

# Lab - Memory Forensics Using the Volatility Framework

## Overview

In this lab, you will learn how to perform a forensic analysis of a Windows memory acquisition. Memory acquisition is the method of capturing and dumping the contents of a volatile content into a non-volatile storage device to preserve it for further investigation. A ram analysis can only be successfully conducted when the acquisition has been performed accurately without corrupting the volatile memory image.

In this lab, you will use the Volatility Framework to analyze a memory dump captured from a Windows XP machine infected with malware known as Cridex.

Cridex is a sophisticated strain of banking malware that can steal banking credentials and other personal information on an infected system to access financial records.

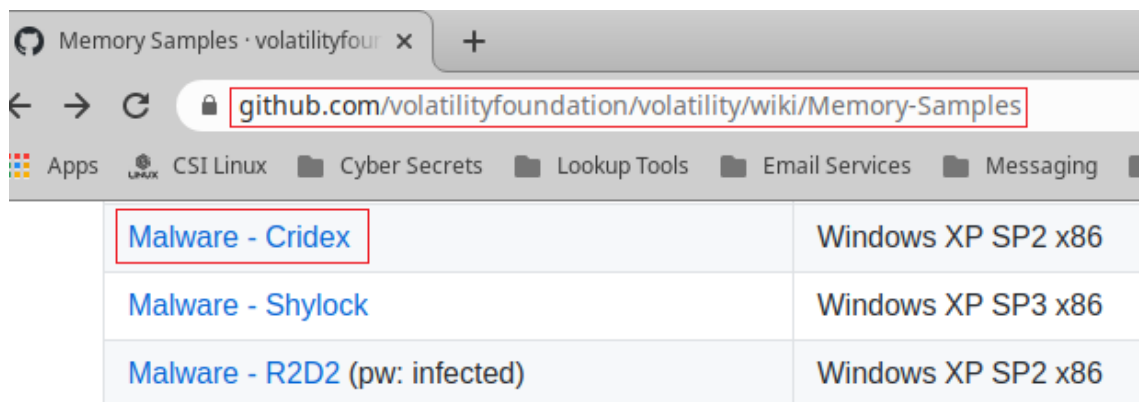
## Lab Requirements

For this lab demonstration, we will be using CSI Linux though Kali Linux could also be used. The steps in the lab are the same, regardless.

- One virtual install of either CSI Linux or Kali. Ensure both have the latest updates and upgrades.
- Internet connection.
- Cridex memory dump file from GitHub.

From the desktop of your CSI Linux or Kali, open a browser and download the **Cridex** memory dump file from [GitHub](https://github.com/volatilityfoundation/volatility/wiki/Memory-Samples).

<https://github.com/volatilityfoundation/volatility/wiki/Memory-Samples>

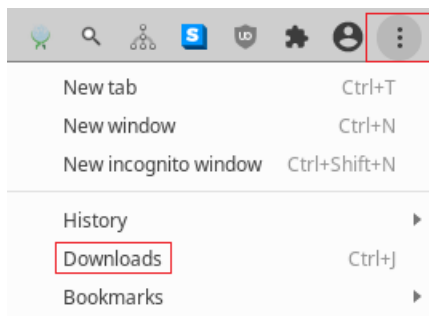


Malware - Cridex	Windows XP SP2 x86
Malware - Shylock	Windows XP SP3 x86
Malware - R2D2 (pw: infected)	Windows XP SP2 x86

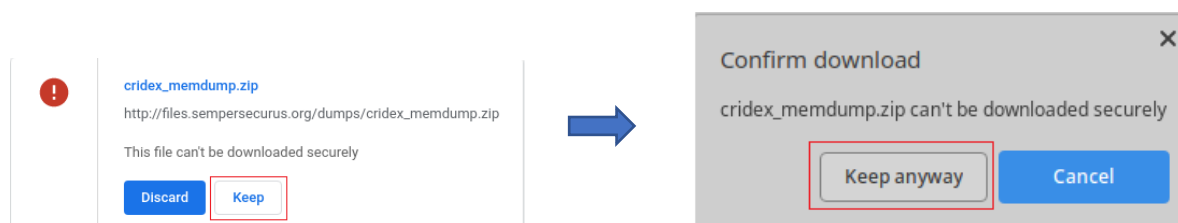
Once the files finish downloading, you can extract the archive to your desktop.

Scroll to the bottom of your browser window. Find each of the downloaded archives. Click the up arrow, and from the context menu, select Show in folder.

