VUPEN security

VUPEN Security – Private Exploits & PoC Service

In-Depth Analysis of IBM Lotus Notes File Viewer WordPerfect Buffer Overflow Vulnerability (CVE-2008-4564)

Table of Contents

This Binary Analysis and Exploit or Proof-of-concept codes are under the copyrights of VUPEN Security. Copying or reproducing the document, exploit or proof-of-concept codes is prohibited, unless such reproduction or redistribution is permitted by the VUPEN Exploits & PoCs Service license agreement. Use of the Binary Analysis, Exploit or Proof-of-concept codes is subject to the VUPEN Exploits & PoCs Service license terms.

Introduction

A vulnerability exists in Autonomy KeyView, in the way it processes specially crafted WordPerfect files (WPD). This module is used in many applications, like Lotus Notes or Symantec Mail Security. Exploitation of this vulnerability could lead to arbitrary code execution under the context of the vulnerable application.

Tested Versions

The vulnerability was analysed on Windows XP SP2 with a default installation of Lotus Notes. The vulnerable code is located in wp6sr.dll version 8.0.0.7213.

Fixed Versions

A fix for this vulnerability is available on demand to IBM.

Technical Details

The bug is triggered when a WordPerfect document contains an overly long font name. Fonts are encoded in Unicode and are handled in sub_7C5DE4:

At this point, edi points to the beginning of the file, si is set to the length of the font and eax is a loop counter to browse the font:

```
.text:007C5ED0 loc_7C5ED0:
.text:007C5ED0          movsx   eax, word ptr [ebp+var_10]  //get the counter
.text:007C5ED4          mov     cl, [eax+edi+0C94h]         //get a character
.text:007C5EDB          add     eax, edi                    //set eax to point to the
current character
.text:007C5EDD          mov     al, [eax+0C95h]
.text:007C5EE3          test    al, al
.text:007C5EE5          jnz     short loc_7C5EF8      //check if the character does not
                                                      //belong to the range 00h-FFh
.text:007C5EE7          test    cl, cl
.text:007C5EE9             jz      short loc_7C5F36           //check for a null
character
.text:007C5EEB          movsx   edx, word ptr [ebp+arg_0]  //edx is a loop counter
.text:007C5EEF          inc     [ebp+arg_0]                 //increment the counter
.text:007C5EF2          mov     [ebp+edx+String], cl        //write the character on
the stack
.text:007C5EF6          jmp     short loc_7C5F36
…
.text:007C5F36 loc_7C5F36:
.text:007C5F36
.text:007C5F36          add     [ebp+var_10], 2             //increment the Unicode
string
.text:007C5F3A          test    cl, cl
.text:007C5F3C             jnz     short loc_7C5F42           //check again for a
character > FFh
.text:007C5F3E          test    al, al
.text:007C5F40          jz      short loc_7C5F48           //check for end of string
.text:007C5F42 loc_7C5F42:
.text:007C5F42           cmp     word ptr [ebp+var_10], si //compare loop counter to
the font's length
.text:007C5F46          jl      short loc_7C5ED0           //loop if below
```

Given that the font variable is only 80 bytes long, submitting a larger font leads to a straight stack overflow with possibility of arbitrary code execution.

Copyright VUPEN Security © Reproduction and redistribution prohibited

Exploitation

Exploitation of this vulnerability is easy because the library was not compiled with the common security protections.

The only difficulty remains in overwriting [ebp+var_10] with a value such that the ending comparison is still valid when the loop variable is modified. This can be achieved by setting a huge length to the font (0x7EFF in the exploit), and overwriting [ebp+var_10] with FCh. This value is then incremented by 2 at 0x007C5F36 and another byte is written at [ebp+var_10 - 1] on the next iteration. If an attacker puts 01h here, then [ebp+var_10] therefore equals 0x01FE and adding 2 makes it equal 0x0200. 0x0200 is less than 0x7EFF, so the program does not exit from the loop and the next character is taken at edi+0C94h+200h, which remains under control.

When the function returns, esi points to the beginning of the file, eax is set to a controlled dword and ecx points to the beginning of the font name. The shellcode can be reached at esi + 0x9E0. The easiest way to proceed here is to return on a "call ecx" and execute:

```
Shr eax, 10      //with eax = 0x09E0XXXX
Add eax, esi
Jmp eax
```

To improve reliability, this exploit uses a return address is Unicode.nls, "call ecx" at 0x0028681B. This should make the exploit run on any vulnerable version of Lotus Note, on Windows XP.

The malicious font is located at offset 0x0771 in the exploit, with a length set to 0x7EFF.

Detection

Attempts to exploit this vulnerability can be detected by watching WordPerfect 6.x files containing long font names. This file format is not very well documented, but finding a font descriptor with a name longer than 80 bytes should instantly trigger the bug.

References

VUPEN/ADV-2009-0757:
http://www.vupen.com/english/advisories/2009/0757

CVE-2008-4564:
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-4564

Changelog

2009-03-24: Initial release
2009-03-25: Exploit improved

Copyright VUPEN Security © Reproduction and redistribution prohibited