International Journal of Cyber-Security and Digital Forensics (IJCSDF) 7(1): 72-79
The Society of Digital Information and Wireless Communications (SDIWC), 2018 ISSN: 2305-001

MALWARE ANALYSIS OF BACKDOOR CREATOR : FATRAT

[1]Rakesh singh kunwar, [2]Priyanka sharma, [3]K. V. Ravi kumar
[1,2,3]Raksha Shakti University, Gujarat, India
[1]rakesh.singh.kunwar@rsu.ac.in, [2]ps.it@rsu.ac.in, [3]dir_issm@rsu.ac.in

ABSTRACT

Malwares have become the new vector of cyber crime and hackers are finding new ways to propagate these in all available platforms. Hackers are using social media to propagate backdoors to install it in victim machines to acquire their important data and resources. In the present scenario, Several automatic readymade tools are available over internet using which any script kiddies can create a dangerous malwares and victimize his target. These malware generator are also have categories & generations. It is important to understand that, all the available malware generator previously used in actual scenario of crime of steeling data either in dark net or as a paid service. It is important to understand the working and efficiency of such malware generator . So, In this paper we analyze FATRAT, a backdoor creator which is one of its type and investigate the details with artifacts about it.

KEYWORDS

Malware, Backdoor, FATRAT, JPEG, Malicious Image, Malware forensics

1. INTRODUCTION

In the past few years of cyber world, cybercriminals are implementing new techniques to hide their malicious code inside other files in such a fashion that it is undetected by antivirus.. For it, they are using several complex infection processes than the previous one. As the technology changes, the new generation of cyber criminals are now putting their steps forward. They are now leaving traditional cybercrimes and using advance techniques where the malicious payload is hidden in encrypted files – which ever be the known file format. There are several example over internet in which cyber attacks or incidents shows that attackers are using sophisticated techniques.

In September 2016, Cisco talos-intel identified an exploitable out-of-bounds vulnerability present in the JPEG 2000 image file format parser which is implemented in OpenJPEG library and now identify by its TALOS-2016-0193 identification number or Common Vulnerabilities and Exposures CVE-2016-8332. This JPEG 2000 is a file format which is specially used for embedding images inside the PDF documents. This specific vulnerability is so dangerous that it allow attacker to write out-of-bound heap which include the heap corruption and then arbitrary code execution is possible [1]. In March 2016 Kaspersky Lab, catch a malicious payload hidden in the PNG file i.e. it is embedded with the PNG file. This attack starts with a simple phishing PDF [2].

Such types of incidents shows that now images over the internet are not seen as innocent. They now can be a medium to compromise the protected system. The attacker manipulate the images and these images are harmless until a trigger or input is given in the form of double click done by the user on that image which immediately start a malicious activity [3].

Researchers of Sucuri in July 2013 reported an incident where they found an backdoor present on a site that which was compromised. This backdoor did not depend on the normal patterns like base64 and gzip encoding which is used to hide the contents contained within it [3].

This backdoor is divided into two parts. Both of part are functions in which the first part is a mix of *exif_read_data* function which is used to read the image headers and the *preg_replace* function which is used to execute the content. both PHP functions are actually stored its data within the EXIF header location of a JPEG image.

```
$exif = exif_read_data('/homepages/clientsitepath/

images/stories/food/bun.jpg');

preg_replace($exif['Make'],$exif['Model'],");
```

Both functions are harmless by themselves. However, preg_replace has a tricky and hidden options. On passing "/e" modifier it execute the content(eval), instead of just searching /replacing [3].On looking to bun.jpg file, second part of backdoor looks like:

```
ÿØÿà^@^PJFIF^@^A^B^@^@d^@d^@^@
ÿá^@¡ Exif^@^@II*^@^H^@^@^@^B^@^
O^A^B^@^F^@^@^@&^@^@^@^P^A^B^
@m^@^@^@,^@^@^@ ^@^@^@^@/.*/e^
@ eval ( base64_decode("aWYgKGl zc2V0K
CRfUE9TVFsie noxIl0pKSB7ZXZhbChzdHJ
pcHNsYXNoZXMoJF9QT1NUWyJ6ejEiXSk
pO30='));@ÿì^@^QDucky^@^A^@^D^@^@
^@<^@^@ÿî^@^NAdobe^
```

This types of incident show that, over internet, there are several freely available tools which are used to hide the malicious payload inside the images. FATRAT is one of them. It is a massive exploiting tool which is easy to understand and create backdoor. This tool compiles a malware with popular payload and then the compiled malware can be execute on windows, android, mac . The malware that created with this tool also have an ability to bypass most AV software protection .This tool is used to post exploitation attack like browser attack, dll, bypass AV, etc. In this paper, We compile the malware and payload with the JPEG images and make it a malicious image. After it, analysis is done in our own malware analysis setup lab and show the result.