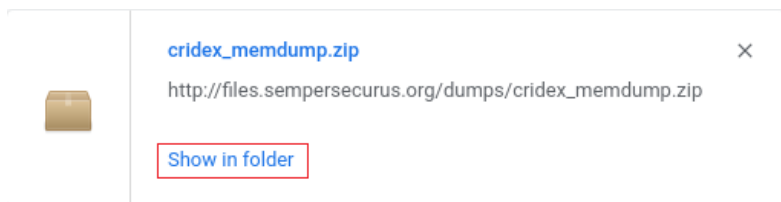
From your browser's taskbar, click on preferences, and from the context menu, click on downloads.



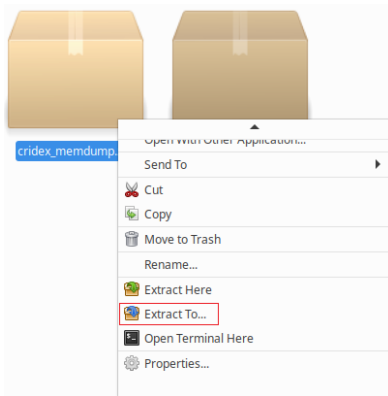
Find the downloaded archive, Select the keep option and then select keep anyway.



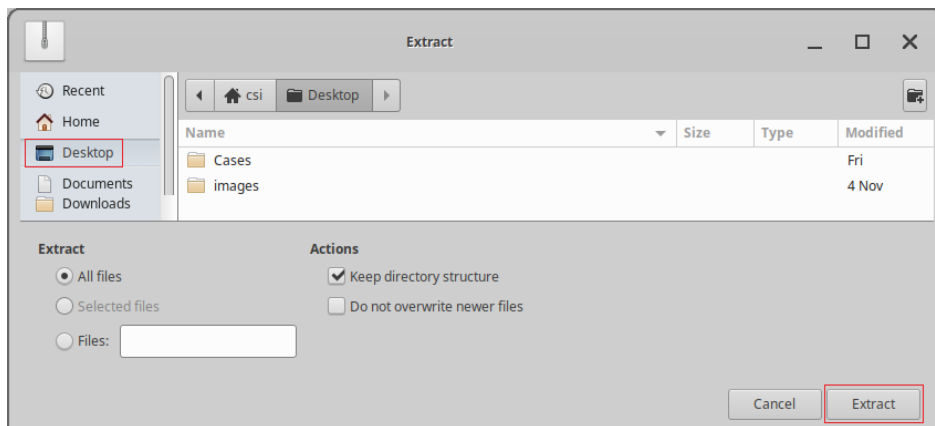
Click to show in folder.



In the download folder, right-click on the downloaded archive, and from the context menu, select Extract To....

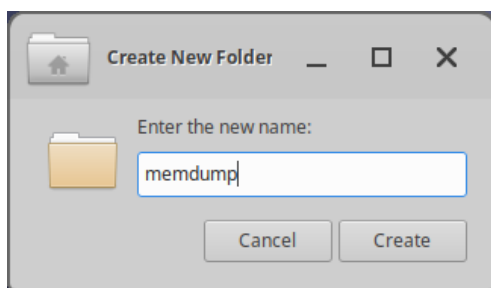


On the next screen, from the left windowpane, select desktop. Scroll to the bottom and click on the Extract button.

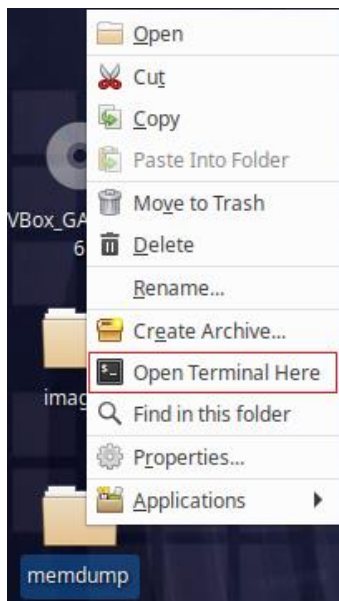


Closeout all open windows and return to your desktop.

Right-click on your desktop and create a new folder. Call the folder **memdump**



Drag and drop the extracted cridx.vmem file into the memdump folder. Right-click on the memdump folder, and from the context menu, select **Open Terminal Here**. This will be our working directory. At the prompt, type **ls** to see the contents of your working directory.



```
Terminal - csi@csi-analyst: ~/Desktop/memdump
File Edit View Terminal Tabs Help
csi@csi-analyst:~/Desktop/memdump$ ls
crindex.vmem
csi@csi-analyst:~/Desktop/memdump$
```

## Lab Update

With the latest version of CSI Linux, you will first need to load support for Volatility by opening a terminal and typing in the following.

```
sudo snap install volatility-phocean
```

```
csi@csi:~/Desktop/membump$ sudo snap install volatility-phocean
```

## Load the volatility.

At the prompt, type in the first three letters of the tool, we need to use, volatility. Press the tab key.

```
csi@csi:~/Desktop/membump$ vola ⇐ Press the tab key to complete the command
```

We are given a list of commands with the first three letters; at the prompt, add the letter 'a' to what you already have typed. Press the tab key one more time, and we have our first part of the command.

