

Memory Forensic using Rekall

Coreflood

R2D2

Cridex

12/10/2021

Abstract

The usage of the Rekall framework gives an insight to a style of memory analysis unlike its predecessors. It allows users to analyze the effects of malware in live time. The memory forensic report encompasses three malware, Coreflood, R2D2, and Cridex. Usage of volatility based framework Rekall is a potent tool. This paper provides critical analysis of three dangerous malware, surveys the tool of memory forensics and provides insight of malware's intricacy.

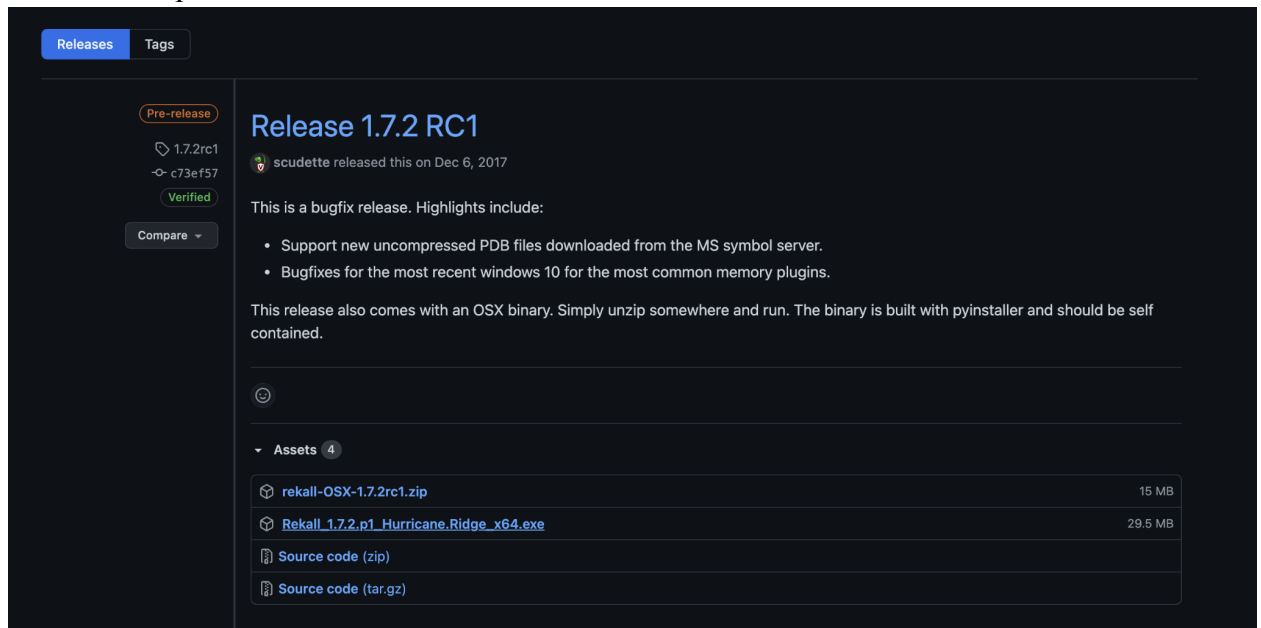
Table of Contents

Rekall Installation Guide	3
Initial Failure Documentation	Error! Bookmark not defined.
Winpmem_2.0.1	7
winpmem_v3.3.rc3	9
winpmem_mini_x64_rc2.exe	11
winpmem-2.1.post4.exe	16
Background Information	18
Key Definitions	19
Analyzing Malwares 1	20
Coreflood	20
Analyzing Malwares 2	28
R2D2	28
Analyzing Malwares 3	32
Cridex	32
References	35