In this paper, we analyze the backdoor creator and demonstrate the Practical approach which are used by the security personals or researcher to find out the hidden files or proving the presence of hidden data inside the image.

2. FATRAT

The Fatrat is a massive exploiting tool [4]. It create backdoor for windows, linux, mac and android. It can bypass antivirus. It checks for metasploit service and start if not present. It is capable of crafting meterpreter reverse_tcp, start multiple meterpreter reverse_tcp listners. It uses the fast search in searchsploit and many more. The functions provided by the fatrat are:

1) Create backdoor with msfvenom
2) Create FUD 100% Backdoor [slow but powerfull]
3) Create FUD Backdoor with Avoid 1.2
4) Create FUD 100% Backdoor with backdoor-factory [embed]
5) Backdooring Original apk [Instagram, Line, etc]
6) Create Fud Backdoor 1000% with PwmWinds [Excelent]
7) Create Backdoor For office with Microsploit
8) Create auto listeners
9) Jump to msfconsole
10) Searchsploit
11) File Pumper [Increase Your Files Size]
12) Configure Default Lhost & Lport
13) Cleanup

Figure 1 :- Home Screen of Fatrat [5]

The FATRAT facilitate the following facilities under different section shown below:-

Figure 2: PawnWinds to create Powershell [6]

Figure 5: Creator for different platform [7]

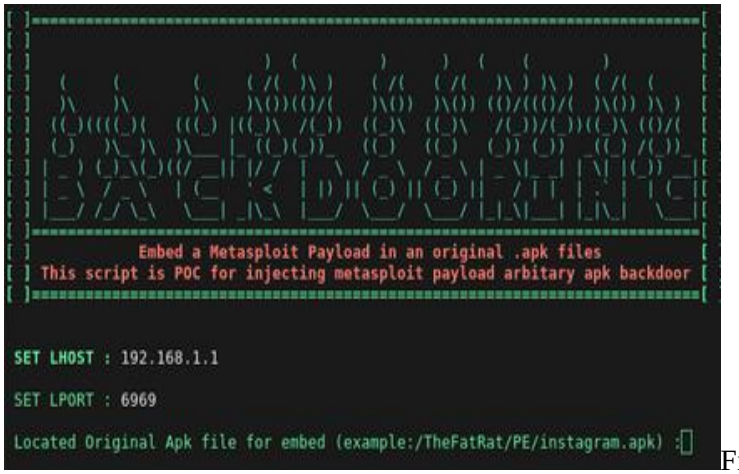

Figure 3: BackDooring for .apk files [6]

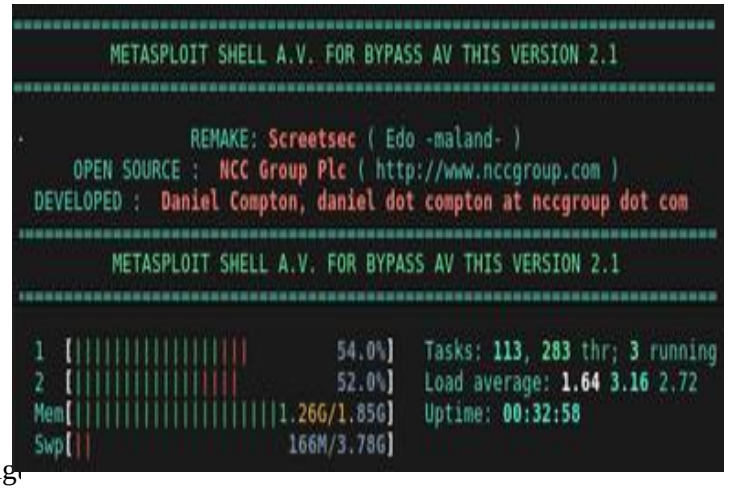

Figure 6: Shell to bypass Antivirus [8]

Figure 4 : SlowButPowerFull meterpreter [7]

Figure 7: Creating Listeners for Payload [8]

2. FATRAT ANALYSIS

Using FATRAT, several samples are created using different functionality provided and discussed previously:

Step 1: Hashing : A Fingerprint for malware-Hashing is used to uniquely identify malware. For it Message Digest Algorithm 5 (MD5) hash function is commonly used.