Give it a space and type or copy and paste the following.

```
-f crindex.vmem imageinfo
```

```
csi@csi:~/Desktop/membump$ volatility -f crindex.vmem imageinfo
```

Press enter.

After a few moments, the terminal comes back with the machine's profile information from which the memory dump was captured.

With **-f** specifying your dump file and **imageinfo** the volatility plugin we used.

```
csi@csi-analyst:~/Desktop/memdump$ volatility cridex.vmem -f cridex.vmem imageinfo
Volatility Foundation Volatility Framework 2.6
INFO : volatility.debug : Determining profile based on KDBG search...
      Suggested Profile(s) : WinXPSP2x86, WinXPSP3x86 (Instantiated with WinXPSP2x86)
                             AS Layer1 : IA32PagedMemoryPae (Kernel AS)
                             AS Layer2 : FileAddressSpace (/home/csi/Desktop/memdump/cridex.vmem)
                             PAE type : PAE
                             DTB : 0x2fe000L
                             KDBG : 0x80545ae0L
      Number of Processors : 1
      Image Type (Service Pack) : 3
      KPCR for CPU 0 : 0xffdf000L
      KUSER SHARED DATA : 0xffdf0000L
      Image date and time : 2012-07-22 02:45:08 UTC+0000
      Image local date and time : 2012-07-21 22:45:08 -0400
csi@csi-analyst:~/Desktop/memdump$
```

With the machine's profile determined, we can begin to analyze what happened to the machine. We start by looking at what processes were running on the machine using the **pslist** plugin.

**volatility -f cridex.vmem --profile=WinXPSP2x86 pslist**

```
$ volatility -f cridex.vmem --profile=WinXPSP2x86
pslistVolatility Foundation Volatility Framework 2.6
Offset(V)  Name                PID   PPID   Thds   Hnds
Sess
-----
-----
0x823c89c8 System                4     0     53    240 -
-----
0x822f1020 smss.exe             368   4      3     19 -
-----
0x822a0598 csrss.exe            584   368    9     326
0
0x82298700 winlogon.exe          608   368   23     519
0
0x81e2ab28 services.exe          652   608   16     243
0
0x81e2a3b8 lsass.exe            664   608   24     330
0
0x82311360 svchost.exe          824   652   20     194
0
0x81e29ab8 svchost.exe          908   652    9     226
0
```

0x823001d0	svchost.exe	1004	652	64	1118
0					
0x821dfda0	svchost.exe	1056	652	5	60
0					
0x82295650	svchost.exe	1220	652	15	197
0					
0x821dea70	explorer.exe	1484	1464	17	415
0					
0x81eb17b8	spoolsv.exe	1512	652	14	113
0					
0x81e7bda0	reader_sl.exe	1640	1484	5	39
0					
0x820e8da0	alg.exe	788	652	7	104
0					
0x821fcda0	wuauclt.exe	1136	1004	8	173
0					
0x8205bda0	wuauclt.exe	1588	1004	5	132
0					

An alternative to the **pslist** plugin can be used to display the processes and their parent processes: **pstree**. We find a process named “reader\_sl.exe” with the “explorer.exe” as parent process (PPID), which was one of the last processes running on the machine.

```
csi@csi-analyst:~/Desktop/memdump$ volatility -f cridex.vmem --profile=WinXPSP2x86 pstree
Volatility Foundation Volatility Framework 2.6
Name                               Pid  PPid  Thds  Hnds  Time
-----
0x823c89c8:System                   4     0    53    240  1970-01-01 00:00:00 UTC+0000
. 0x822f1020:smss.exe               368    4     3     19  2012-07-22 02:42:31 UTC+0000
.. 0x82298700:winlogon.exe          608   368    23    519  2012-07-22 02:42:32 UTC+0000
... 0x81e2ab28:services.exe         652   608    16    243  2012-07-22 02:42:32 UTC+0000
.... 0x821dfda0:svchost.exe         1056   652     5     60  2012-07-22 02:42:33 UTC+0000
..... 0x81eb17b8:spoolsv.exe         1512   652    14    113  2012-07-22 02:42:36 UTC+0000
..... 0x81e29ab8:svchost.exe         908   652     9    226  2012-07-22 02:42:33 UTC+0000
..... 0x823001d0:svchost.exe        1004   652    64   1118  2012-07-22 02:42:33 UTC+0000
..... 0x8205bda0:wuauclt.exe        1588  1004     5    132  2012-07-22 02:44:01 UTC+0000
..... 0x821fcda0:wuauclt.exe        1136  1004     8    173  2012-07-22 02:43:46 UTC+0000
.... 0x82311360:svchost.exe         824   652    20    194  2012-07-22 02:42:33 UTC+0000
.... 0x820e8da0:alg.exe             788   652     7    104  2012-07-22 02:43:01 UTC+0000
.... 0x82295650:svchost.exe        1220   652    15    197  2012-07-22 02:42:35 UTC+0000
... 0x81e2a3b8:lsass.exe            664   608    24    330  2012-07-22 02:42:32 UTC+0000
.. 0x822a0598:csrss.exe             584   368     9    326  2012-07-22 02:42:32 UTC+0000
. 0x821dea70:explorer.exe          1484  1464    17    415  2012-07-22 02:42:36 UTC+0000
. 0x81e7bda0:reader_sl.exe         1640  1484     5     39  2012-07-22 02:42:36 UTC+0000
csi@csi-analyst:~/Desktop/memdump$
```