Rekall Installation Guide

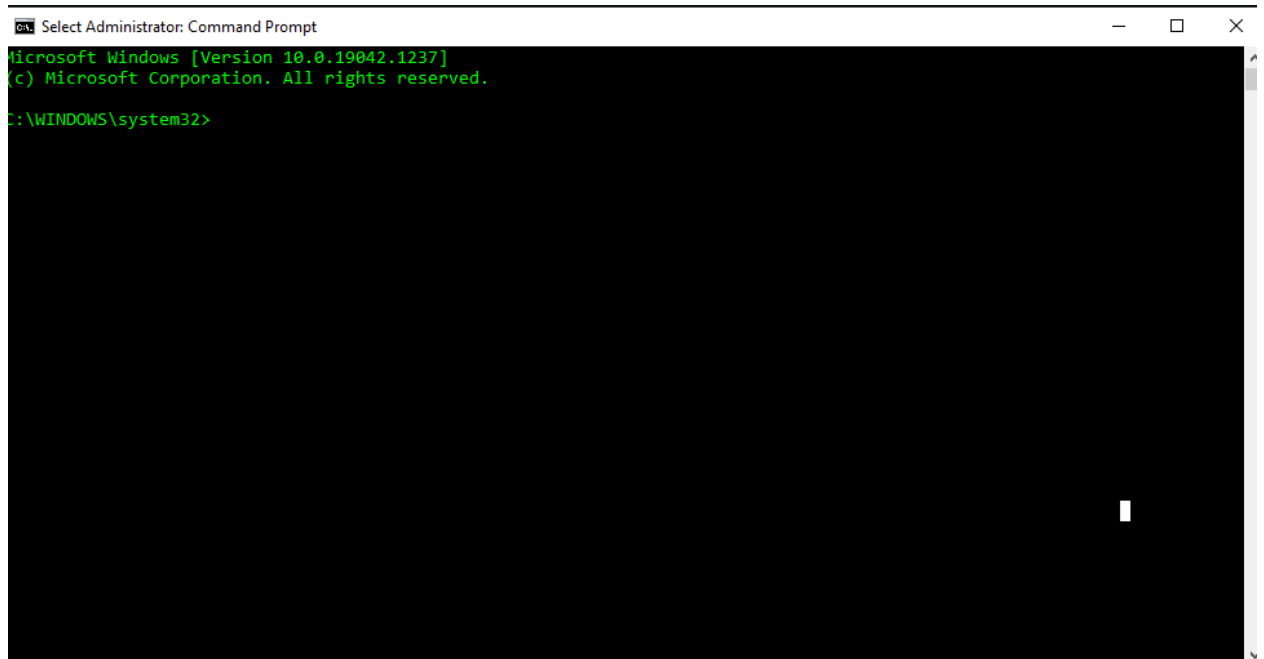
Windows:

1. Make sure you have python downloaded. Preferably Python 3.6 or below
 - a. Link: <https://www.python.org/downloads/release/python-360/>
2. Your Python should have installed pip as well.
3. If for some reason, you do not have pip, install pip by
 - a. Open CMD, type in “curl <https://bootstrap.pypa.io/get-pip.py> -o get-pip.py”
 - b. Once its done, type in “python get-pip.py”
4. Go to <https://github.com/google/rekall/releases>
5. Click the drop down menu on “Assets”



6. Install “Rekall_1.7.2.p1_Hurricane.Ridge_x64.exe”
7. Run through the installation process.
 - a. If it lets you choose the download destination, download it under **C:\Program Files**. **NOT C:\Program Files(x86)**

