

# **Analysis of volatile and non-volatile data images using various tools**

## **The case of M57.biz**

Priyank Jani

Will Van Wart

Weeam Alshangiti

[ppj4900@rit.edu](mailto:ppj4900@rit.edu)

[wv8672@rit.edu](mailto:wv8672@rit.edu)

[waa8642@rit.edu](mailto:waa8642@rit.edu)

# Table of Contents

<b>Executive Summary.....</b>	<b>3</b>
<b>Objectives.....</b>	<b>4</b>
<b>Evidence Analyzed.....</b>	<b>4</b>
<b>Steps taken.....</b>	<b>4</b>
<b>Relevant findings.....</b>	<b>6</b>
<b>Timeline.....</b>	<b>19</b>
<b>Conclusion.....</b>	<b>20</b>

## **Executive Summary**

M57.Biz, a small start-up company, has a serious problem which involves the exfiltration of corporate documents from the laptop of a senior executive Jean. The document is a confidential spreadsheet that contains the names and salaries of the company's key employees. It was found posted to the "comments" section of one of the firm's competitors. As far as we know, the spreadsheet only existed on Jean's computer who said that she has no idea how the data left her laptop and that she must have been hacked. Therefore, in this investigation, we had to look at and analyze a disk image of Jean's laptop in order to figure out how the data was stolen, or if Jean isn't as innocent as she claims.

For this case, we used different forensics tools to analyze Jean's hard drive. We started the investigation process by uploading the hard drive image into EnCase V8 which has many features such as signature analysis which we performed over the evidence that we have. We were able to find out that no files were deliberately hidden on Jean's drive. Then, we went through the different folders and emails trying to find suspicious hints. We found outlook.pst which are used by Microsoft Outlook to store the emails. Therefore, we figured out that we need to perform further analysis on Jean's emails. The only issue we faced was that we couldn't view emails using ECase. That led us to use another forensics tool which is FTK.

When we used FTK, we were able to view all emails in Jean's computer. We found couple of emails between Jean and Alison, the president of the company. But, after couple of emails, we found that an email from Alison asking Jean for some confidential information about the employees in the company such as their names, salaries, and SSNs. Jean replied that she will send the information requested but she didn't. Then another email was sent to Jean from Alison but this one was showing urgency and asking for the confidential information again. But, this time the attacker modified the Return-to header of the email and inserted his/her own email. After that, Jean replied with an emailed and attached the document to it. The file was M57biz.xls and it has the names, salaries, and SSNs of some of the employees in the company. The attacker tanked Jean and asked her to not tell anyone about this. He/she was then able to make the document public by attaching it on the 'comments' section of a competitor's website.

## Objectives

M57 is virtualized web Startup Company. The president of this company is Alison and the CFO is Jean. These individuals were not given last names in the case introduction context. A spreadsheet containing confidential information of all M57 employees was posted on a competitor's website. This spreadsheet came from the CFO Jeans computer. The investigation analyzes Jeans PC hard drive in an effort to determine how this took place and the related events leading up to said spreadsheet being posted on the company's competitor website.

## Evidence Analyzed:

The analysis was performed on Jeans (no last name given) hard drive. Jean is the CFO at M57.biz. The images analyzed are noted here (nps-2008-jean.E01 & nps-2008-jean.E02). on the other hand, a system was diagnosed having active ransomware. The binary file was found and volatile memory was dumped while the binary was active. The hash values of the binary and the memory dump are displayed in table 1.

Evidence	Hash values
A binary file of the machine infected with WannaCry	SHA256 = 24d004a104d4d54034dbcffc2a4b19a11f3900 8a575aa614ea04703480b1022c
Memory Dump of the machine infected with WannaCry	SHA256 = 76e8be1a3761878325fdff39a5ab1ff84922a0b 18947e5268dd9175795ad2bf0

## Steps Taken:

### 1- Processing Evidence with Encase

Using EnCase V8, we started by making sure the time settings were set to the appropriate zones so as to not affect the proper timestamp reporting. Next was to process evidence. The

process evidence option contains major forensic analysis functions that can and often do offer valuable insight. The first analysis performed was file signature analysis. We used this to verify all files and potentially indicate any files that were hidden files. The results rendered the same number of files before and after the analysis was performed. Based on this evidence we can conclude that no files were deliberately hidden on Jeans drive. The next function performed was to expand the compound files with the intention of gaining access to the child files sitting inside of a container, particularly PST files(Personal Table Storage) files, which are used by Microsoft for Outlook Express emails. It was determined that additional tools such as FTK would be required to successfully analyse these emails as EnCase alone did not perform this function. Lastly, the find email/internet artifacts function was run, specifically with the unallocated space function checked to view webmail activity, cache, history, and cookies from browsers like Internet Explorer. After processing this evidence and viewing the artifacts, several files were identified as key evidence as well as potential or supporting clues to the continued investigation. See [Relevant Findings for additional details].

## **2- Processing evidence with FTK**

After analyzing the evidence using EnCase and from the scenarios, we figured out that this case revolves around a bunch of emails that were sent back and forth between Jean and other people from the company and may be from the outside. But, the fact EnCase was not able to display the emails was certainly a big challenge. Therefore, in order to overcome the problems and the limitations we found in EnCase, we decided to perform further analysis using other tool that would give us the ability to view the emails. So, we chose AccessData Forensics Toolkit version 6.2.1.10. There are many different functionalities and features of FTK that can be used to find various things that are related to the case but the most important feature that we focused on was the emails tab since we weren't able to see the different emails between Jean and other people when we were analyzing using EnCase. Therefore, we opened the image of Jean's hard drive in FTK and clicked on the emails tab and we were able to see all emails.

However, there are hundreds of emails saved in Jean's computer and most of them are not relevant to the case such as Google Alert emails, Microsoft Outlook email, and some music and news websites. But, we wanted to look only at the emails going between Jean and the employees in the company. Therefore, we selected and checked every single email associated

with any kind of communication between Jean and other employees. Then we created a bookmark called “suspicious emails” and added the selected emails to that bookmark in order to be able to look at the sequence of events that happened through the emails exchange process.

After clicking on the bookmarks tab, we clicked on the bookmarks folder that we created which is suspicious emails. At this point of the analysis process, we had only the emails between Jean and the employees in the company including Alison who is the president of the company. The emails were ordered in a time manner starting from the first email Jean received to the last one before the image of the hard drive has been taken. We went through every single email in the bookmark looking for anything that could raise an alarm.

