Analyzed document files hashes

0431F3E850CF50C5735E0BF36A1C97B9

160A45F1BA0DAAEDB4ED457E8AD874A6

306B97C61C2FC793E2B32CFB932DE825

782BA5D56A3D94B941CC9C772CBFA176

695C000179737A74ECA071CC8EF814AA

Quick Heal detects this threat as “Exp:PS.CVE-2015-2545.B”

One common trait found in all documents is that they all were created by the same author and at the same time.

Below is a snapshot from virustotal:

Core document properties	
creator	\u68ee \u52c9(\u96fb\u30b7\u672c)
lastModifiedBy	\u68ee\u52c9
revision	17
created	2015-11-19T06:08:00Z
modified	2015-11-23T08:41:00Z

Based on our analysis, we suspect that all samples were created by the same exploit tool. The names associated with these documents are forged and in Japanese.

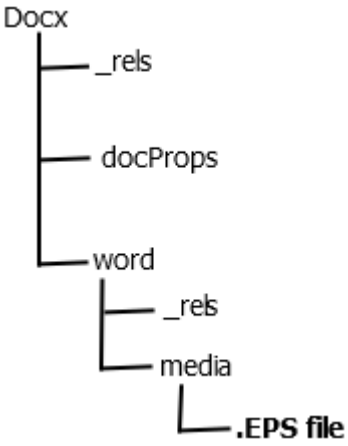
Company Name: 三菱電機株式会社 (Mitsubishi Electric Corporation)
Creator: 森 勉(電シ本) {Tsutomu Mori (Denshihon)}

Details of the Exploitation

EPS file:

Information about Encapsulated post script (EPS) can be found here.

Following is the location of the EPS file in a document file (Docx)



By using UAF (use after free) vulnerability, the exploit writes fake data in freed memory and later convert it type to a string. By dereferencing the fake data as a string, the length of the string will be dereferenced as 0x7FFFFFFF. The exploit uses this string to access whole memory to bypass ASLR and later resolve ROP gadget address to bypass DEP. After bypassing the DEP, the exploit transfers the control to Shellcode.

ROP gadgets

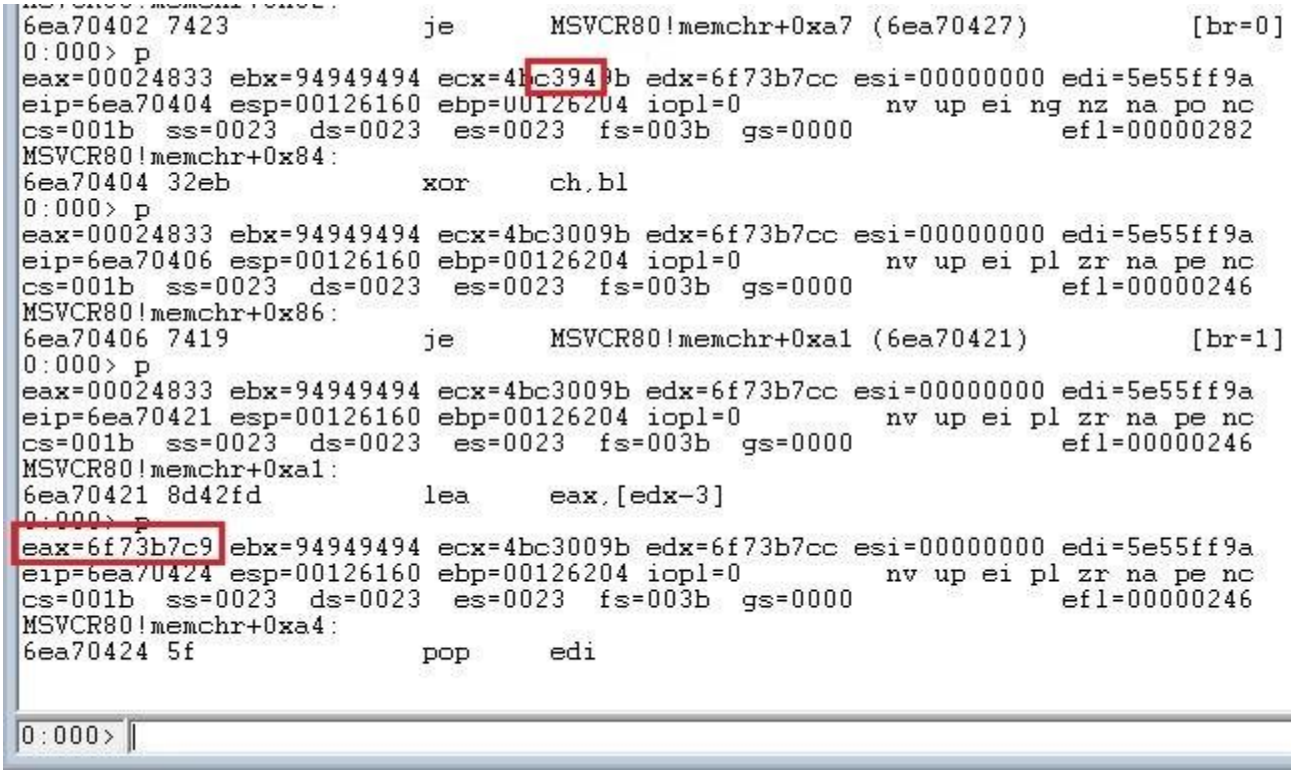
To bypass DEP, the exploit searches for byte sequence (94 C3) and (C2 0C 00) in its current module (epsimp32.dll).