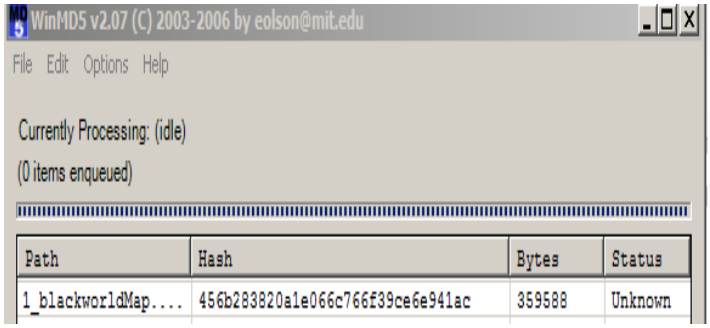

Figure 8: Hash value of the sample

Output Hash :
456b283820a1e066c766f39ce6e941ac

Step 2: Finding Strings:

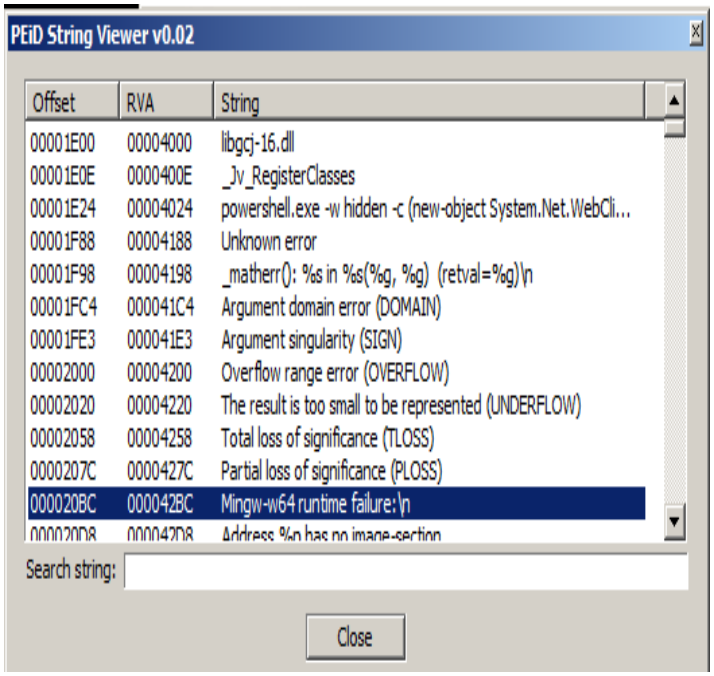

Figure 9: Presence of powershell and mingw

Output : Presence of powershell.exe in hidden mode detected Presence of Mingw detected but failed during execution