To see any processes that may be attempting to hide, we can use the **psxview** plugin.

```
csi@csi-analyst:~/Desktop/memdump$ volatility -f cridex.vmem --profile=WinXPSP2x86 psxview
Volatility Foundation Volatility Framework 2.6
Offset(P)  Name  PID  pslist  psscan  thrdproc  pspcid  csrss  session  deskthrd  ExitTime
-----
0x02498700 winlogon.exe 608 True True True True True True True
0x02511360 svchost.exe 824 True True True True True True True
0x022e8da0 alg.exe 788 True True True True True True True
0x020b17b8 spoolsv.exe 1512 True True True True True True True
0x0202ab28 services.exe 652 True True True True True True True
0x02495650 svchost.exe 1220 True True True True True True True
0x0207bda0 reader_sl.exe 1640 True True True True True True True
0x025001d0 svchost.exe 1004 True True True True True True True
0x02029ab8 svchost.exe 908 True True True True True True True
0x023fcd00 wuauclt.exe 1136 True True True True True True True
0x0225bda0 wuauclt.exe 1588 True True True True True True True
0x0202a3b8 lsass.exe 664 True True True True True True True
0x023dea70 explorer.exe 1484 True True True True True True True
0x023dfda0 svchost.exe 1056 True True True True True True True
0x024f1020 smss.exe 368 True True True True False False False
0x025c89c8 System 4 True True True True False False False
0x024a0598 csrss.exe 584 True True True True False True True
csi@csi-analyst:~/Desktop/memdump$
```

Unless you see false in the first two columns, no hidden processes are running. We see none.

Once we have identified our running processes, we can next check for running sockets and open connections. To do this, we will call upon three different volatility plugins, **connscan**, **netscan**, and **sockets**.

We begin with the **connscan** plugin.

```
volatility -f cridex.vmem --profile=WinXPSP2x86 connscan
```

```
csi@csi-analyst:~/Desktop/memdump$ volatility -f cridex.vmem --profile=WinXPSP2x86 connscan
Volatility Foundation Volatility Framework 2.6
Offset(P)  Local Address  Remote Address  Pid
-----
0x02087620 172.16.112.128:1038 41.168.5.140:8080 1484
0x023a8008 172.16.112.128:1037 125.19.103.198:8080 1484
csi@csi-analyst:~/Desktop/memdump$
```

Next, we use the **sockets** plugins.

Using the **connscan** plugin, we discovered two TCP connections using the process ID (PID)1484. (we learn by looking at our command history outputs that the PID 1484 is assigned to explorer.exe). We can see that one TCP connection is still open, the one using port 1038 and communicating with the destination IP address 41.168.5.140.



```
csi@csi-analyst:~/Desktop/memdump$ volatility -f cridex.vmem --profile=WinXPSP2x86 sockets
Volatility Foundation Volatility Framework 2.6
Offset(V)      PID      Port  Proto Protocol      Address      Create Time
-----
0x81ddb780     664      500    17  UDP           0.0.0.0      2012-07-22 02:42:53 UTC+0000
0x82240d08     1484     1038    6  TCP           0.0.0.0      2012-07-22 02:44:45 UTC+0000
0x81dd7618     1220     1900    17  UDP          172.16.112.128 2012-07-22 02:43:01 UTC+0000
0x82125610      788     1028    6  TCP          127.0.0.1     2012-07-22 02:43:01 UTC+0000
0x8219cc08       4       445    6  TCP           0.0.0.0      2012-07-22 02:42:31 UTC+0000
0x81ec23b0      908      135    6  TCP           0.0.0.0      2012-07-22 02:42:33 UTC+0000
0x82276878       4       139    6  TCP          172.16.112.128 2012-07-22 02:42:38 UTC+0000
0x82277460       4       137    17  UDP          172.16.112.128 2012-07-22 02:42:38 UTC+0000
0x81e76620     1004     123    17  UDP          127.0.0.1     2012-07-22 02:43:01 UTC+0000
0x82172808      664       0    255 Reserved    0.0.0.0      2012-07-22 02:42:53 UTC+0000
0x81e3f460       4       138    17  UDP          172.16.112.128 2012-07-22 02:42:38 UTC+0000
0x821f0630     1004     123    17  UDP          172.16.112.128 2012-07-22 02:43:01 UTC+0000
0x822cd2b0     1220     1900    17  UDP          127.0.0.1     2012-07-22 02:43:01 UTC+0000
0x82172c50      664     4500    17  UDP           0.0.0.0      2012-07-22 02:42:53 UTC+0000
0x821f0d00       4       445    17  UDP           0.0.0.0      2012-07-22 02:42:31 UTC+0000
csi@csi-analyst:~/Desktop/memdump$
```