'0x94' instruction is for exchanging stack with the exploit controlled data and '0xC3' is for return.

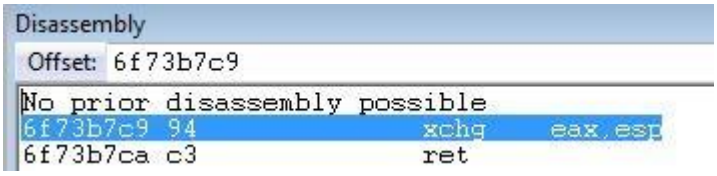
Following is EPS code for searching <94 C3> bytes sequence.

```
colortonee
ImageCurve3 458752 getinterval <94 C3>
search { length /offset exch def pop pop }
{ pop }
ifelse
/colortone1 ImageCurve3 offset add2
def colortonee ImageCurve3 458752 getinterval <C2 0C 00>
search { length /offset exch def pop pop }
{ pop }
```

Windbg snapshot for gadget search:

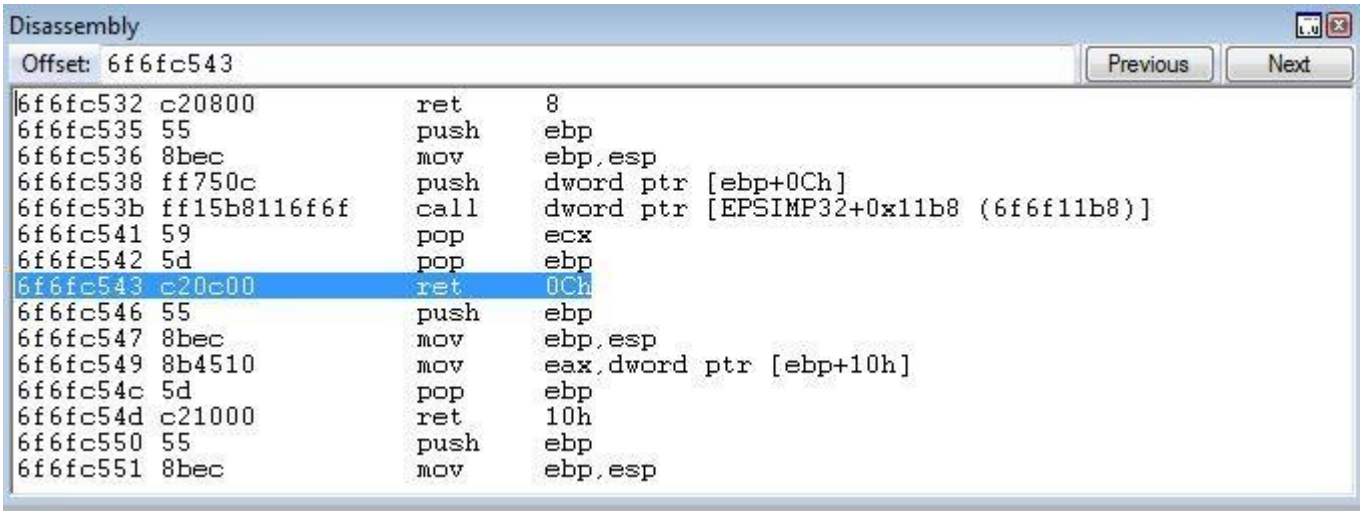


From the above snapshot we can see that it is searching for 94C3, which is resolved to be at address **0x6f73b7c9**.