### **3- Processing evidence with Volatility, Strings, Bulk\_Extractor, and Wireshark**

We used the Six-step investigative methodology by SANS

1. Identify rogue processes
2. Analyze process DLLs and handles
3. Review network artifacts
4. Look for evidence of code injection
5. Check for signs of rootkit
6. Dump suspicious processes and drivers

### **Relevant Findings:**

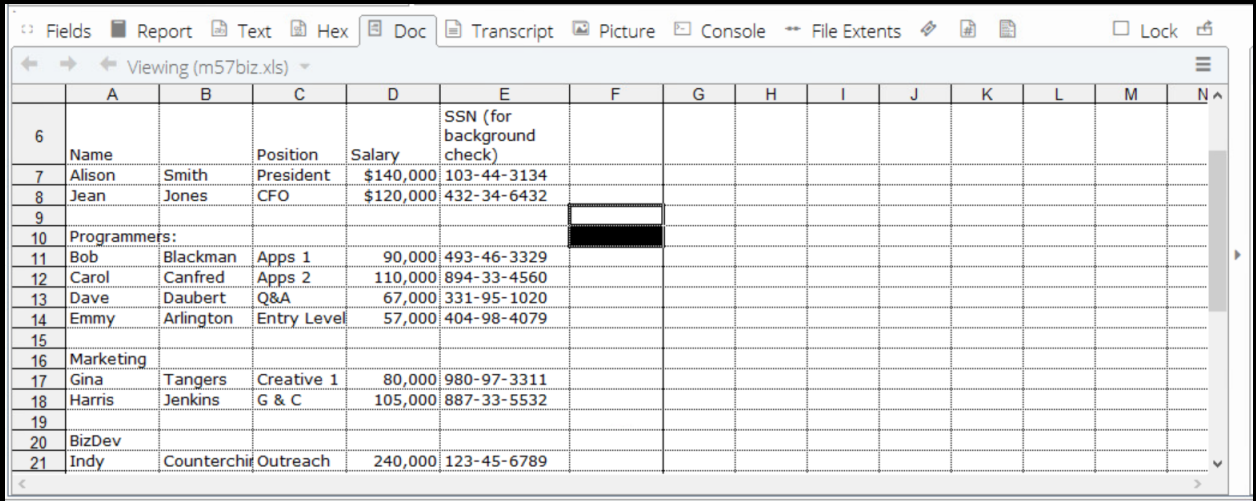
#### **1- Findings from EnCase**

The C directory consisted of 6 directories which are listed below with their creation time and hash values.

Directory name	Created	MD5 Hash value
\$Extend	05/13/08 06:18:43 PM	
Documents and Settings	05/13/08 06:20:13 PM	2b9e92d11f5dd7010af98eacc3e93c22
Program Files	05/13/08 06:20:41 PM	442ffabc18f23ec5ffa79f27192b517b
RECYCLER	07/11/08 02:00:56 PM	4b9ad55e74a05476bd4f1092811461b3

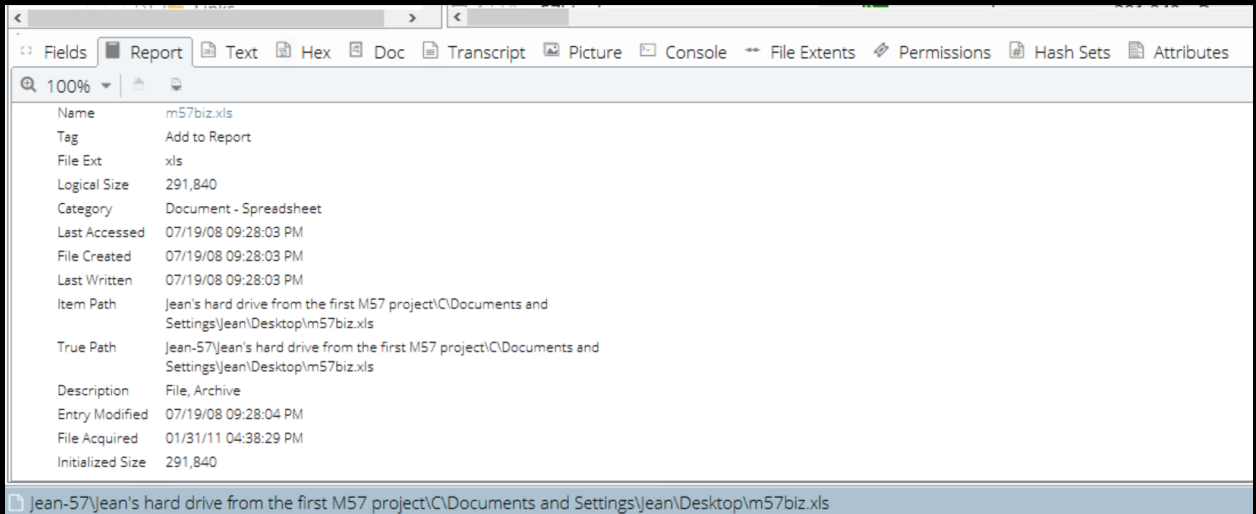
System Volume Information	07/20/08 09:22:03 PM	b45d3aef2fd175143d1997b4386707b6
WINDOWS	07/20/08 09:22:26 PM	eabccca6dbf48906656e6e1e8ff49667

Inside C/Documents and Settings/Jean/Desktop/m57biz.xls → In this file, it is apparent that all employees have vulnerable personal information exposed. Ex: [name, position, salary, ssn]. This file was compared to the file given before the analysis began and it is identical. The file stored on Jeans drive is the same as the file provided for the analysis. Of further note, the report of this file indicates that it was created on 07/19/08 at 9:28:03 pm. See Image 1 & 2 for more details.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
6	Name		Position	Salary	SSN (for background check)									
7	Alison	Smith	President	\$140,000	103-44-3134									
8	Jean	Jones	CFO	\$120,000	432-34-6432									
9														
10	Programmers:													
11	Bob	Blackman	Apps 1	90,000	493-46-3329									
12	Carol	Canfred	Apps 2	110,000	894-33-4560									
13	Dave	Daubert	Q&A	67,000	331-95-1020									
14	Emmy	Arlington	Entry Level	57,000	404-98-4079									
15														
16	Marketing													
17	Gina	Tangers	Creative 1	80,000	980-97-3311									
18	Harris	Jenkins	G & C	105,000	887-33-5532									
19														
20	BizDev													
21	Indy	Counterchir	Outreach	240,000	123-45-6789									

Image 1: m57biz.xls contents



Attribute	Value
Name	m57biz.xls
Tag	Add to Report
File Ext	xls
Logical Size	291,840
Category	Document - Spreadsheet
Last Accessed	07/19/08 09:28:03 PM
File Created	07/19/08 09:28:03 PM
Last Written	07/19/08 09:28:03 PM
Item Path	Jean's hard drive from the first M57 project\C\Documents and Settings\Jean\Desktop\m57biz.xls
True Path	Jean-57\Jean's hard drive from the first M57 project\C\Documents and Settings\Jean\Desktop\m57biz.xls
Description	File, Archive
Entry Modified	07/19/08 09:28:04 PM
File Acquired	01/31/11 04:38:29 PM
Initialized Size	291,840

Image 2: Report/Metadata for m57biz.xls

Inside C/Documents and Settings/Jean/My Documents/AIM Logger/m57 jean/IM Logs  
 → This particular file has two main components that are worth noting(see Image 3). The first is that there appears to be a conversation taking place between the CFO and the President of M57.biz. However, the person utilizing Jeans username does not seem to be aware the he/she is in fact talking to Alison, who is the boss. Additionally, the person utilizing Jeans's username seems to believe that the boss is a male by referring to the boss as a “he” rather than a she. Note: Alison the President is a female.