The **netscan** plugin is not supported with our XP current profile.

```
csi@csi-analyst:~/Desktop/memdump$ volatility -f cridex.vmem --profile=WinXPSP2x86 netscan
Volatility Foundation Volatility Framework 2.6
ERROR : volatility.debug : This command does not support the profile WinXPSP2x86
csi@csi-analyst:~/Desktop/memdump$
```

Both the first two plugins, **consoles** that extract command history by scanning for `_CONSOLE_INFORMATION` and **cmdscan** extracts command history by scanning for `_COMMAND_HISTORY` did not contain any information in their buffers.

Using the **cmdline** plugin displays process command-line arguments, and we now have the full path of the processes launched with PID 1484 and 1640, the “Reader\_sl.exe” process.

```
$ volatility -f cridex.vmem --profile=WinXPSP2x86 cmdline
Volatility Foundation Volatility Framework 2.6
*****
****
System pid:      4
*****
****
smss.exe pid:    368
Command line : \SystemRoot\System32\smss.exe
*****
****
csrss.exe pid:   584
Command line : C:\WINDOWS\system32\csrss.exe
ObjectDirectory=\Windows SharedSection=1024,3072,512 Windows=On
SubSystemType=Windows ServerDll=basesrv,1
ServerDll=winsrv:UserServerDllInitialization,3
ServerDll=winsrv:ConServerDllInitialization,2
ProfileControl=Off MaxRequestThreads=16
*****
```



```

*****
winlogon.exe pid:      608
Command line : winlogon.exe
*****
*****
services.exe pid:     652
Command line : C:\WINDOWS\system32\services.exe
*****
*****
lsass.exe pid:        664
Command line : C:\WINDOWS\system32\lsass.exe
*****
*****
svchost.exe pid:      824
Command line : C:\WINDOWS\system32\svchost -k DcomLaunch
*****
*****
svchost.exe pid:      908
Command line : C:\WINDOWS\system32\svchost -k rpcss
*****
*****
svchost.exe pid:      1004
Command line : C:\WINDOWS\System32\svchost.exe -k netsvcs
*****
*****
svchost.exe pid:      1056
Command line : C:\WINDOWS\system32\svchost.exe -k
NetworkService
*****
*****
svchost.exe pid:      1220
Command line : C:\WINDOWS\system32\svchost.exe -k LocalService
*****
*****
explorer.exe pid:     1484
Command line : C:\WINDOWS\Explorer.EXE
*****
*****
spoolsv.exe pid:      1512
Command line : C:\WINDOWS\system32\spoolsv.exe
*****
*****
reader_sl.exe pid:    1640
Command line : "C:\Program Files\Adobe\Reader
9.0\Reader\Reader_sl.exe"
*****
*****

```

```

alg.exe pid:      788
Command line : C:\WINDOWS\System32\alg.exe
*****
****
wuauc.lt.exe pid:  1136
Command line : "C:\WINDOWS\system32\wuauc.lt.exe"
/RunStoreAsComServer
Local\[3ec]SUSDSb81eb56fa3105543beb3109274ef8ec1
*****
****
wuauc.lt.exe pid:  1588
Command line : "C:\WINDOWS\system32\wuauc.lt.exe"

```

We know the process using PID 1640 was launched by the explorer process and is supposed to be a classic Adobe reader application; however, we have observed this running connection being launched towards an external IP address of 41.168.5.140 by this very same process.

We next need to dump the executable for further analysis later on. We will use the following two plugins, **procdump**, and **memdump**, by specifying the **-p 1640** (the executables PID) and **--dump-dir** (the directory where we want to extract these dumps).