Step 3: Detecting Packers with PEiD

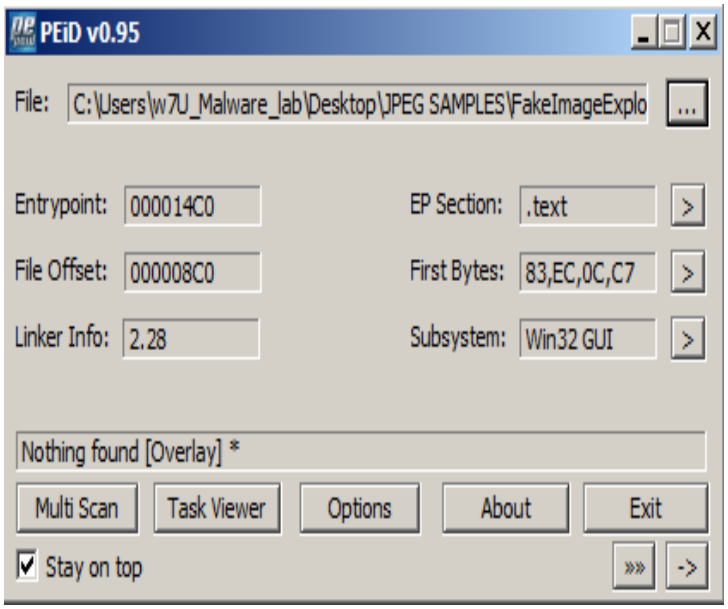

Figure 10: Searching for packers

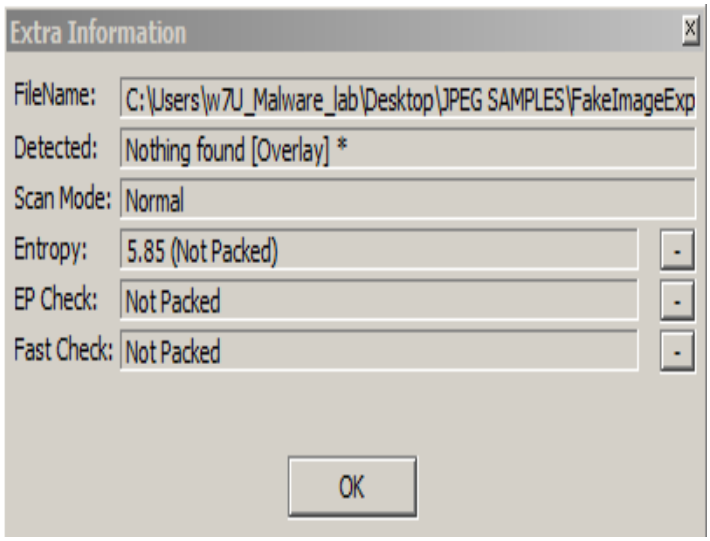

Figure 11: Extra information of sample

Output:- Sample is not packed with any kind of UPX, beside it on digging gets Magic literal: PE32 executable for MS Windows (GUI)

Step 4: Check PE Files Headers and Sections with Image file header

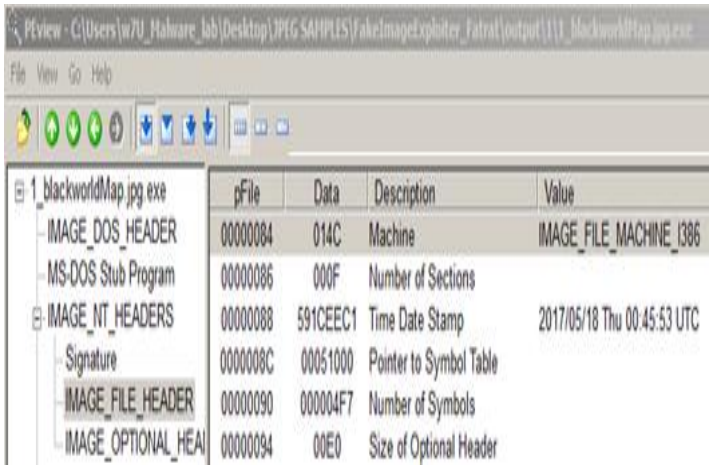

Figure 12: Image file header information

Final Output:

Target machine Intel 386 or later processors and compatible processors
Compilation timestamp 2017-05-18 00:45:53
Entry Point 0x000014C0
Number of sections 15

Step 5: Analysis using IDA Pro. In this step, we show to difference of real genuine Image vs Malicious crafted coded Image.

Real Image :- As we see in IDA pro disassembler, there is no import or export funtions are used as it is a real genuine Image.

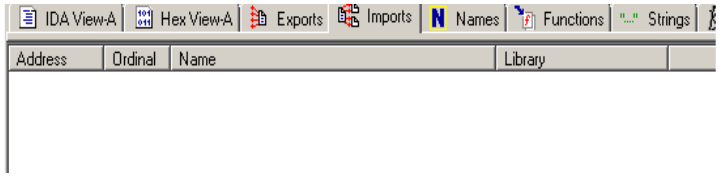

Figure. 13: No import functions in real Image

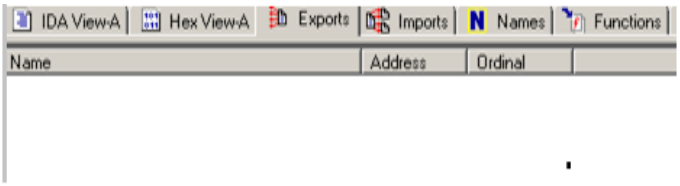

Figure 14: No export functions in real Image

Malicious crafted coded Image.:- There are several import or export functions are used.

