrss.exe reads the properties from the registry but it's not validated when csrss.exe reads the properties from a shortcut file, then the vulnerability it's still exploitable by setting a long string as a font name property on a console application shortcut file.
When a console application is ran by a shortcut, csrss.exe reads the shortcut file in order to set the console window properties, all this is done by functions of winsrv.dll such as SetupConsole (), LoadLinkInfo(), AllocateConsole(), DoCreateScreenBuffer(), etc.

In the next graphic you can see how the vulnerable function DoFontEnum() can be reached from two different paths and one of them hasn't any validation:



The Final Fix:

On October 11, 2005 Microsoft released MS05-049 patch which correctly fixed the bug, finally Microsoft decided to remove wcscpy() from vulnerable function and to copy only the proper amount of bytes to not overflow the buffer, this fix is good but Microsoft should have done it in first patch.

Snip of the new code of Windows 2000 sp4:

```
push    [ebp+arg_4]           //pointer to  FaceName Unicode string.
lea     eax, [ebp+var_74.lfFaceName]
push    20h                   //very important: size !!
push    eax                   //pointer to buffer were Unicode string will be copied.
call    StringCopyWorkerW    //this new function will only copy the specified bytes
```

Bonus:

While I was testing this bug with a console application shortcut file crafted with a long string as a font name property, I was on a Windows 2000 system, when I tried to look at the properties of the .lnk file explorer.exe crashed and quickly debugging I realized that Windows Explorer had a buffer overflow vulnerability. So the same .lnk file that crashed csrss.exe also crashed explorer.exe, this was really nice two vulnerabilities at once :).

Important:

This vulnerability is caused by a specially crafted .lnk file and for regular users this file type is not familiar to them since Windows hide the extension when browsing with Windows Explorer, but when this file is delivered via web or email the extension is showed (yes my grandma can think it is a picture if the email says that!), this could be used by malware writers to create a new worm, etc. So if you haven't did it, start blocking .lnk files at the gateway and Antivirus vendors should start updating their signatures.
The vulnerability on CSRSS (Shell Vulnerability-CAN-2005-2122) affects all Windows versions and the Windows Explorer vulnerability (Shell Vulnerability-CAN-2005-2118) affects Windows 2000 all service packs and Windows XP prior sp2.

Conclusion:

As you have seen Microsoft did what you never have to do, Microsoft failed to build a good fix thus losing a lot of money and time, also Microsoft make users to lost time (maybe also money if you think time=money) since users have to patch again what they have already patched. I personally have seen Microsoft improvements on all aspects of security over the last years, but I think that Microsoft still needs some fine tunning on the patching process in order to avoid this kind of mistakes. I also must say that Microsoft is 1000% better than Oracle at handling and patching vulnerabilities.

The moral of the story: always patch the vulnerable function or at least patch all paths to vulnerable function.

References:

- Vulnerabilities in Windows Kernel Could Allow Elevation of Privilege and Denial of Service
http://www.microsoft.com/technet/security/Bulletin/MS05-018.mspx

- Vulnerabilities in Windows Shell Could Allow Remote Code Execution
http://www.microsoft.com/technet/security/Bulletin/MS05-049.mspx

- Microsoft Windows CSRSS.EXE Stack Overflow Vulnerability
http://www.idefense.com/application/poi/display?id=230&type=vulnerabilities

- [1] Hacking Windows Internals
http://www.argeniss.com/research/hackwininter.zip

Thanks:

To Pedram Amini for some bug tips.

About Argeniss

Argeniss is an information security company specialized on application security, we offer services such as vulnerability information, software auditing, penetration testing, training and also we offer exploits for widely deployed software.

Contact us

Corrientes 240
Parana, Entre Rios
Argentina

E-mail: info>.at.<argeniss>.dot.<com

Tel: +54-343-4231076
Fax: 1-801-4545614