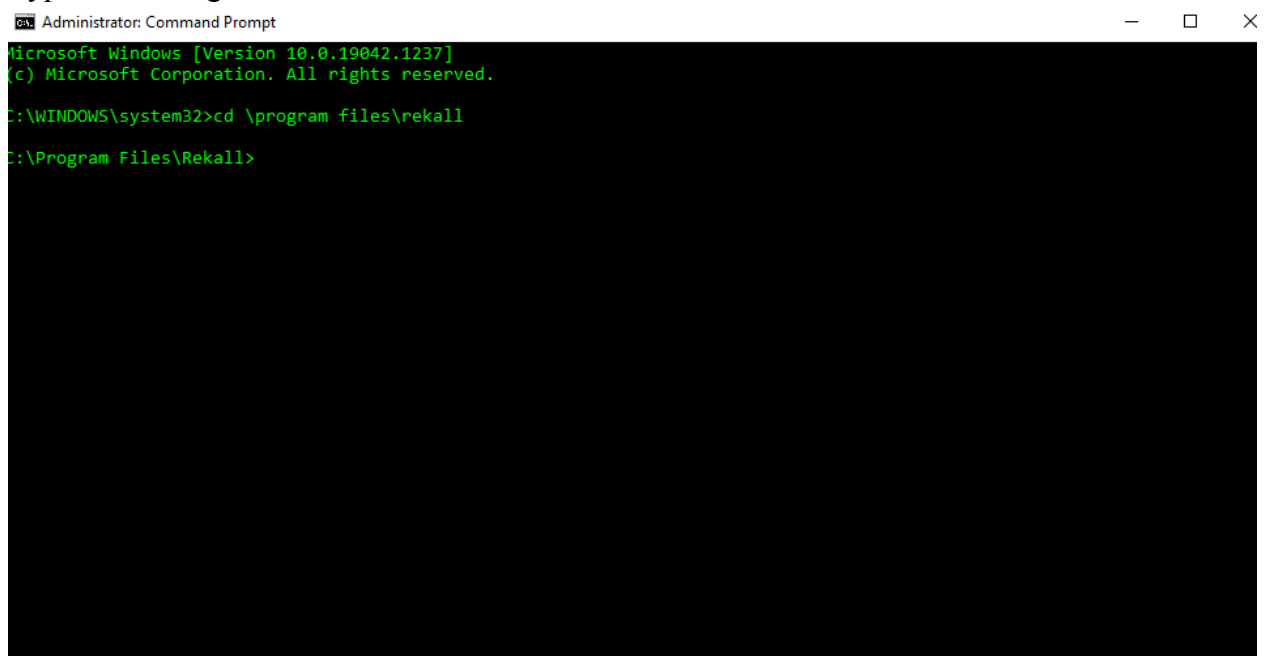
8. Open up your CMD in administrator mode.



A screenshot of a Windows Command Prompt window titled "Select Administrator: Command Prompt". The window has standard Windows window controls (minimize, maximize, close) in the top right corner. The text inside the window is as follows:

```
Microsoft Windows [Version 10.0.19042.1237]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\WINDOWS\system32>
```