Thereafter, it will search for <C2 0C 00> bytes sequence which is a “ret 0xC” instruction.

This bytes sequence is found at **0x6FFC543**

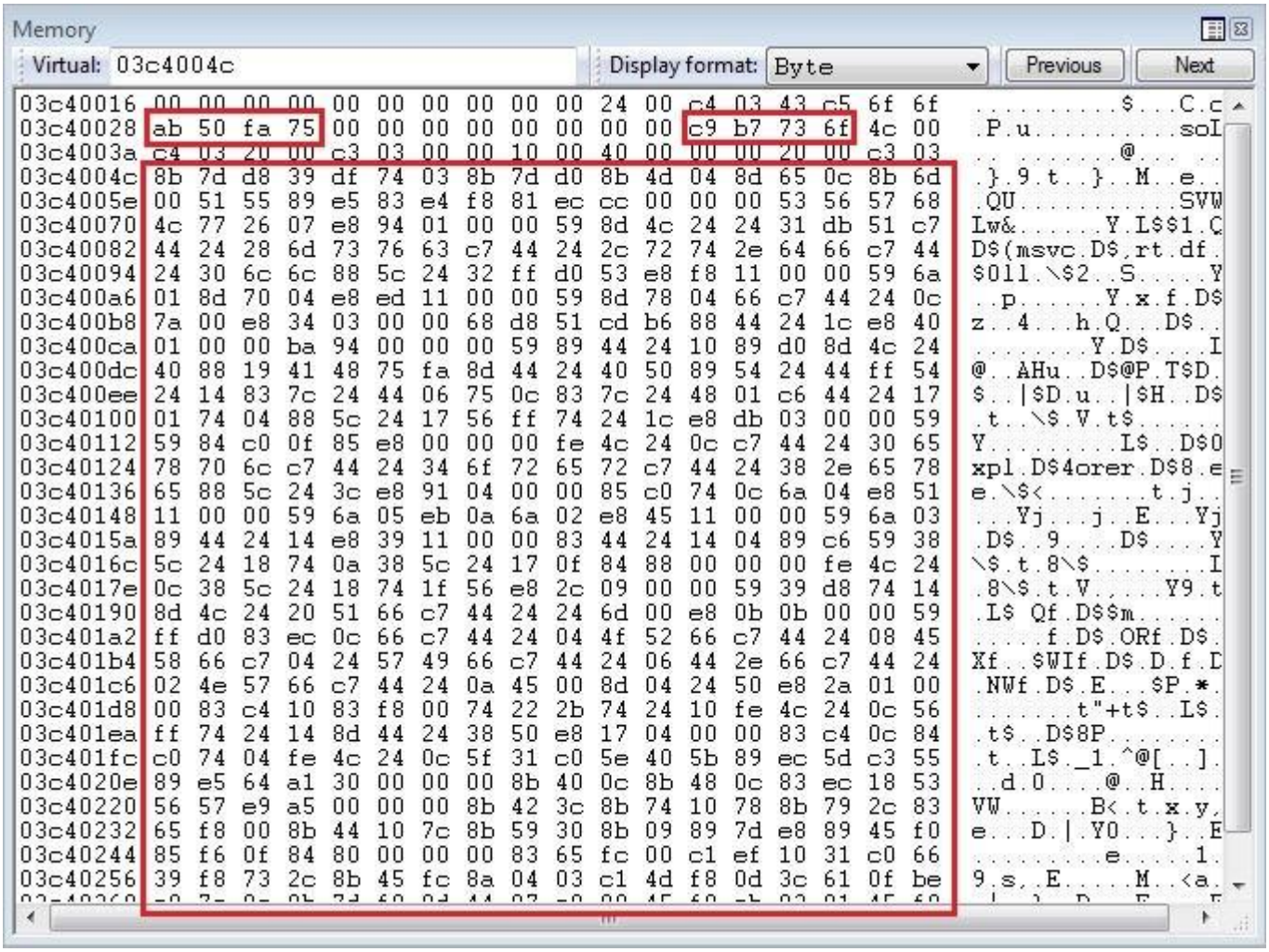


Thereafter, the exploit searches for “Kernel32.dll” and “VirtualProtect” in full memory.