Same file but with Embedded codes

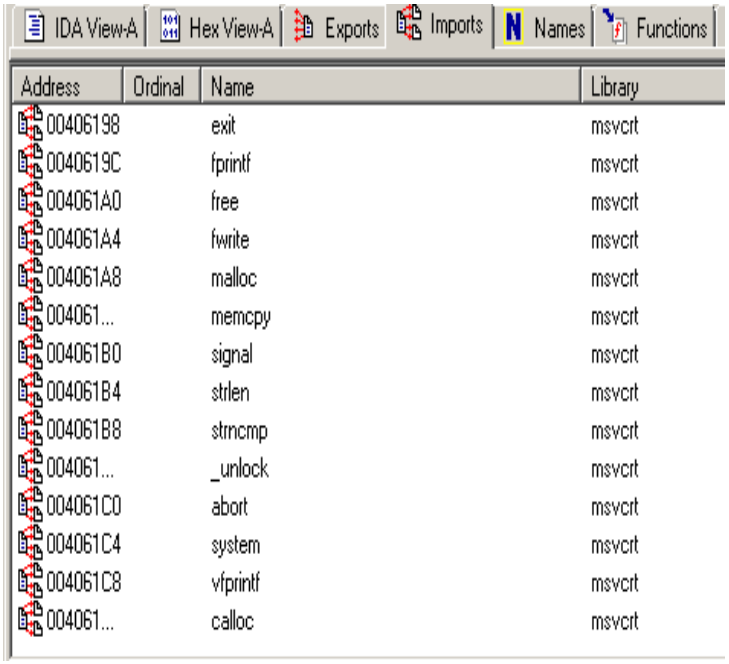

Figure 15(a): Import functions in crafted Image

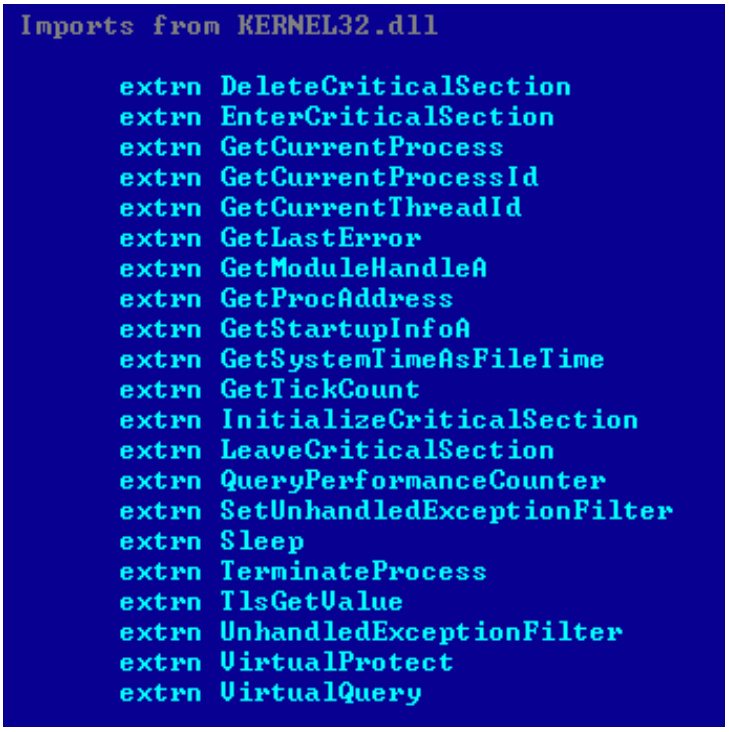

Figure 15(b): Import functions in crafted Image

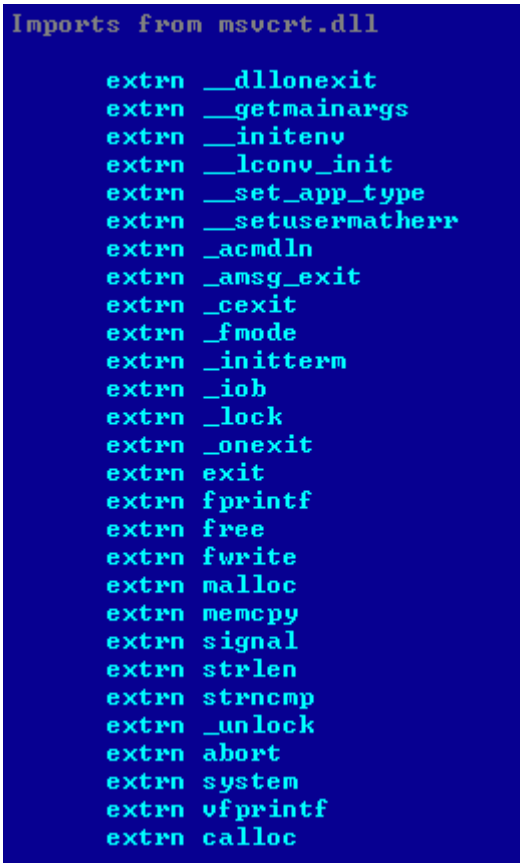

Figure 15(c): Import functions in crafted Image

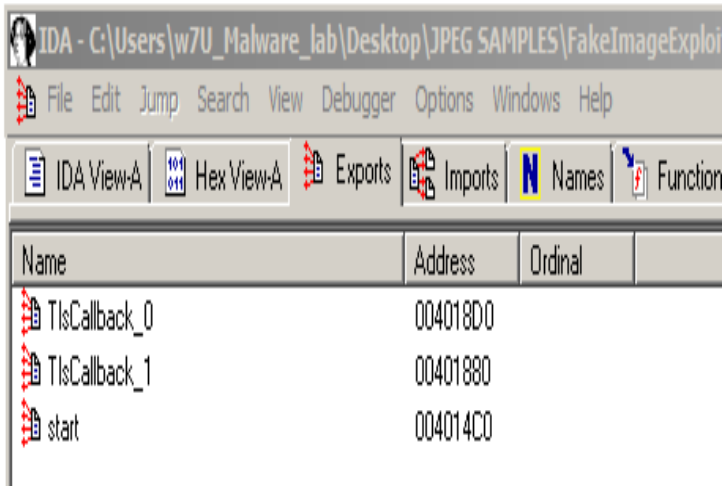

Figure 16: Export functions in crafted Image

Step 7:- Opening shellcode

As there are lots of Import functions hide inside the images and using on executing it.

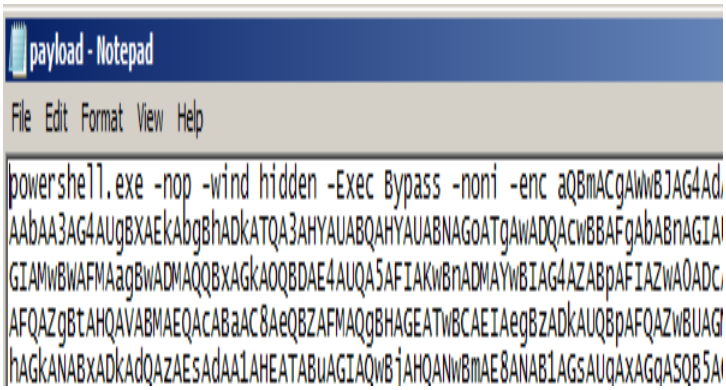

Figure 17: Shellcode embedded with image

Step 8 :- Analyzing the genuine Image vs Malicious crafted coded Image in Hex Editor Neo