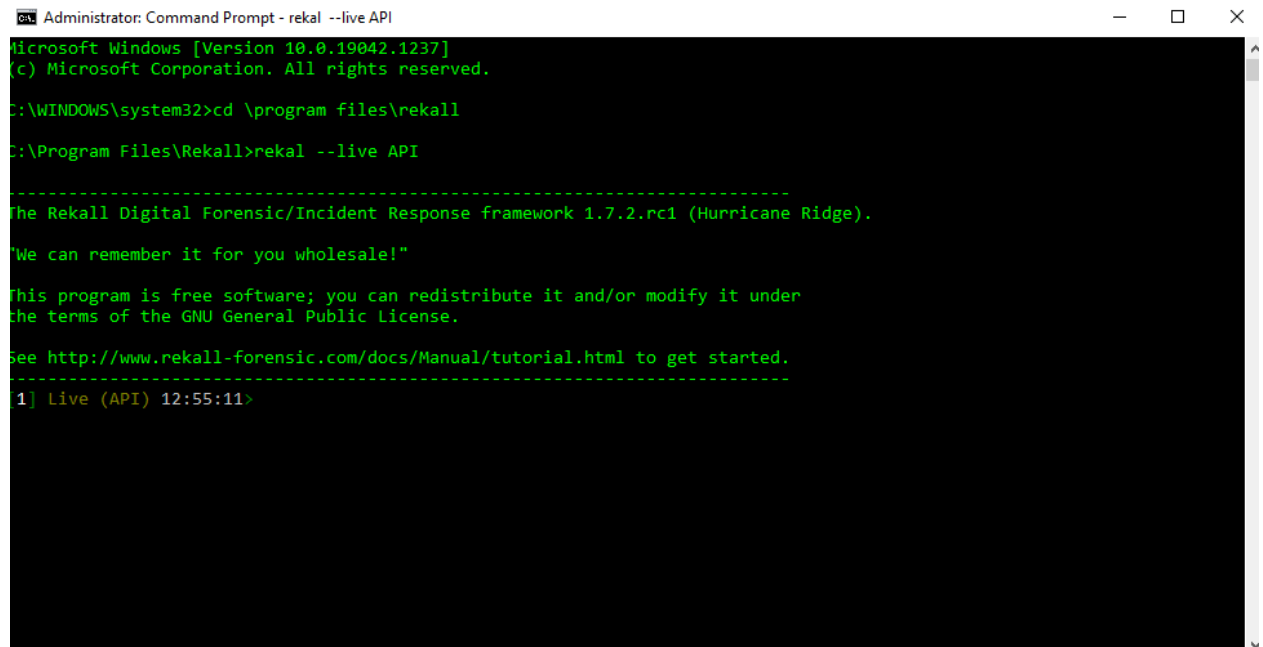
9. Type in `cd \Program Files\Rekall`



A screenshot of a Windows Command Prompt window titled "Administrator: Command Prompt". The window has standard Windows window controls (minimize, maximize, close) in the top right corner. The text inside the window is as follows:

```
Microsoft Windows [Version 10.0.19042.1237]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\WINDOWS\system32>cd \program files\rekall  
  
C:\Program Files\Rekall>
```

10. Type `rekal --live API` (make sure API is all capital letters)



```
Administrator: Command Prompt - recal --live API
Microsoft Windows [Version 10.0.19042.1237]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd \program files\rekall

C:\Program Files\Rekall>rekal --live API

-----
The Rekall Digital Forensic/Incident Response framework 1.7.2.rc1 (Hurricane Ridge).

"We can remember it for you wholesale!"

This program is free software; you can redistribute it and/or modify it under
the terms of the GNU General Public License.

See http://www.rekall-forensic.com/docs/Manual/tutorial.html to get started.
-----
1] Live (API) 12:55:11>
```

Initial Failure Documentation

Based on the Rekall official documentation and other sources, they seem to grab a .dmp, .img or .aff4 file to analyze.

Winpmem is required to acquire memory images.

There are multiple versions of Winpmem floating around on the internet. For example, “winpmem_2.0.1”, “winpmem_v3.3.rc3”, “winpmem_mini_x64_rc2.exe”, and “winpmem-2.1.post4.exe”.

Environment Sidenote

- Rekall is located under C:\Program Files\Rekall
- All of the different versions of winpmem is downloaded under C:\Program Files
- All commands are run under administrator privileges

Winpmem_2.0.1

Displaying -h output

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.19042.1288]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd c:\program files

c:\Program Files>winpmem_2.0.1.exe -h

USAGE:

    winpmem_2.0.1.exe [-l] [-u] [--write-mode] [--mode <MmMapIoSpace,
PhysicalMemory, PTERemapping>] [--driver <Path to
driver.>] [--elf] [-m] [-p </path/to/pagefile>] ...
[-V] [-d] [-v] [-t] [-i </path/to/file/or/device>]
... [--category <string>] [-e <string>] [-o
</path/to/file>] [-c <zlib, snappy, none>] [--]
[--version] [-h] </path/to/aff4/volume> ...