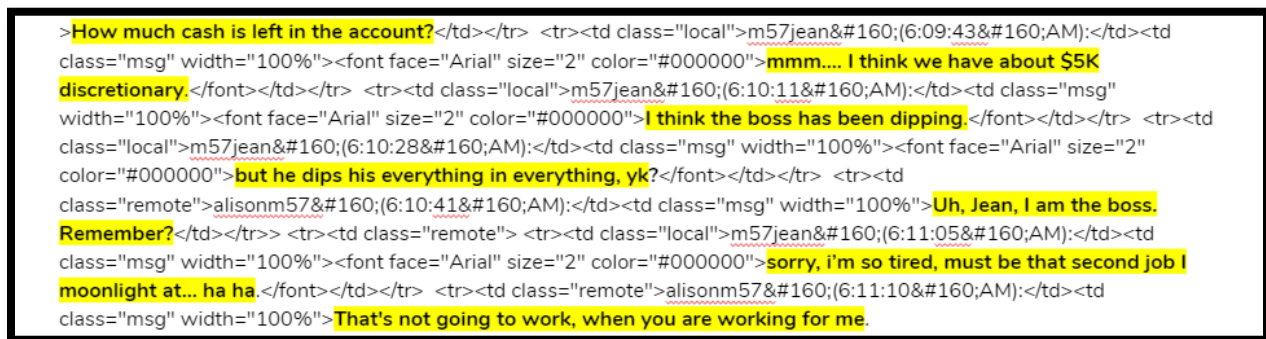


Image 3: AIM conversation detailing possible phishing scam

Inside C/Documents and Settings/Jean/Application Data/ Microsoft/Outlook →  
 As noted in the “Processing Evidence with EnCase” paragraph, processing evidence on the compound files allowed for the possibility of gaining access to the child files sitting inside of a container, particularly PST files(Personal Table Storage) files, which are used by Microsoft for Outlook Express emails. Below is a Image 4, showing the inability to view said PST files even after running the unallocated space function. To move forward, additional tools were considered and FTK was selected to successfully view the contents and analyze the emails for further evidence.



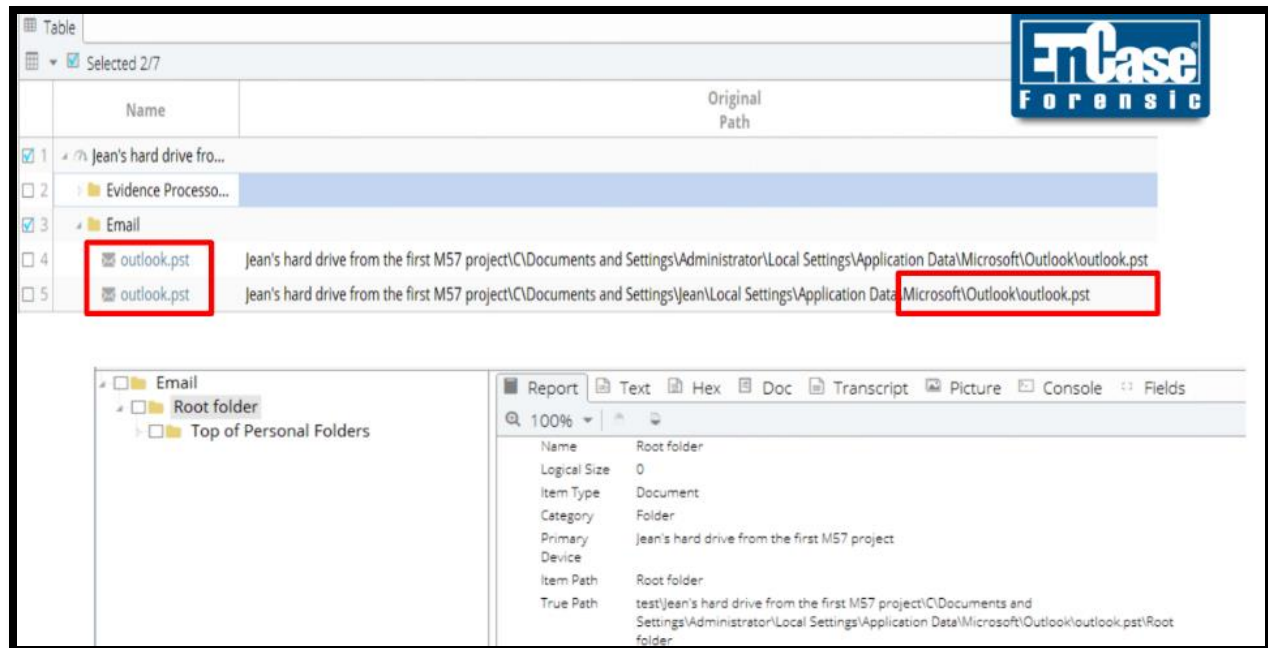


Image 4: Microsoft Outlook Email folder with 2 outlook.pst files

## 2- Findings from FTK

When we decided on using FTK, we mainly chose it because of its great way of representing emails. We specifically wanted to look at and analyze email headers because they store lots of information that are hidden from the user who can only see the body, or the message, of the email. Therefore, we started to look at the headers of every single email in the bookmark that we created, which has only the emails between Jean and some of the employees in the company. The total number of emails was 377 from different sources but as mentioned previously, most emails were from Google Alerts and ads from different websites. However, the emails in the bookmarks tab, specifically in the suspicious emails folder are the ones that we analyzed in depth. After reading and analyzing the emails, we found a very interesting email which we thought was very suspicious as in image 5.