We first extract the reader\_sl.exe executable for later analysis using the following command. Note the period at the from of the command. This tells volatility to save the output to the same directory as it is currently working from, our work folder called memdumps.

```
volatility -f cridex.vmem --profile=WinXPSP2x86 procdump -p 1640 --dump-dir .
```

```

csi@csi-analyst:~/Desktop/memdumps$ volatility -f cridex.vmem --profile=WinXPSP2x86 procdump -p 1640 --dump-dir .
Volatility Foundation Volatility Framework 2.6
Process(V) ImageBase Name Result
-----
0x81e7bda0 0x00400000 reader_sl.exe OK: executable.1640.exe
csi@csi-analyst:~/Desktop/memdumps$

```

We have extracted the executable using the PID of 1640 to our work folder.

We now have a file copy of the executable.1640.exe and by using the following command, we will have a memory dump of what the executable was doing.

```
volatility -f cridex.vmem --profile=WinXPSP2x86 memdump -p 1640 --dump-dir .
```

```

csi@csi-analyst:~/Desktop/memdumps$ volatility -f cridex.vmem --profile=WinXPSP2x86 memdump -p 1640 --dump-dir .
Volatility Foundation Volatility Framework 2.6
*****
Writing reader_sl.exe [ 1640] to 1640.dmp
csi@csi-analyst:~/Desktop/memdumps$

```

To analyze the memory dump of the executable, reader\_sl.exe, we can use the Linux sting command. This string command will pull apart the memory dump, giving us a detailed look at the executable was doing each time it launched.

I used the grep command combined with the -C #NUMBER to get the previous and next lines, thus giving us more context for the information found. We want the information as it relates to the executable communicating with the outside IP address.

```
strings 1640.dmp | grep -Fi "41.168.5.140" -C 5
```

```
csi@csi-analyst:~/Desktop/memdumps$ strings 1640.dmp | grep -Fi "41.168.5.140" -C 5
ABACFPFPENFDECFCFPFHFEFFFPACAB
DpI8
POST /zb/v_01_a/in/ HTTP/1.1
Accept: */*
User-Agent: Mozilla/5.0 (Windows; U; MSIE 7.0; Windows NT 6.0; en-US)
Host: 41.168.5.140:8080
Content-Length: 229
Connection: Keep-Alive
Cache-Control: no-cache
>mtvR
`06!
csi@csi-analyst:~/Desktop/memdumps$
```

We now have our confirmation that the executable is forcing a connection with the outside IP address of 41.168.5.140.

Inside the memory dump is a long list of banking domains. By telling the string command to show use the information inside the memory dump, one page at a time, we can better analyze the contents of the dump.

To do this, we use the following command.

```
strings 1640.dmp | less
```

Each time you press the space bar on your keyboard, the next page of information is loaded. Keep pressing the space bar until you come to the list of banking domains.

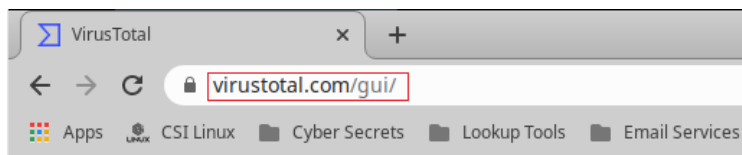
```
$ strings 1640.dmp | less...
*hsbc.co.uk*
*ablv.com*
*accessbankplc.com*
*alphabank.com.cy*
*baltikums.com*
*baltikums.eu*
*banesco.com.pa*
*bankaustria.at*
*banknet.lv*
*bankofcyprus.com*
*bobibanking.com*
*butterfieldonline.ky*
*cimbanque.net*
```

```
*cs.directnet.com*
*directnet.com*
*danskebanka.lv*
*ebank.laiki.com*
*ebank.rcbcy.com*
*ebemo.bemobank.com*
*e-norvik.lv*
*eurobankefg.com*
*eurobankefg.com.cy*
*ekp.lv*
*fbmedirect.com*
*hbllibank.com*
*hellenicnetbanking.com*
*hiponet.lv*
*hkbea*
*ibanka.seb.lv*
*ib.baltikums.com*
*geonline.lv*
*ib.dnb.lv*
*bib.lv*
*ib.snoras.com*
*i-linija.lt*
*loyalbank.com*
*marfinbank.com.cy*
*multinetbank.eu*
*nordea.com*
*secure.ltbbank.com*
*secure.ltblv.com*
*swedbank.lv*
*norvik.lv*
*online.alphabank.com.cy*
*online.citadele.lv*
*online.lkb.lv*
...
```

The more evidence we gather, the more it appears that our reader\_sl.exe executable is up to no good. The next step is to see if the executable we extracted is malicious or not.