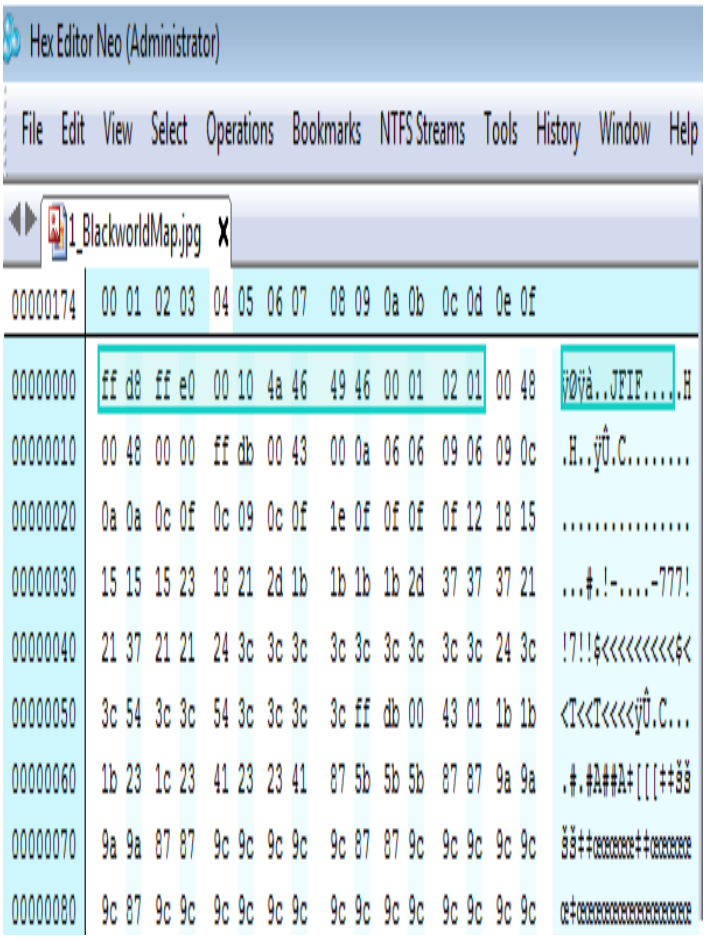

Figure 18: Genuine Image header

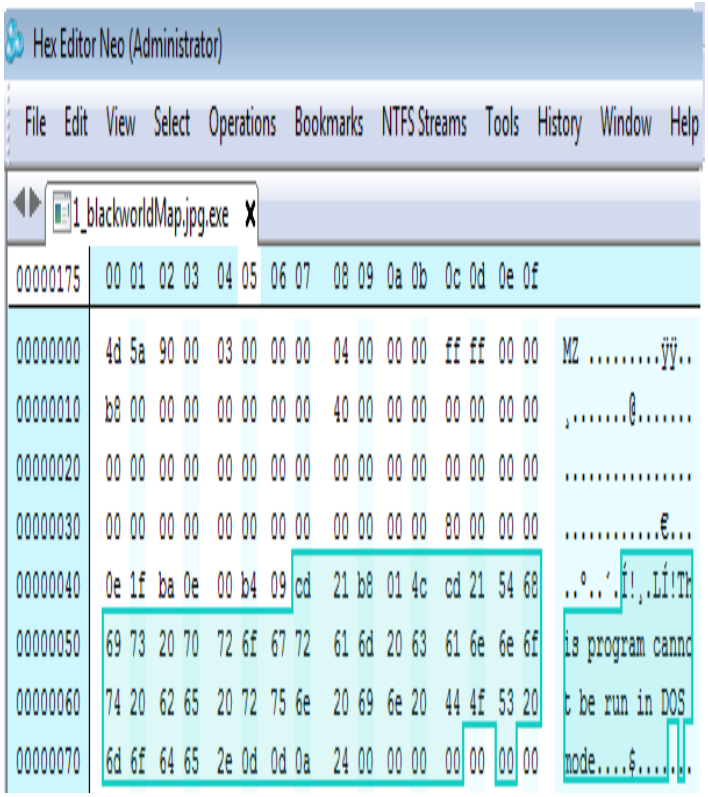

Figure 19: Malicious crafted coded Image header

During the searching of artifacts, we find out the attacker IP and powershell in hidden