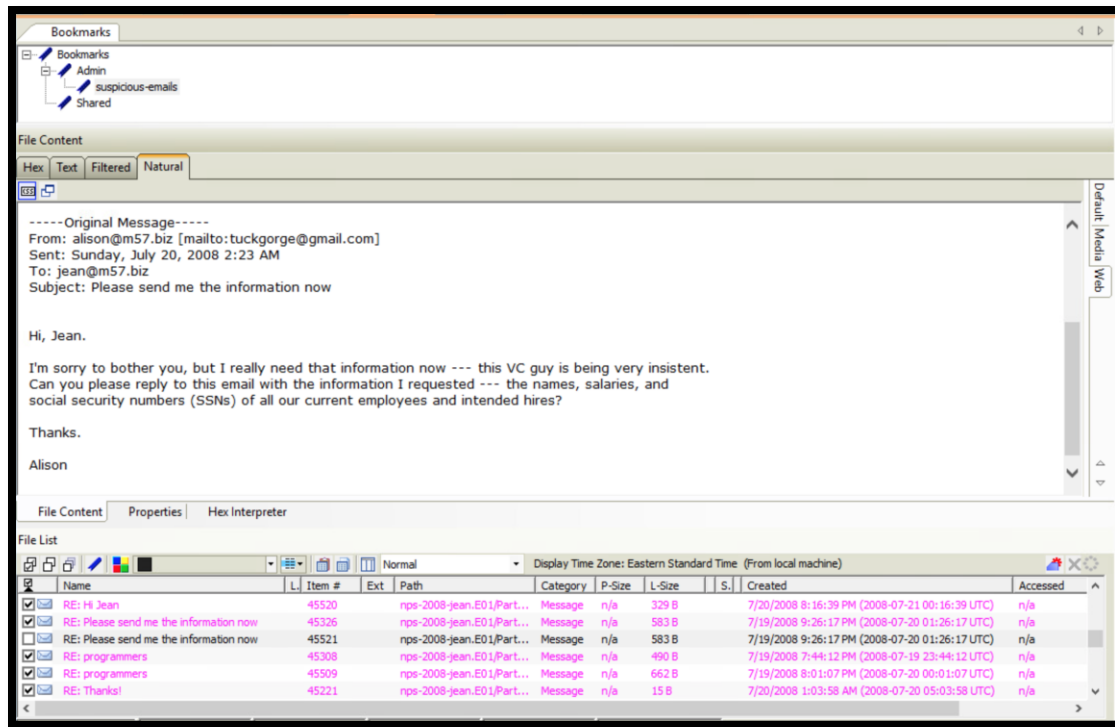


Image 5: An email from the attacker to Jean

The email displayed in the image below was coming from Alison's email address, who is the president of the company. The email has the company's domain which is m57.biz. She was asking Jean for some confidential information, such as salaries and SSNs, of all current employees. We noticed that the email was sent from alison@m57.biz while the return-to header or the return path was set to be tuckgorge@gmail.com. This means that the attacker has obtained Alison's email ID from M57's website probably and used it to send forged emails to Jean who didn't see the return-to email address which was the attacker's email.

We looked further to the sequence of the emails and specifically to Jean's reply to the attacker's email in order to find out if she had given him/her the information. But of course, as a result of the request that Jean got from the attacker who was pretending to be Alison, she created the document with all information needed and attached it to the reply email. She specifically mentioned that she attached it to the email. Therefore, we knew that we have to look for the file in the attachments. Eventually, we clicked on the emails tab again and clicked on the email attachments. We searched for any file that could have the confidential information until we found it as seen in image 6. The file was XLS document and the attacker was able to get it in

some phishing and spoofing attacks. Then, he/she made it public by attaching it on the ‘comments’ section of a competitor’s website.

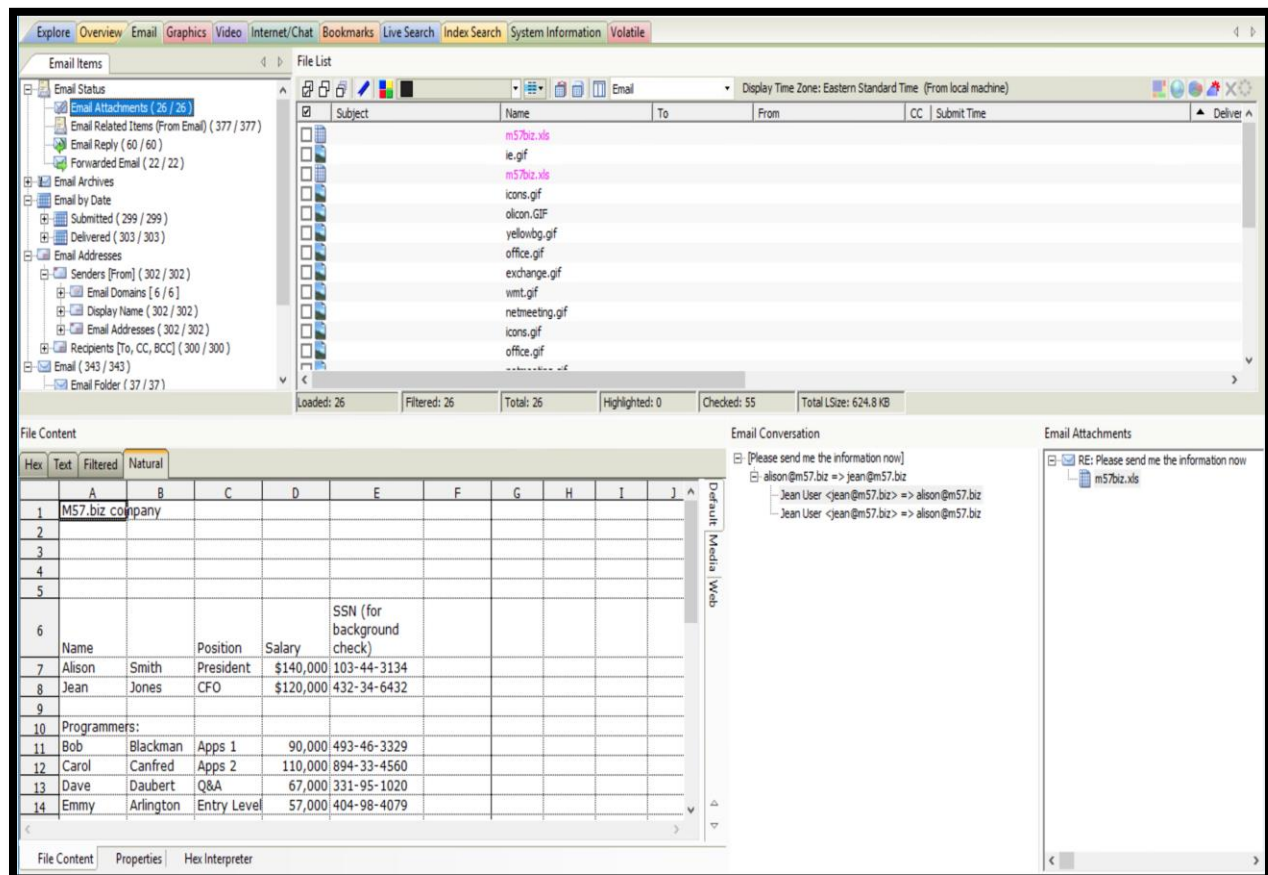


Image 6: The exfiltrated document in the attachments

### 3- Findings from Volatility, Strings, Bulk\_Extractor, and Wireshark

The first step was using strings and using grep, finding executables and web urls. Below are the commands and the output of using the strings command:

**strings -n 6 filename.bin | grep http**

The URL found was one having random characters and later it was discovered that the URL was a killswitch. Wannacry infects system maliciously and tries connecting to a registered domain. If the connection is successful, it doesn't take action else it proceeds with the infection.