Where:

-l, --load-driver
    Load the driver and exit

-u, --unload-driver
    Unload the driver and exit

--write-mode
    Enable write mode. You must have the driver compiled with write
support and be on a system with test signing enabled.
```

Attempt to create .aff4 file: Fail

```
Administrator: Command Prompt

c:\Program Files>winpmem_2.0.1.exe -o output.aff4
Driver Unloaded.
E1107 10:23:13.520354 18160 win_pmem.cc:594] Error: StartService(), Cannot start the driver:A device attached to the sys
tem is not functioning.
E1107 10:23:13.520354 18160 pmem_imager.cc:165] Imaging failed with error: -8
```

Attempt to load the driver: Fail

```
Administrator: Command Prompt

c:\Program Files>winpmem_2.0.1.exe -l
Driver Unloaded.
E1107 10:25:00.424062 18140 win_pmem.cc:594] Error: StartService(), Cannot start the driver:A device attached to the system is not functioning.
E1107 10:25:00.424062 18140 pmem_imager.cc:165] Imaging failed with error: -8

c:\Program Files>
```

Attempt to unload the driver: **Success**

```
Administrator: Command Prompt

c:\Program Files>winpmem_2.0.1.exe -u
Driver Unloaded.
```


winpmem_v3.3.rc3

Displaying -h output

```
Administrator: Command Prompt
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd c:\program files

c:\Program Files>winpmem_v3.3.rc3.exe -h

USAGE:

winpmem_v3.3.rc3.exe [-L] [-U] [--write-mode] [--mode <MmMapIoSpace,
PhysicalMemory, PTERemapping>] [--driver <Path to
driver.>] [--format <map, elf, raw>]
[--volume_format <aff4, raw>] [-m] [-p
</path/to/pagefile>] ... [-V] [-l] [-d] ... [-v]
[-t] [-i </path/to/file/or/device>] ...
[--relative] [-e <string>] [--logfile <string>]
[-D <path to directory>] [-o </path/to/file>] [-s
<Size (E.g. 100Mb)>] [-c <deflate, zlib, snappy,
lz4, none>] [--threads <(default 2)>] [--]
[--version] [-h] </path/to/aff4/volume> ...

Where:

-L, --load-driver
    Load the driver and exit

-U, --unload-driver
    Unload the driver and exit

--write-mode
    Enable write mode. You must have the driver compiled with write
    support and be on a system with test signing enabled.

--mode <MmMapIoSpace, PhysicalMemory, PTERemapping>
    Select the acquisition mode. Default is PTERemapping.

--driver <Path to driver.>
    Use this driver instead of the included one. This option is rarely
    used.

--format <map, elf, raw>
    Specify the output format of memory streams:
```

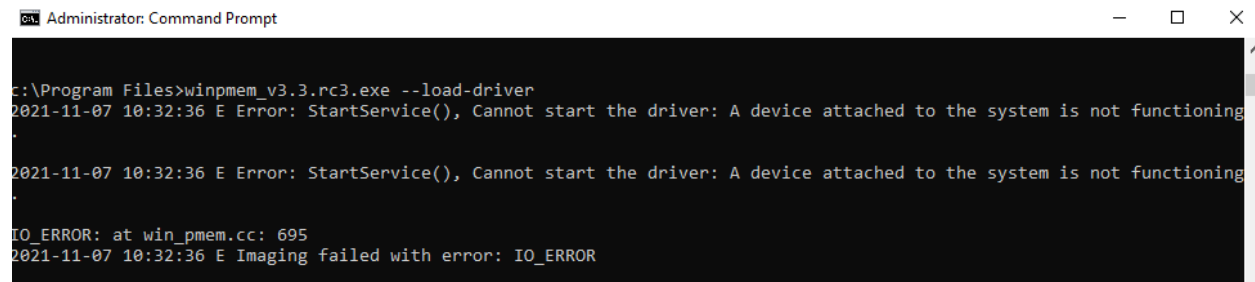
Attempt to create .aff4 file: **Fail**

```
c:\Program Files>winpmem_v3.3.rc3.exe -o output.aff4
2021-11-07 10:31:43 E Error: StartService(), Cannot start the driver: A device attached to the system is not functioning
.

2021-11-07 10:31:43 E Error: StartService(), Cannot start the driver: A device attached to the system is not functioning
.

IO_ERROR: at win_pmem.cc: 695
2021-11-07 10:31:43 E Imaging failed with error: IO_ERROR
```

Attempt to load the driver: **Fail**



```
Administrator: Command Prompt

c:\Program Files>winpmem_v3.3.rc3.exe --load-driver
2021-11-07 10:32:36 E Error: StartService(), Cannot start the driver: A device attached to the system is not functioning
.