To do this, we can call upon some online tools that analyze potentially malicious executables. One such online tool is [www.virustotal.com](http://www.virustotal.com)

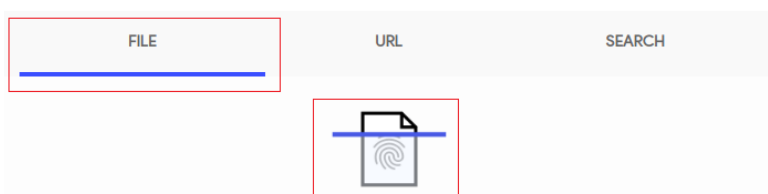
Minimize your CSI Linux or Kali terminal and open a browser. Point your browser to <https://www.virustotal.com/gui/>



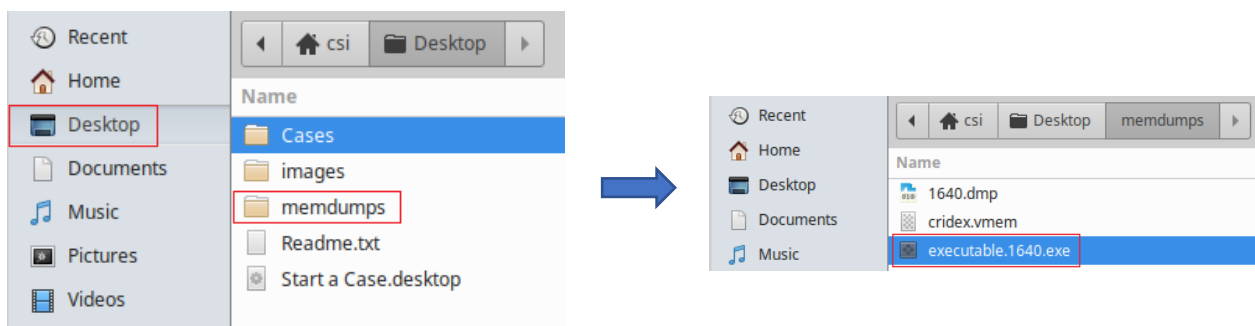
Select file and click on the upload option.



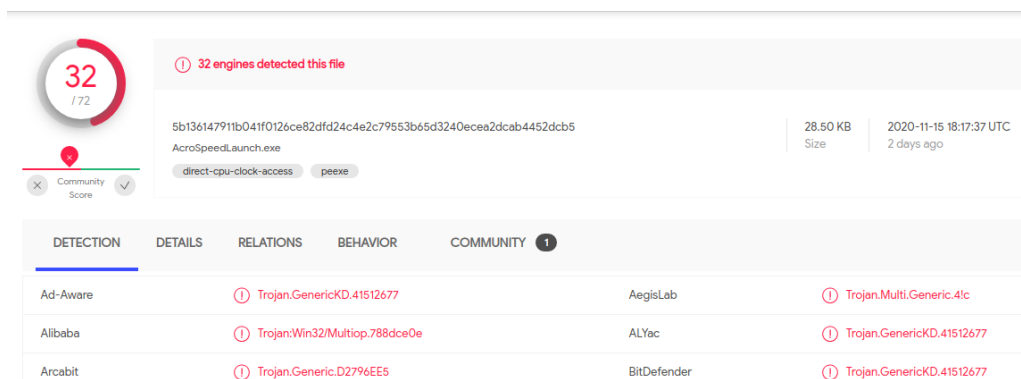
Analyze suspicious files and URLs to detect types of malware, automatically share them with the security community



On the next screen, browse to your memdump directory, find the file named 'executable.1640.exe.'



X2 click the file to upload. 32 out of 72 virus scanners have detected this executable as being malicious.



We confirmed that read\_sl.exe is a piece of malware. But what is launching the payload at each startup, and how can we delete it.

To find this out, we need to look at the machine's registry and what is occurring at each startup. Suppose this Cridex malware is found on other machines. In that case, you should either use an antivirus that can detect and eradicate the threat (using the anti-viruses that we're able to detect our malicious executable) and see if this malicious trojan is persistent or not.

Most malware will look to see if the infected files are still present at every system startup. Sophisticated malware will also communicate with the executable every few minutes to ensure it is still present. If it has been removed, they will rebuild the executable but this time using a different naming. If you remove the executable without removing the payload that rebuilds it each time it is cleaned from the machine, you are just going in circles.

To see if this is the case with Cridex, we can look at the registry entries used during the system startup.

Bring back up your terminal, and at the prompt, type in the following. You can type q for quit to get back your prompt.