Wannacry can drop binaries on the fly to run different tasks on system :

**strings -n 6 filename.bin | grep exe**

We observe more interesting strings related to Mutex creation on the infected system and also granting or modifying discretionary access controls on infected system. A strange password type string 'WNcry@2ol7' could also be spotted. A further deep dive into strings shows various files with .wnry extension.

Using Volatility we uncover memory resident artifacts. We start with volatility *imageinfo*

**vol.py -f wcry.raw imageinfo**

Next, we run the *pslist*, *psscan* command to investigate processes running at the time of acquiring memory. Knowledge of native Windows processes helps as other processes can be identified easily. PID 1940 initiated PID 740. Both processes look completely strange and also tasksche was spotted in strings of wannacry binary.

**vol.py -f wcry.raw --profile=WinXPSP2x86 pslist**

**vol.py -f wcry.raw --profile=WinXPSP2x86 psscan**

```
sansforensics@siftworkstation -> ~/D/volatility-master
$ vol.py -f wcry.raw --profile=WinXPSP2x86 pslist
Volatility Foundation Volatility Framework 2.6
Offset(V)  Name                PID  PPID  Thds  Hnds  Sess  Wow64  Start              Exit
-----
0x823c8830  System              4    0     51   244   0     0     2017-05-12 21:21:55 UTC+0000
0x82169020  smss.exe            348   4     3    19   0     0     2017-05-12 21:22:00 UTC+0000
0x82161da0  csrss.exe           596   348   12   352   0     0     2017-05-12 21:22:01 UTC+0000
0x8216e020  winlogon.exe        620   348   23   536   0     0     2017-05-12 21:22:01 UTC+0000
0x821937f0  services.exe        664   620   15   265   0     0     2017-05-12 21:22:01 UTC+0000
0x82191650  lsass.exe           676   620   23   353   0     0     2017-05-12 21:22:01 UTC+0000
0x8221a2c0  svchost.exe         836   664   19   211   0     0     2017-05-12 21:22:02 UTC+0000
0x821b5230  svchost.exe         904   664   9    227   0     0     2017-05-12 21:22:03 UTC+0000
0x821af7e8  svchost.exe        1024   664   79   1366   0     0     2017-05-12 21:22:03 UTC+0000
0x8203b7a8  svchost.exe        1084   664   6    72    0     0     2017-05-12 21:22:03 UTC+0000
0x821bea78  svchost.exe        1152   664   10   173   0     0     2017-05-12 21:22:06 UTC+0000
0x821e2da0  spoolsv.exe        1484   664   14   124   0     0     2017-05-12 21:22:09 UTC+0000
0x821d9da0  explorer.exe       1636  1608   11   331   0     0     2017-05-12 21:22:10 UTC+0000
0x82218da0  tasksche.exe       1940  1636   7    51    0     0     2017-05-12 21:22:14 UTC+0000
0x82231da0  ctfmon.exe         1956  1636   1    86    0     0     2017-05-12 21:22:14 UTC+0000
0x81fb95d8  svchost.exe        260   664   5    105   0     0     2017-05-12 21:22:18 UTC+0000
0x81fde308  @WanaDecryptor@    740   1940   2    70    0     0     2017-05-12 21:22:22 UTC+0000
0x81f747c0  wuaucnt.exe       1768  1024   7    132   0     0     2017-05-12 21:22:52 UTC+0000
0x82010020  alg.exe            544   664   6    101   0     0     2017-05-12 21:22:55 UTC+0000
0x81fea8a0  wscntfy.exe       1168  1024   1    37    0     0     2017-05-12 21:22:56 UTC+0000
sansforensics@siftworkstation -> ~/D/volatility-master
$
```

Image 7: pslist plugin results



```
sansforensics@siftworkstation -> ~/D/volatility-master
$ vol.py -f wcry.raw --profile=WinXPSP2x86 psscan
Volatility Foundation Volatility Framework 2.6
Offset(P)  Address  Name  PID  PPID  PDB  Time created  Time exited
-----
0x0000000001f4daf0 taskdl.exe 860 1940 0x199f6000 2017-05-12 21:26:23 UTC+0000 2017-05-12 21:26:23 UTC+0000
0x0000000001f53d18 taskse.exe 536 1940 0x1986c000 2017-05-12 21:26:22 UTC+0000 2017-05-12 21:26:23 UTC+0000
0x0000000001f69b50 @WanaDecryptor@ 424 1940 0x18fa2000 2017-05-12 21:25:52 UTC+0000 2017-05-12 21:25:53 UTC+0000
0x0000000001f747c0 wuauclt.exe 1768 1024 0x11629000 2017-05-12 21:22:52 UTC+0000
0x0000000001f8ba58 @WanaDecryptor@ 576 1940 0x19671000 2017-05-12 21:26:22 UTC+0000 2017-05-12 21:26:23 UTC+0000
0x0000000001fb95d8 svchost.exe 260 664 0x0ce48000 2017-05-12 21:22:18 UTC+0000
0x0000000001fde308 @WanaDecryptor@ 740 1940 0x0de3a000 2017-05-12 21:22:22 UTC+0000
0x0000000001fea8a0 wscntfy.exe 1168 1024 0x12217000 2017-05-12 21:22:56 UTC+0000
0x0000000001ffa710 0 0 0x17d3f000
0x0000000002010020 alg.exe 544 664 0x1238d000 2017-05-12 21:22:55 UTC+0000
0x000000000203b7a8 svchost.exe 1084 664 0x0838c000 2017-05-12 21:22:03 UTC+0000
0x0000000002161da0 csrss.exe 596 348 0x07752000 2017-05-12 21:22:00 UTC+0000
0x0000000002169020 smss.exe 348 4 0x0683e000 2017-05-12 21:21:55 UTC+0000
0x000000000216a020 winlogon.exe 620 348 0x07957000 2017-05-12 21:22:01 UTC+0000
0x0000000002191658 lsass.exe 676 620 0x07bb7000 2017-05-12 21:22:01 UTC+0000
0x00000000021937f0 services.exe 664 620 0x07bad000 2017-05-12 21:22:01 UTC+0000
0x00000000021af7e8 svchost.exe 1024 664 0x081f7000 2017-05-12 21:22:03 UTC+0000
0x00000000021b5230 svchost.exe 904 664 0x08131000 2017-05-12 21:22:03 UTC+0000
0x00000000021bea78 svchost.exe 1152 664 0x08a15000 2017-05-12 21:22:06 UTC+0000
0x00000000021d9da0 explorer.exe 1636 1608 0x0add4000 2017-05-12 21:22:10 UTC+0000
0x00000000021e2da0 spoolsv.exe 1484 664 0x0a462000 2017-05-12 21:22:09 UTC+0000
0x0000000002218da0 tasksche.exe 1940 1636 0x0c0a2000 2017-05-12 21:22:14 UTC+0000
0x000000000221a2c0 svchost.exe 836 664 0x07e3e000 2017-05-12 21:22:02 UTC+0000
0x0000000002231da0 ctfmon.exe 1956 1636 0x0c01f000 2017-05-12 21:22:14 UTC+0000
0x00000000023c8830 System 4 0 0x00039000
sansforensics@siftworkstation -> ~/D/volatility-master
$
```