Below is the EPS code for the same.

```
if colortonee ImageCurve3
ImageCurve6 add2 50 getinterval (KERNEL32.dll)
search { length /lenPre exch def pop pop }
{/lenPre -1 def pop}
ifelse lenPre 0 eq { /ImageCurve7
ImageCurve3 color1
add2 forItr add2 circle2 def 0
{ /offset exch def /aFuncNameOffset ImageCurve3
ImageCurve7 add2 offset add2 circle2 def aFuncNameOffset
0 eq {exit}
if colortonee ImageCurve3 aFuncNameOffset add2 50
getinterval (VirtualProtect) search { length /lenPre2 exch def pop pop }
```

After resolving the required addresses, the memory snapshot of this exploit is as following:



In the above snapshot, address of "VirtualProtect" and stack pivot gadget address is followed by a malicious shell code.

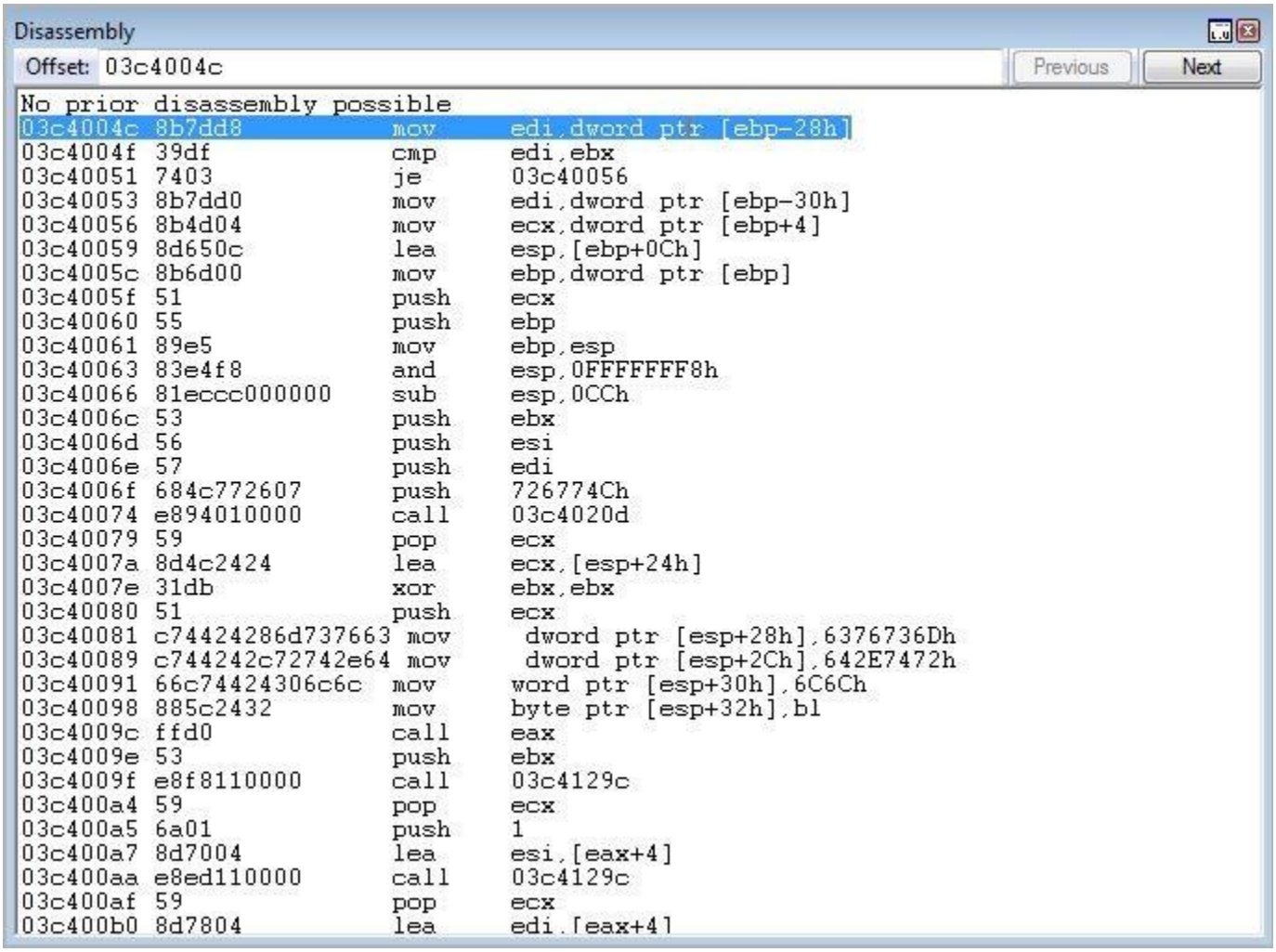
Later, this exploit creates a file type object in EPS code and changes the bytesavailable function pointer to point pivot gadget (ROP).

```
canvas1 type /filetype eq
{ 1 1 atan
pop
1 sin
pop
canvas1
bytesavailable
pop 1 cos
pop exit }
if } if } if globaldict /aDictD undef
```