**volatility -f cridex.vmem --profile=WinXPSP2x86 hivelist**

```
$ volatility -f cridex.vmem --profile=WinXPSP2x86 hivelist

Volatility Foundation Volatility Framework 2.6
Virtual      Physical      Name
-----
0xe18e5b60 0x093f8b60 \Device\HarddiskVolume1\Documents and
Settings\Robert\Local Settings\Application
Data\Microsoft\Windows\UsrClass.dat
0xe1a19b60 0x0a5a9b60 \Device\HarddiskVolume1\Documents and
Settings\Robert\NTUSER.DAT
0xe18398d0 0x08a838d0 \Device\HarddiskVolume1\Documents and
Settings\LocalService\Local Settings\Application
Data\Microsoft\Windows\UsrClass.dat
0xe18614d0 0x08e624d0 \Device\HarddiskVolume1\Documents and
Settings\LocalService\NTUSER.DAT
0xe183bb60 0x08e2db60 \Device\HarddiskVolume1\Documents and
Settings\NetworkService\Local Settings\Application
Data\Microsoft\Windows\UsrClass.dat
0xe17f2b60 0x08519b60 \Device\HarddiskVolume1\Documents and
Settings\NetworkService\NTUSER.DAT
0xe1570510 0x07669510
\Device\HarddiskVolume1\WINDOWS\system32\config\software
0xe1571008 0x0777f008
\Device\HarddiskVolume1\WINDOWS\system32\config\default
```



```

0xe15709b8 0x076699b8
\Device\HarddiskVolume1\WINDOWS\system32\config\SECURITY
0xe15719e8 0x0777f9e8
\Device\HarddiskVolume1\WINDOWS\system32\config\SAM
0xe13ba008 0x02e4b008 [no name]
0xe1035b60 0x02ac3b60
\Device\HarddiskVolume1\WINDOWS\system32\config\system
0xe102e008 0x02a7d008 [no name]

```

The **hivelist** plugin allows us to print the list of registry hives. The **printkey** plugin will help us to see the content of a registry key, its subkeys, and values. We can use the -K option to navigate towards the registry key path we are looking for.

```

volatility -f cridex.vmem --profile=WinXPSP2x86 printkey -K
"Software\Microsoft\Windows\CurrentVersion\Run"

```

```

$ volatility -f cridex.vmem --profile=WinXPSP2x86 printkey -K
"Software\Microsoft\Windows\CurrentVersion\Run"Volatility
Foundation Volatility Framework 2.6
Legend: (S) = Stable (V) = Volatile

-----
Registry: \Device\HarddiskVolume1\Documents and
Settings\Robert\Local Settings\Application
Data\Microsoft\Windows\UsrClass.dat
Key name: Run (S)
Last updated: 2011-04-13 00:55:13 UTC+0000

Subkeys:

Values:
-----
Registry: \Device\HarddiskVolume1\Documents and
Settings\Robert\NTUSER.DAT
Key name: Run (S)
Last updated: 2012-07-22 02:31:51 UTC+0000

Subkeys:

Values:
REG_SZ KB00207877.exe : (S) "C:\Documents and
Settings\Robert\Application Data\KB00207877.exe"
-----
Registry:
\Device\HarddiskVolume1\WINDOWS\system32\config\default
Key name: Run (S)
Last updated: 2011-04-12 20:31:49 UTC+0000

```

Subkeys:

Values:

-----

Registry: \Device\HarddiskVolume1\Documents and  
Settings\LocalService\Local Settings\Application  
Data\Microsoft\Windows\UsrClass.dat

Key name: Run (S)

Last updated: 2011-04-13 00:55:13 UTC+0000

Subkeys:

Values:

-----

Registry: \Device\HarddiskVolume1\Documents and  
Settings\NetworkService\NTUSER.DAT

Key name: Run (S)

Last updated: 2011-04-13 00:49:16 UTC+0000

Subkeys:

Values:

-----

Registry: \Device\HarddiskVolume1\Documents and  
Settings\LocalService\NTUSER.DAT

Key name: Run (S)

Last updated: 2011-04-13 00:49:28 UTC+0000

Subkeys:

Values:

-----

Registry: \Device\HarddiskVolume1\Documents and  
Settings\NetworkService\Local Settings\Application  
Data\Microsoft\Windows\UsrClass.dat

Key name: Run (S)

Last updated: 2011-04-13 00:55:13 UTC+0000

As you can see, the only hive that has been recently modified is the following registry  
“\Device\HarddiskVolume1\Documents and Settings\Robert\NTUSER.DAT”. Let us confirm  
that the concerned executable named “KB00207877.exe” is linked with our trojan:

```
strings 1640.dmp | grep -Fi "KB00207877.exe"
```

```
csi@csi-analyst:~/Desktop/memdumps$ strings 1640.dmp | grep -Fi "KB00207877.exe"
KB00207877.exe
C:\Documents and Settings\Robert\Application Data\KB00207877.exe(,
KB00207877.EXEn
KB00207877.exe
KB00207877.exe
C:\Documents and Settings\Robert\Application Data\KB00207877.exe(,
csi@csi-analyst:~/Desktop/memdumps$
```

Since the executable is found in our trojan memory dump, we are now sure that Cridex modified the starting up registry key of the victim's computer to make itself persistent. Deleting this "KB00207877.exe" is needed to make a good cleanup of the infected machine.

End of the lab!