Image 8: psscan plugin results

We can see terminated processes taskdl.exe, taskse.exe along with parent process PID 1940.

```
sansforensics@siftworkstation -> ~/D/volatility-master
$ vol.py -f wcry.raw --profile=WinXPSP2x86 psscan | grep 1940
Volatility Foundation Volatility Framework 2.6
Offset(P)  Address  Name  PID  PPID  PDB  Time created  Time exited
-----
0x0000000001f4daf0 taskdl.exe 860 1940 0x199f6000 2017-05-12 21:26:23 UTC+0000 2017-05-12 21:26:23 UTC+0000
0x0000000001f53d18 taskse.exe 536 1940 0x1986c000 2017-05-12 21:26:22 UTC+0000 2017-05-12 21:26:23 UTC+0000
0x0000000001f69b50 @WanaDecryptor@ 424 1940 0x18fa2000 2017-05-12 21:25:52 UTC+0000 2017-05-12 21:25:53 UTC+0000
0x0000000001f8ba58 @WanaDecryptor@ 576 1940 0x19671000 2017-05-12 21:26:22 UTC+0000 2017-05-12 21:26:23 UTC+0000
0x0000000001fde308 @WanaDecryptor@ 740 1940 0x0de3a000 2017-05-12 21:22:22 UTC+0000
0x0000000002218da0 tasksche.exe 1940 1636 0x0c0a2000 2017-05-12 21:22:14 UTC+0000
sansforensics@siftworkstation -> ~/D/volatility-master
$
```

Image 9: psscan grepped for pid 1940

Next, we run *dlllist* plugin to identify process DLLs and the path from which the process executed from. This gives us a clear understanding of malicious processes if they are running by dropping binaries in uncommon folders.

```
vol.py -f wcry.raw --profile=WinXPSP2x86 dlllist -p 1940
```

```
sansforensics@siftworkstation -> ~/D/volatility-master
$ vol.py -f wcry.raw --profile=WinXPSP2x86 dlllist -p 1940
Volatility Foundation Volatility Framework 2.6
*****
tasksche.exe pid: 1940
Command line : "C:\Intel\ivecuqnanpnirkt615\tasksche.exe"
Service Pack 3

Base_PIR-ThreatIntel- Size LoadCount Path
-----
0x00400000 0x35a000 0xffff C:\Intel\ivecuqnanpnirkt615\tasksche.exe
0x7c900000 0xb2000 0xffff C:\WINDOWS\system32\ntdll.dll
0x7c800000 0xf6000 0xffff C:\WINDOWS\system32\kernel32.dll
0x7e410000 0x91000 0xffff C:\WINDOWS\system32\USER32.dll
0x77f10000 0x49000 0xffff C:\WINDOWS\system32\GDI32.dll
0x77dd0000 0x9b000 0xffff C:\WINDOWS\system32\ADVAPI32.dll
0x77e70000 0x93000 0xffff C:\WINDOWS\system32\RPCRT4.dll
0x77fe0000 0x11000 0xffff C:\WINDOWS\system32\Secur32.dll
0x77c10000 0x58000 0xffff C:\WINDOWS\system32\MSVCRT.dll
0x76390000 0x1d000 0x1 C:\WINDOWS\system32\IMM32.DLL
0x629c0000 0x9000 0x1 C:\WINDOWS\system32\LPK.DLL
0x74d90000 0x6b000 0x1 C:\WINDOWS\system32\USP10.dll
0x77b40000 0x22000 0x1 C:\WINDOWS\system32\Apphelp.dll
0x77c00000 0x8000 0x1 C:\WINDOWS\system32\VERSION.dll
0x68000000 0x36000 0x1 C:\WINDOWS\system32\rsaenh.dll
0x7c9c0000 0x818000 0x1 C:\WINDOWS\system32\SHELL32.dll
0x77f60000 0x76000 0x3 C:\WINDOWS\system32\SHLWAPI.dll
0x773d0000 0x103000 0x2 C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.2600.6028_x-ww_61e65202\comctl32.dll
0x76080000 0x65000 0x1 C:\WINDOWS\system32\MSVCP60.dll
0x77690000 0x21000 0x1 C:\WINDOWS\system32\NTMART4.DLL
0x774e0000 0x13e000 0x1 C:\WINDOWS\system32\ole32.dll
0x71b10000 0x13000 0x1 C:\WINDOWS\system32\SAMLIB.dll
0x76f60000 0x2c000 0x1 C:\WINDOWS\system32\WLDAP32.dll
0x769c0000 0xb4000 0x1 C:\WINDOWS\system32\USERENV.dll
0x5ad70000 0x38000 0x2 C:\WINDOWS\system32\uxtheme.dll

sansforensics@siftworkstation -> ~/D/volatility-master
```