2021-11-07 10:32:36 E Error: StartService(), Cannot start the driver: A device attached to the system is not functioning
.

IO_ERROR: at win_pmem.cc: 695
2021-11-07 10:32:36 E Imaging failed with error: IO_ERROR
```

Attempt to unload the driver: **Fail**



```
Administrator: Command Prompt

c:\Program Files>winpmem_v3.3.rc3.exe -U
```

winpmem_mini_x64_rc2.exe

Displaying -h output

```
Administrator: Command Prompt

C:\WINDOWS\system32>cd c:\program files
c:\Program Files>winpmem_mini_x64_rc2.exe -h
WinPmem64
Winpmem - A memory imager for windows.
Copyright Michael Cohen (scudette@gmail.com) 2012-2014.

Version 2.0.1 Oct 13 2020
Usage:
    winpmem_mini_x64_rc2.exe [option] [output path]

Option:
    -l    Load the driver and exit.
    -u    Unload the driver and exit.
    -d [filename]
           Extract driver to this file (Default use random name).
    -h    Display this help.
    -w    Turn on write mode.
    -0    Use MmMapIoSpace method.
    -1    Use \\Device\\PhysicalMemory method (Default for 32bit OS).
    -2    Use PTE remapping (AMD64 only - Default for 64bit OS).

NOTE: an output filename of - will write the image to STDOUT.

Examples:
winpmem_mini_x64_rc2.exe physmem.raw
Writes an image to physmem.raw
```

Attempt to create .aff4 file: Fail

```
Administrator: Command Prompt

c:\Program Files>winpmem_mini_x64_rc2.exe -o output.aff4
WinPmem64
Winpmem - A memory imager for windows.
Copyright Michael Cohen (scudette@gmail.com) 2012-2014.

Version 2.0.1 Oct 13 2020
Usage:
    winpmem_mini_x64_rc2.exe [option] [output path]

Option:
    -l    Load the driver and exit.
    -u    Unload the driver and exit.
    -d [filename]
           Extract driver to this file (Default use random name).
    -h    Display this help.
    -w    Turn on write mode.
    -0    Use MmMapIoSpace method.
    -1    Use \\Device\\PhysicalMemory method (Default for 32bit OS).
    -2    Use PTE remapping (AMD64 only - Default for 64bit OS).

NOTE: an output filename of - will write the image to STDOUT.

Examples:
winpmem_mini_x64_rc2.exe physmem.raw
Writes an image to physmem.raw
```

Administrator: Command Prompt

```
c:\Program Files>winpmem_mini_x64_rc2.exe -l
WinPmem64
Extracting driver to C:\Users\KICHAN~1\AppData\Local\Temp\pmeE8E9.tmp
Driver Unloaded.
Loaded Driver C:\Users\KICHAN~1\AppData\Local\Temp\pmeE8E9.tmp.
Deleting C:\Users\KICHAN~1\AppData\Local\Temp\pmeE8E9.tmp
```

Attempt to unload the driver: **Success**

Administrator: Command Prompt

```
c:\Program Files>winpmem_mini_x64_rc2.exe -u
WinPmem64
Driver Unloaded.
```

Attempt to create .raw file: **Success**

```
c:\Program Files>winpmem_mini_x64_rc2.exe physmem.raw
WinPmem64
Extracting driver to C:\Users\KICHAN~1\AppData\Local\Temp\pmeBC13.tmp
Driver Unloaded.
Loaded Driver C:\Users\KICHAN~1\AppData\Local\Temp\pmeBC13.tmp.
Deleting C:\Users\KICHAN~1\AppData\Local\Temp\pmeBC13.tmp
The system time is: 16:41:48
Will generate a RAW image
- buffer_size_: 0x1000
CR3: 0x00001AD002
7 memory ranges:
Start 0x00001000 - Length 0x00057000
Start 0x00059000 - Length 0x00046000
Start 0x00100000 - Length 0xA22F1000
Start 0xA23F3000 - Length 0x06ED6000
Start 0xA9611000 - Length 0x00170000
Start 0xAAEFF000 - Length 0x00001000
Start 0x100000000 - Length 0x34F000000
max_physical_memory_ 0x44f000000
Acquisition mode PTE Remapping
Padding from 0x00000000 to 0x00001000
pad
- length: 0x1000