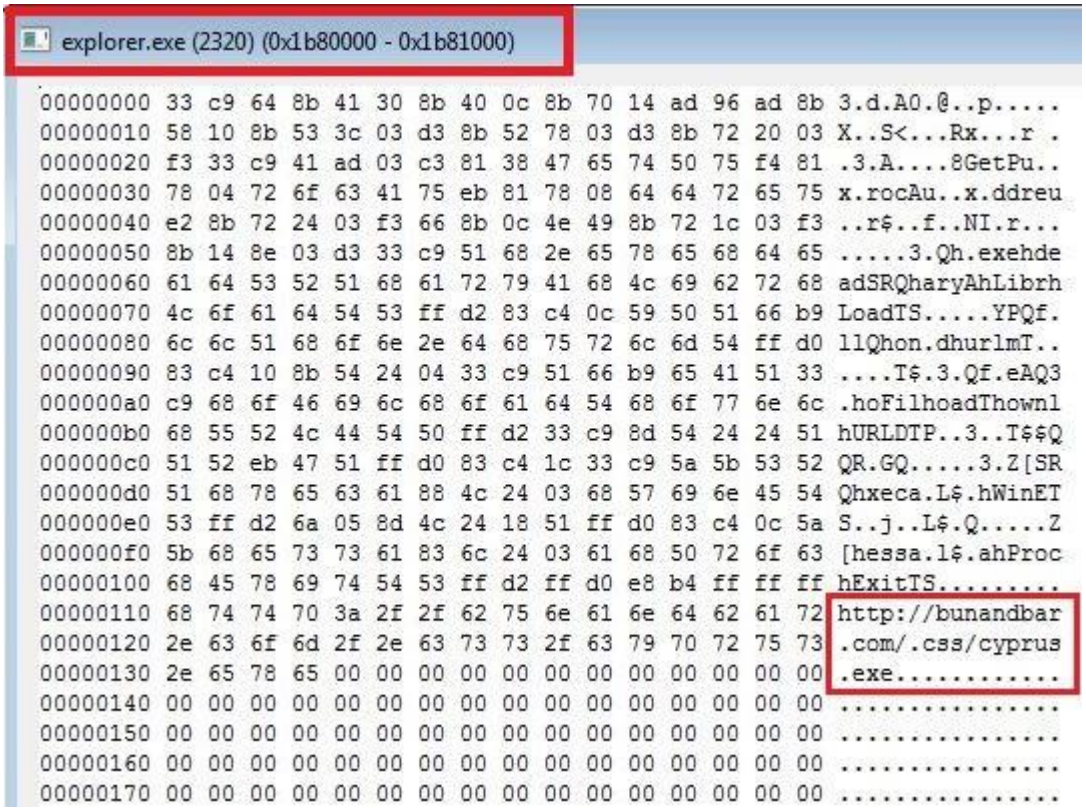
Finally, the exploit makes a call to bytesavailable function, which was overwritten with ROP gadget address, so that it starts executing the ROP chain. Once the ROP chain is executed, it calls "VirtualProtect" to change the shell code memory protection to PAGE_EXECUTE_READ_WRITE.

By using "VIrtualProtect", the exploit successfully bypasses DEP and then control transfers to shellcode.

Shellcode



Shell code creates a temp file to "C:\windows\Temp" and deletes it to check current process (winword.exe) privilege. If the above file operation succeeds, then it injects some shell code in "explorer.exe" as shown below.



It creates a remote thread in explorer.exe to execute injected shellcode, which is responsible for downloading and executing a payload from the below URL path:

hxxp://bunandbar.com/.css/cyprus.exe

Downloaded payload is detected as **TrojanPSW.Tepfer.r3**

This malware can steal a victim's personal information of victim, can download other malware or give a hacker the access to the infected machine.

Conclusion

Previously observed in APT campaigns, this CVE is now being used by spammers for targeting private organizations in India. This is a more recent CVE, which started replacing some old and well known CVEs of MS Office like CVE-2010-3333 and CVE-2012-0158. We expect to see more of this CVE in future spam campaigns.