Image 10: dlllist plugin for pid 1940

vol.py -f wcry.raw --profile=WinXPSP2x86 dlllist -p 740

```
sansforensics@siftworkstation -> ~/D/volatility-master
$ vol.py -f wcry.raw --profile=WinXPSP2x86 dlllist -p 740
Volatility Foundation Volatility Framework 2.6
*****
@WanaDecryptor@ pid: 740
Command line : @WanaDecryptor@.exe
Service Pack 3

Base_PIR-ThreatIntel- Size LoadCount Path
-----
0x00400000 0x3d000 0xffff C:\Intel\ivecuqnanpnirkt615\@WanaDecryptor@.exe
0x7c900000 0xb2000 0xffff C:\WINDOWS\system32\ntdll.dll
0x7c800000 0xf6000 0xffff C:\WINDOWS\system32\kernel32.dll
0x73dd0000 0xf2000 0xffff C:\WINDOWS\system32\MFC42.DLL
0x77c10000 0x58000 0xffff C:\WINDOWS\system32\msvcrt.dll
0x77f10000 0x49000 0xffff C:\WINDOWS\system32\GDI32.dll
0x7e410000 0x91000 0xffff C:\WINDOWS\system32\USER32.dll
0x77dd0000 0x9b000 0xffff C:\WINDOWS\system32\ADVAPI32.dll
0x77e70000 0x93000 0xffff C:\WINDOWS\system32\RPCRT4.dll
0x77fe0000 0x11000 0xffff C:\WINDOWS\system32\Secur32.dll
0x7c9c0000 0x818000 0xffff C:\WINDOWS\system32\SHELL32.dll
0x77f60000 0x76000 0xffff C:\WINDOWS\system32\SHLWAPI.dll
0x773d0000 0x103000 0xffff C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.2600.6028_x-ww_61e65202\COMCTL32.dll
0x77120000 0x8b000 0xffff C:\WINDOWS\system32\OLEAUT32.dll
0x774e0000 0x13e000 0xffff C:\WINDOWS\system32\ole32.dll
0x78130000 0x134000 0xffff C:\WINDOWS\system32\urlmon.dll
0x3df00000 0x1ec000 0xffff C:\WINDOWS\system32\iertutil.dll
0x76080000 0x65000 0xffff C:\WINDOWS\system32\MSVCP60.dll
0x71ab0000 0x17000 0xffff C:\WINDOWS\system32\WS2_32.dll
0x71aa0000 0x8000 0xffff C:\WINDOWS\system32\WS2HELP.dll
0x3d930000 0xe7000 0xffff C:\WINDOWS\system32\WININET.dll
0x00340000 0x9000 0xffff C:\WINDOWS\system32\Normaliz.dll
0x76390000 0x1d000 0x4 C:\WINDOWS\system32\IMM32.DLL
0x629c0000 0x9000 0x1 C:\WINDOWS\system32\LPK.DLL
0x74d90000 0x6b000 0x2 C:\WINDOWS\system32\USP10.dll
0x732e0000 0x5000 0x1 C:\WINDOWS\system32\RICHED32.DLL
```

Image 11: dlllist plugin for plugin 740

We identify the path of the binary for process tasksche.exe and it looks uncommon and suspicious. It's recommended to look at the DLLs loaded to understand the characteristics of the process like encryption, registry modification and socket creation etc. Process

@WanaDecryptor@ with PID 740 also uses the same path of process tasksche.exe. Based on DLLs loaded by @WanaDecryptor@ process, it can perform socket creation (Ws2\_32.dll), high level network communications (WININET.DLL), querying registry (ADVAPI32.DLL), encryption (SECURE32.DLL) and interacting with browsers (URLMON.DLL) like internet explorer etc.

Next, we analyze handles.

```
vol.py -f wcry.raw --profile=WinXPSP3x86 handles -p 1940 -t key
```

```
vol.py -f wcry.raw --profile=WinXPSP3x86 handles -p 1940 -t Mutant
```

```
vol.py -f wcry.raw --profile=WinXPSP3x86 handles -p 1940 -t File
```

```
vol.py -f wcry.raw --profile=WinXPSP3x86 handles -p 740 -t key
```

Looking at the handles of PID 1940, the process has created a mutex. **Mutexes have long been used by malware authors to prevent more than one instance of the malware running on the same machine. An old anti-malware trick consists in the creation of a specific mutex, to prevent the execution of a specific malware.** The mutex is named "MsWinZonesCacheCounterMutexA" and it can be one of IOCs for identifying infected systems.

Like mutex as one of types of handles for any process, volatility *handles* plugin can also identify File, Key, Event, threads and port type of handles for any process. A quick look at files accessed by PID 1940.

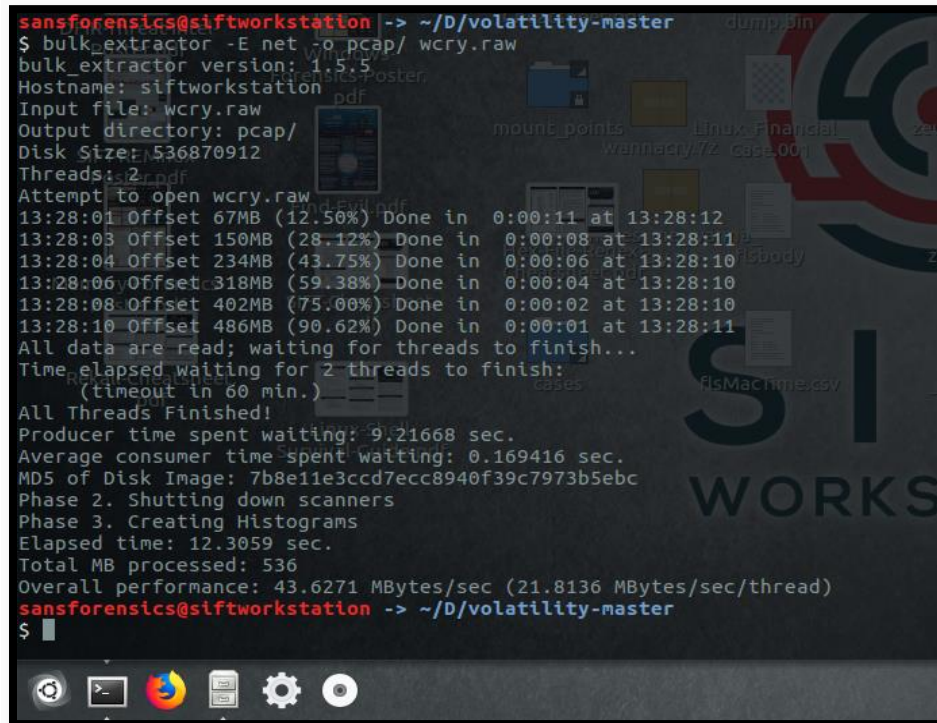
Volatility *ethscan* plugin can also extract pcap from memory dump.

Here, we use Bulk\_Extractor on the raw dump to extract network activity as a pcap and



analyze it using Wireshark.