00% 0x00000000 .
copy_memory
- start: 0x1000
- end: 0x58000
```

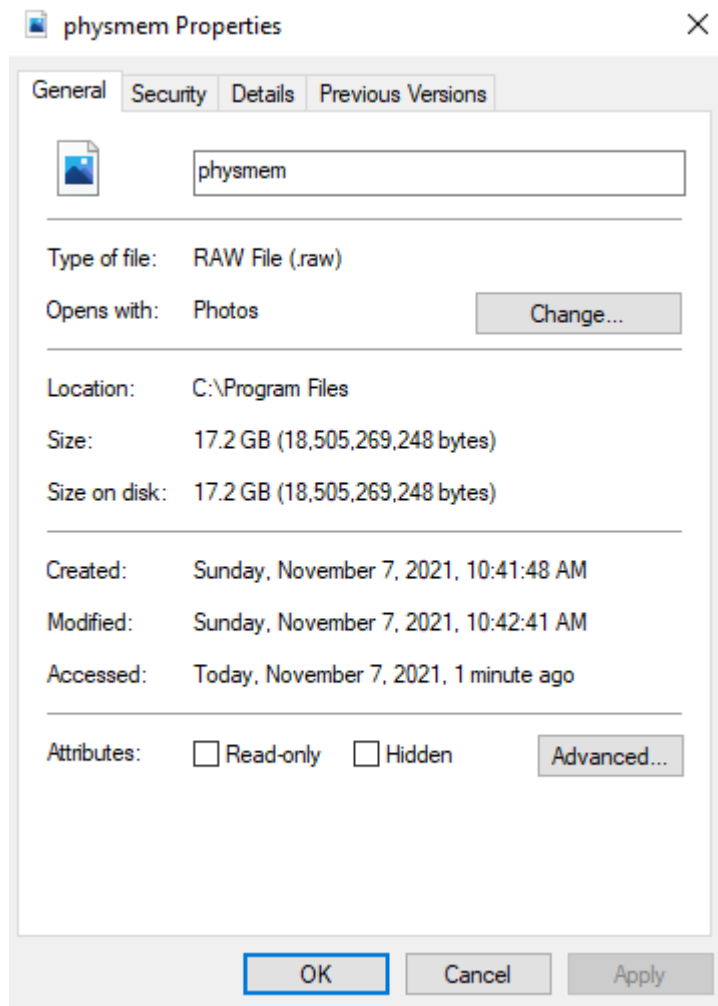
...

```
Padding from 0xAAF00000 to 0x100000000
bad
- length: 0x55100000

15% 0xAAF00000 .....
15% 0xAAF00000 .....
copy_memory
- start: 0x100000000
- end: 0x44f000000

23% 0x100000000 .....XXX.....
27% 0x132000000 .....
32% 0x164000000 .....
36% 0x196000000 .....
41% 0x1C8000000 .....
45% 0x1FA000000 .....
50% 0x22C000000 .....x.....
54% 0x25E000000 .....
59% 0x290000000 .....
64% 0x2C2000000 .....
68% 0x2F4000000 .....
73% 0x326000000 .....x.....
77% 0x358000000 .....
82% 0x38A000000 .....
86% 0x3BC000000 .....
91% 0x3EE000000 .....
95% 0x420000000 .....X.....
The system time is: 16:42:41
Driver Unloaded.
```

Output file of physmem.raw has been created under C:\Program Files



Attempting to analyze physmem.raw with rekall threw back an error because it couldn't find the path of physmem.raw. I moved the physmem.raw file inside of the rekall folder.

Outcome: [Partial Success](#)

I ran `Rekal -f physmem.raw` then `netstat` then `pslist`

```
Administrator: Command Prompt - rekall -f physmem.raw
C:\Program Files\Rekall>rekall -f physmem.raw

-----
The Rekall Digital Forensic/Incident Response framework 1.7.2.rc1 (Hurricane Ridge).