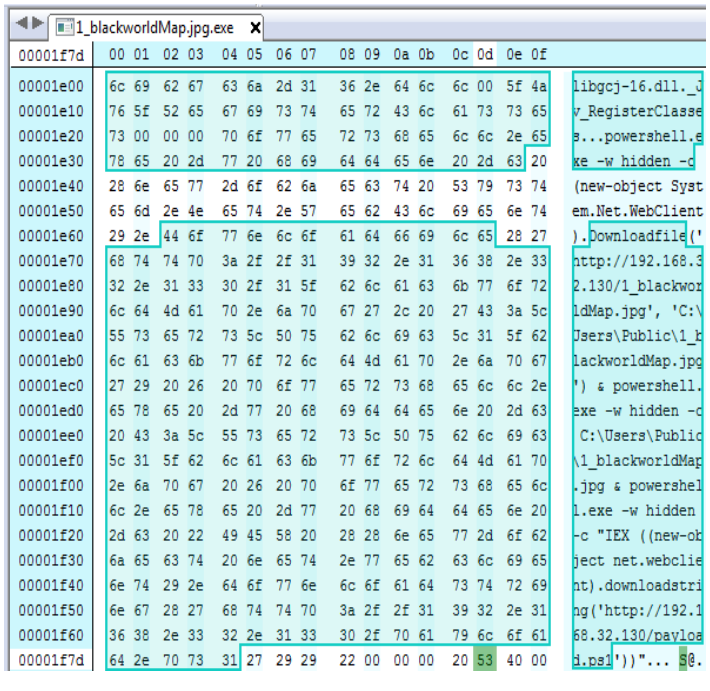

Figure 20: Artifact of malicious images

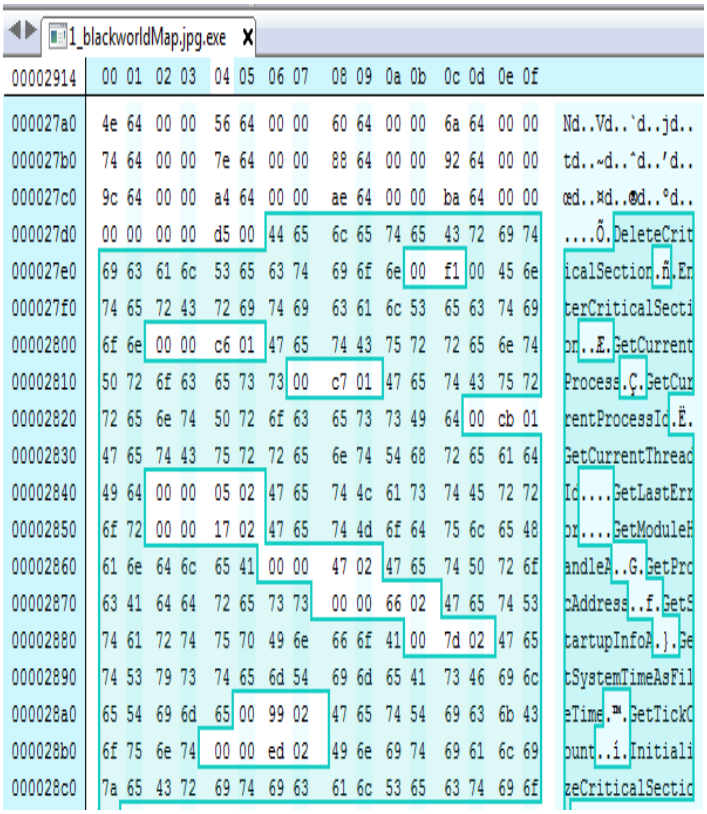

Figure 21: Other critical functions implanted in crafted image

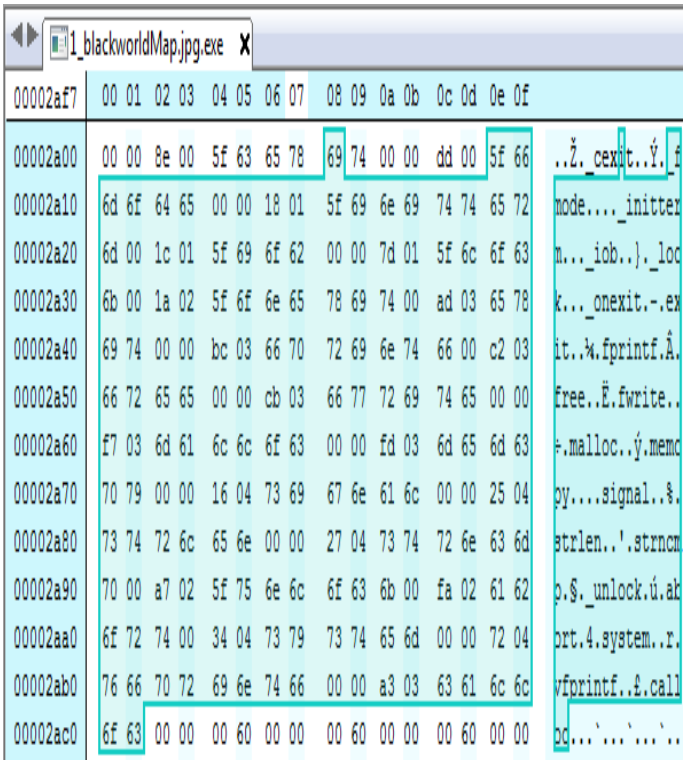

Figure 22: Memory function used in hidden form

CONCLUSION & FUTURE WORK

Malicious payload which is hide using FATRAT are hard to detect & this scheme is generally used by the criminal to act maliciously in other area. For it, they generally used the various types of file format in which JPEG is the most innocent one. So, the challenges of scanning billions of image which are crossing the organization borders, irrelevant to their size, which are non-impacting anomalies are huge.

This provide an opportunity to the malware authors to take it as a advantage and using it to hide malicious code which leave an organization, stealthily send commands to infected victim and transferring various types of malwares across existing types of defenses. So as a researcher it is required to analyze such types samples and detect the images containing the malicious content in the real time scenario.

REFERENCES

