Threat Research

Threat Actor Leverages Windows Zero-day Exploit in Payment Card Data Attacks

May 11, 2016 | by Dhanesh Kizhakkinan, Yu Wang, Dan Caselden, Erica Eng | Vulnerabilities

0DAY EXPLOITS | VULNERABILITIES | ZERO-DAY VULNERABILITY | ZERO-DAY EXPLOIT | 0DAY | 0-DAY

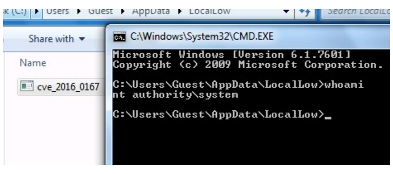
In March 2016, a financially motivated threat actor launched several tailored spear phishing campaigns primarily targeting the retail, restaurant, and hospitality industries. The emails contained variations of Microsoft Word documents with embedded macros that, when enabled, downloaded and executed a malicious downloader that we refer to as PUNCHBUGGY.

PUNCHBUGGY is a dynamic-link library (DLL) downloader, existing in both 32-bit and 64-bit versions, that can obtain additional code over HTTPS. This downloader was used by the threat actor to interact with compromised systems and move laterally across victim environments.

FireEye identified more than 100 organizations in North America that fell victim to this campaign. FireEye investigated a number of these breaches and observed that the threat actor had access to relatively sophisticated tools including a previously unknown elevation of privilege (EoP) exploit and a previously unnamed point of sale (POS) memory scraping tool that we refer to as PUNCHTRACK.

CVE-2016-0167 – Microsoft Windows Zero-Day Local Privilege Escalation

In some victim environments, the threat actor exploited a previously unknown elevation of privilege (EoP) vulnerability in Microsoft Windows to selectively gain SYSTEM privileges on a limited number of compromised machines (Figure 1).



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.

C:\Users\Guest\AppData\LocalLow>whoami
nt authority\system

C:\Users\Guest\AppData\LocalLow>_
```

C:\Windows\System32\CMD.EXE
Users ▸ Guest ▸ AppData ▸ LocalLow
Search LocalLo
Share with ▾
Name
cve_2016_0167

Figure 1. CVE-2016-0167 Local privilege escalation exploit elevates to system

We coordinated with Microsoft, who patched CVE-2016-0167 on the April 12, 2016, Patch Tuesday (MS16-039). Working together, we were able to observe limited, targeted use of this particular exploit dating back to March 8, 2016.

The Threat Actor

We attribute the use of this EoP to a financially motivated threat actor. In the past year, not only have we observed this group using similar infrastructure and tactics, techniques, and procedures (TTPs), but they are also the only group we have observed to date who uses the downloader PUNCHBUGGY and POS malware PUNCHTRACK. Designed to scrape both Track 1 and Track 2 payment card data, PUNCHTRACK is loaded and executed by a highly obfuscated launcher and is never saved to disk.

This actor has conducted operations on a large scale and at a rapid pace, displaying a level of operational awareness and ability to adapt their operations on the fly. These abilities, combined with targeted usage of an EoP exploit and the reconnaissance required to individually tailor phishing emails to victims, potentially speaks to the threat actors' operational maturity and sophistication.

Exploitation Details

Win32k!xxxMNDestroyHandler Use-After-Free

CVE-2016-0167 is a local elevation of privilege vulnerability in the win32k Windows Graphics subsystem. An attacker who had already achieved remote code execution (RCE) could exploit this vulnerability to elevate privileges. In the attack from the wild, attackers first achieved RCE with malicious macros in documents attached to spear phishing emails. They then downloaded and ran a CVE-2016-0167 exploit to run subsequent code as SYSTEM.

CVE-2016-0167 is patched as of April 12, 2016, meaning the attacker's EoP exploit will no longer function on fully updated systems. Microsoft released an additional update (MS16-062) on May 10, 2016, to further improve Windows against similar issues.

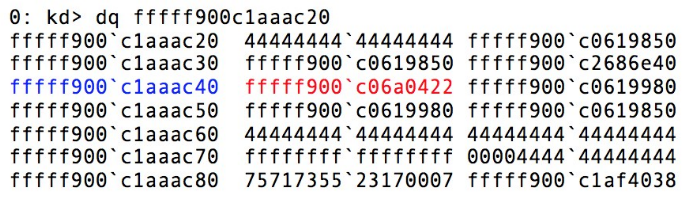
Vulnerability Setup

First, the exploit calls CreateWindowEx() to create a main window. It sets the WNDCLASSEX.lpfnWndProc field to a function that we name WndProc. It installs an application-defined hook (that we name MessageHandler) and an event hook (that we name EventHandler) using SetWindowsHookEx() and SetWinEventHook(), respectively.

Next, it creates a timer with IDEvent 0x5678 in SetTimer(). When the timeout occurs, WndProc receives the WM_TIMER message and will invoke TrackPopupMenuEx() to display a shortcut menu. EventHandler will capture the EVENT_SYSTEM_MENUPOPUPSTART event from xxxTrackPopupMenuEx()and post a message to the kernel. In handling the message, the kernel eventually calls the vulnerable function xxxMNDestroyHandler(), which calls the usermode callback MessageHandler. MessageHandler then causes a use-after-free scenario by calling DestroyWindow()

Heap Control

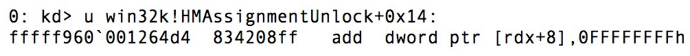
The exploit uses SetSysColors() to perform heap Feng Shui which manipulates the layout of the heap by carefully making heap allocations. In the following snippet, one of the important fields is at address fffff900`c1aaac40, where fffff900`c06a0422 is a window kernel object's (tagWND) base address plus 0x22:



```
0: kd> dq fffff900c1aaac20
fffff900`c1aaac20  44444444`44444444 fffff900`c0619850
fffff900`c1aaac30  fffff900`c0619850 fffff900`c2686e40
fffff900`c1aaac40  fffff900`c06a0422 fffff900`c0619980
fffff900`c1aaac50  fffff900`c0619980 fffff900`c0619850
fffff900`c1aaac60  44444444`44444444 44444444`44444444
fffff900`c1aaac70  ffffffff`ffffffff 00004444`44444444
fffff900`c1aaac80  75717355`23170007 fffff900`c1af4038
```

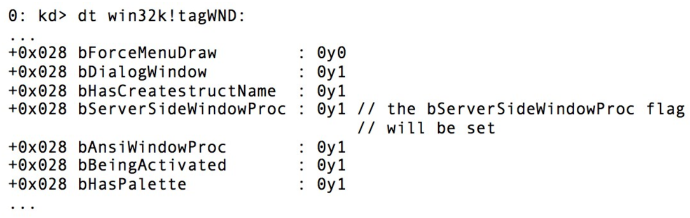
Memory Corruption

The USE operation occurs at HMAssignmentUnlock()+0x14 as shown below:



```
0: kd> u win32k!HMAssignmentUnlock+0x14:
fffff960`001264d4  834208ff   add  dword ptr [rdx+8],0FFFFFFFFh
```

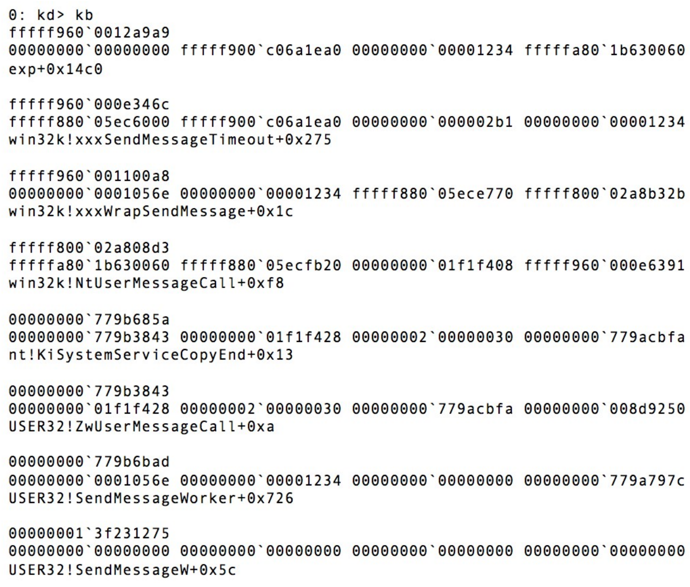
Since RDX contains the base address of tagWND plus 0x22, this instruction will add 0xffffffff to the win32k!tagWND.state field, changing its value from 0x07004000 to 0x07003fff. 0x07004000 indicates that the bServerSideWindowProc flag is unset. When the change occurs, it sets the bServerSideWindowProc flag as shown below.