**bulk\_extractor -E net -o pcap/ wcry.raw**



```
sansforensics@siftworkstation -> ~/D/volatility-master
$ bulk_extractor -E net -o pcap/ wcry.raw
bulk_extractor version: 1.5.5
Hostname: siftworkstation
Input file: wcry.raw
Output directory: pcap/
Disk Size: 536870912
Threads: 2
Attempt to open wcry.raw
13:28:01 Offset 67MB (12.50%) Done in 0:00:11 at 13:28:12
13:28:03 Offset 150MB (28.12%) Done in 0:00:08 at 13:28:11
13:28:04 Offset 234MB (43.75%) Done in 0:00:06 at 13:28:10
13:28:06 Offset 318MB (59.38%) Done in 0:00:04 at 13:28:10
13:28:08 Offset 402MB (75.00%) Done in 0:00:02 at 13:28:10
13:28:10 Offset 486MB (90.62%) Done in 0:00:01 at 13:28:11
All data are read; waiting for threads to finish...
Time elapsed waiting for 2 threads to finish:
(timeout in 60 min.)
All Threads Finished!
Producer time spent waiting: 9.21668 sec.
Average consumer time spent waiting: 0.169416 sec.
MD5 of Disk Image: 7b8e11e3ccd7ecc8940f39c7973b5ebc
Phase 2. Shutting down scanners
Phase 3. Creating Histograms
Elapsed time: 12.3059 sec.
Total MB processed: 536
Overall performance: 43.6271 MBytes/sec (21.8136 MBytes/sec/thread)
sansforensics@siftworkstation -> ~/D/volatility-master
$
```


Image 12: extracting pcap using bulk\_extractor

The IP addresses can be used as IoC.

Next, volatility plugin *memdump* was used to dump the address space of @WanaDecryptor@ and taskssche.exe processes for any indicators.

**vol.py -f wcry.raw --profile=WinXPSP3x86 memdump -p1940,740 -D memdumps/**



A terminal window with a dark background and light-colored text. The prompt is 'sansforensics@siftworkstation'. The user enters '~ /D/volatility-master'. Then '\$ mkdir memdumps'. Then 'sansforensics@siftworkstation -> ~/D/volatility-master'. Then '\$ vol.py -f wcry.raw --profile=WinXPSP3x86 memdump -p1940,740 -D memdumps/'. The output shows 'Volatility Foundation Volatility Framework 2.6' followed by a separator line of asterisks. Then 'Writing tasksche.exe [ 1940] to 1940.dmp' followed by another separator line. Then 'Writing @WanaDecryptor@ [ 740] to 740.dmp'. Then 'sansforensics@siftworkstation -> ~/D/volatility-master'. Then '\$ cd memdumps/'. Then 'sansforensics@siftworkstation -> ~/D/v/memdumps'. Then '\$ ls'. The output shows '1940.dmp 740.dmp'. Then 'sansforensics@siftworkstation -> ~/D/v/memdumps'. Then '\$'. At the bottom, there are icons for 'Network' and 'Computer'.

```
sansforensics@siftworkstation -> ~/D/volatility-master
$ mkdir memdumps
sansforensics@siftworkstation -> ~/D/volatility-master
$ vol.py -f wcry.raw --profile=WinXPSP3x86 memdump -p1940,740 -D memdumps/
Volatility Foundation Volatility Framework 2.6
*****
Writing tasksche.exe [ 1940] to 1940.dmp
*****
Writing @WanaDecryptor@ [ 740] to 740.dmp
sansforensics@siftworkstation -> ~/D/volatility-master
$ cd memdumps/
sansforensics@siftworkstation -> ~/D/v/memdumps
$ ls
1940.dmp 740.dmp
sansforensics@siftworkstation -> ~/D/v/memdumps
$
```

Image 13: using memdump plugin to get dumps for pid's 1940 and 740

Inspecting the stings of process tasksche.exe (PID 1940), we find that tasksche.exe started the @WanaDecryptor@ process with cli arguments.

Further, it was found that the @WanaDecryptor@ process using scripts, set up a registry key for itself in the Run key for persistence mechanism and killed few services like DB, MS Exchange etc .

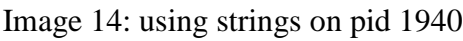


Image 15: using strings on pid 740

**Timeline:**

Date	Event
System created	05/13/08 6:18:43 PM
Possible phishing AIM convo	06/10/08 6:09AM → 06:11AM
First email Jean received from the attacker masquerading as Alison and asking for background check information.	07/19/2008 7:40:36 PM
Jean replied to the attacker's email that she will send it but she didn't	07/19/2008 7:44:28 PM
Jean received an email from the attacker who shows urgency asking for the confidential information to be sent to him as soon as possible. The Return-Path is modified to 'tuckgorge@gmail.com'	07/19/2008 9:26:11 PM
M57biz.xls created	07/19/2008 9:28:03 PM
Jean attach M57biz.xls to the reply email she received from Alison with the Return-Path modified to the attacker's email.	07/19/2008 9:28:47 PM
The attacker sends an email to Jean thanking her and asking her not to tell anyone.	07/20/2008 01:03:5 AM
Jean receives an email from the real Alison telling her that something strange is going and asking her if she knows anything about it.	07/20/2008 07:56:38 PM
Jean receives an email from Bob asking her why about his SSN being on the Internet.	07/20/2008 07:56:38 PM
Jean receives an email from Carol asking her if she populate the database that has the employees information	07/20/2008 08:16:39 PM
Jean replays to Carol with an email denying that she knows anything	07/20/2008 08:45:02 PM

NOTE: An infected system was isolated, binary and memory dumps were taken and analyzed to be the Wannacry Ransomware.

**Conclusion:**

The aim of this incident response exercise was to find how a confidential document from the CFO of M57.biz computer's landed on the competitor's website. A proper Incident Response strategy was followed to find conclusive evidence of a breach using Encase, FTK and Volatility as well as other tools. File signature analysis was done as well as email/internet artifacts were discovered using Encase. A large number of files and other evidence were recovered from Jean's (CFO's) computer using both Encase as well as FTK. FTK assisted in analyzing email applications and gathering clues about email data by studying email headers. Evidence found led the investigators to the conclusion that a phishing and impersonation attack led to the compromise of sensitive information.

During the investigation, a system was found infected by Ransomware and it was critically isolated to obtain the malicious binary as well as the memory dump. Strings was used to find information from the binary as well as indicators of compromise. Volatility was used and a proper analysis was conducted to find which processes were critical for the ransomware to function and a behavioral study of the implication of the ransomware was conducted. It was concluded that the ransomware was of the Wannacry family and the Indicators of Compromise were discovered while studying the infection vectors found in the memory dump using Volatility.

Future directions for this project would be solving the case using tools other than Encase and Volatility and comparing the journey with the current use of Encase and Volatility. Methods of isolating malicious binaries and extracting volatile memory while the malware is running can be studied and documented. Signature analysis for the files could be conducted to get a complete idea. A Timeline formation for the malware's behavior could be done using Volatility.