[1] Cisco, 2016, "Vulnerability Spotlight:OpenJPEG JPEG2000 mcc record Code Execution Vulnerability",Available at:<http://blogs.cisco.com /security /talos/vulnerability-spotlight-jpeg2000>, [Accessed on 19 Oct 2016].

[2] Securelist, 2016, " PNG Embedded – Malicious payload hidden in a PNG file", Available at: <https:// securelist.com/blog/virus-watch/74297/png-embedded-malic ious-payload-hidden-in-a-png-file/>, [Accessed on 20 Oct 2016].

[3] Sacuri, 2013, " Malware Hidden Inside JPG EXIF Headers", Available at: <https:// blog.sucuri.net /2013/07/malware-hidden-inside-jpg-exif- headers. html/>, [Accessed on 2 Nov 2016].

[4] Fatrat, 2017, "The Fatrat", Available at: <https:// github.com/Screetsec/TheFatRat>,{Accessed on 5/09/2016]

[5] Github, https://cloud.githubusercontent.com/assets/ 17976841 / 25420100/9ee12cf6-2a80-11e7-8dfa-c2 e3cfe71366.png

[6] Github, https://cloud.githubusercontent.com/assets/ 17976841 / 18483873/39d54372-7a10-11e6-890f-41803a33b9c9.png

[7] Github, https://cloud.githubusercontent.com/assets/ 17976841/18483871/39cb81ca-7a10-11e6-84f3-1683067fa4f5.png

[8] Github, https://cloud.githubusercontent.com/assets/ 17976841/18483870/39cb46ba-7a10-11e6-859b-1c1baa3c1b0a.png