```
0: kd> dt win32k!tagWND:
...
+0x028 bForceMenuDraw        : 0y0
+0x028 bDialogWindow         : 0y1
+0x028 bHasCreatestructName  : 0y1
+0x028 bServerSideWindowProc : 0y1 // the bServerSideWindowProc flag
                                   // will be set
+0x028 bAnsiWindowProc       : 0y1
+0x028 bBeingActivated       : 0y1
+0x028 bHasPalette           : 0y1
...
```

Code Execution

If a window is marked as server-side (bServerSideWindowPro is set), the lpfnWndProc function pointer will be trusted by default and this can be user-mode shellcode. The following backtrace shows the kernel calling the exploit's shellcode:



```
0: kd> kb
fffff960`0012a9a9
00000000`00000000 fffff900`c06a1ea0 00000000`00001234 fffffa80`1b630060
exp+0x14c0

fffff960`000e346c
fffff880`05ec6000 fffff900`c06a1ea0 00000000`000002b1 00000000`00001234
win32k!xxxSendMessageTimeout+0x275

fffff960`001100a8
00000000`0001056e 00000000`00001234 fffff880`05ece770 fffff800`02a8b32b
win32k!xxxWrapSendMessage+0x1c

fffff800`02a808d3
fffffa80`1b630060 fffff880`05ecfb20 00000000`01f1f408 fffff960`000e6391
win32k!NtUserMessageCall+0xf8

00000000`779b685a
00000000`779b3843 00000000`01f1f428 00000002`00000030 00000000`779acbfa
nt!KiSystemServiceCopyEnd+0x13

00000000`779b3843
00000000`01f1f428 00000002`00000030 00000000`779acbfa 00000000`008d9250
USER32!ZwUserMessageCall+0xa

00000000`779b6bad
00000000`0001056e 00000000`00001234 00000000`00000000 00000000`779a797c
USER32!SendMessageWorker+0x726

00000001`3f231275
00000000`00000000 00000000`00000000 00000000`00000000 00000000`00000000
USER32!SendMessageW+0x5c
```

The shellcode then steals the System process token to elevate a child cmd.exe process.

Mitigation

FireEye products and services identify this activity as Exploit.doc.MVX, Malware.Binary.Doc, PUNCHBUGGY, Malware.Binary.exe, and PUNCHTRACK within the user interfaces.

The latest Windows updates address CVE-2016-0167, and fully protect systems from exploits targeting CVE-2016-0167.

In addition, effective mitigations exist to prevent social engineering attacks that utilize Office macros. Individual users can disable Office macros in their settings and enterprise administrators can enforce a Group Policy to control macro execution for all Office 2016 users. More details about Office macro attacks and mitigations are available here.

Acknowledgements

Thank you to Elia Florio and the Secure@ staff of Microsoft, and Dimiter Andonov, Erye Hernandez, Nick Richard, and Ryann Winters of FireEye for their collaboration on this issue.

< PREVIOUS POST   NEXT POST >

Recent Posts

17 Mar 2020
Six Facts about Address Space Layout Randomization on Windows >

16 Mar 2020
They Come in the Night: Ransomware Deployment Trends >

09 Mar 2020
Crescendo: Real Time Event Viewer for macOS >

RSS FEED: STAY CONNECTED

Company
Why FireEye?
Customer Stories
Careers
Certifications and Compliance
Investor Relations
Supplier Documents

News and Events
Newsroom
Press Releases
Webinars
Events
Awards and Honors
Email Preferences

Technical Support
Incident?
Report Security Issue
Contact Support
Customer Portal
Communities
Documentation Portal

FireEye Blogs
Threat Research
FireEye Stories
Industry Perspectives

Threat Map
View the Latest Threats

Contact Us
+1 877-347-3393

Stay Connected