We can remember it for you wholesale!"

This program is free software; you can redistribute it and/or modify it under
the terms of the GNU General Public License.

See http://www.rekall-forensic.com/docs/Manual/tutorial.html to get started.
-----
1] physmem.raw 10:46:08> netstat
2021-11-07 10:46:09,598:WARNING:rekall.1:Inventory for repository "http://profiles.rekall-forensic.com" seems malformed.
Are you behind a captive portal or proxy? If this is a custom repository, did you forget to create an inventory? You must
use the tools/profiles/build_profile_repo.py tool with the --inventory flag.
2021-11-07 10:46:09,599:WARNING:rekall.1:Repository http://profiles.rekall-forensic.com will be disabled.
-----> netstat()400000 (0xa00000)
2021-11-07 10:48:32,788:CRITICAL:rekall.1:Traceback (most recent call last):
  File "rekall-core\rekall\session.py", line 866, in RunPlugin
  File "rekall-core\rekall\session.py", line 925, in _GetPluginObj
  File "rekall-core\rekall\obj.py", line 153, in __call__
  File "rekall-lib\rekall_lib\registry.py", line 96, in __call__
  File "rekall-core\rekall\plugins\overlays\windows\tcpip_vtypes.py", line 742, in __init__
AttributeError: 'NoneType' object has no attribute 'profile'

-----
AttributeError                                Traceback (most recent call last)
ipython-input-1-5c1889516dcf> in <module>()
----> 1 netstat()

C:\Program Files\Rekall\rekall\obj.py in __call__(self, *args, **kwargs)
C:\Program Files\Rekall\rekall\session.py in RunPlugin(self, plugin_obj, *args, **kwargs)
C:\Program Files\Rekall\rekall\session.py in RunPlugin(self, plugin_obj, *args, **kwargs)
C:\Program Files\Rekall\rekall\session.py in _GetPluginObj(self, plugin_obj, *args, **kwargs)
```

Pslist returned the correct form but no live processors

```
Administrator: Command Prompt - rekall -f physmem.raw
AttributeError: 'NoneType' object has no attribute 'profile'
1] physmem.raw 10:49:46> pslist
-----> pslist()
   _EPROCESS      name          pid  ppid  thread_count handle_count session_id wow64  process_create_time
process_exit_time
-----
Out<10:49:46> Plugin: pslist (WinPsList)
1] physmem.raw 10:49:46>
```

winpmem-2.1.post4.exe

Displaying -h output

Administrator: Command Prompt

```
C:\WINDOWS\system32>cd c:\program files
C:\Program Files>winpmem-2.1.post4.exe -h
SAGE:
winpmem-2.1.post4.exe [-l] [-u] [--write-mode] [--mode <MmMapIoSpace,
PhysicalMemory, PTERemapping>] [--driver <Path to
driver.>] [--format <map, elf, raw>] [-m] [-p
</path/to/pagefile>] ... [-V] [-d] [-v] [-t] [-i
</path/to/file/or/device>] ... [-e <string>] [-o
</path/to/file>] [-c <zlib, snappy, none>] [--]
[--version] [-h] </path/to/aff4/volume> ...
```

here:

```
-l, --load-driver
    Load the driver and exit

-u, --unload-driver
    Unload the driver and exit

--write-mode
    Enable write mode. You must have the driver compiled with write
    support and be on a system with test signing enabled.

--mode <MmMapIoSpace, PhysicalMemory, PTERemapping>
    Select the acquisition mode. Default is PTERemapping.

--driver <Path to driver.>
    Use this driver instead of the included one. This option is rarely
    used.

--format <map, elf, raw>
    Specify the output format of memory streams:

    map: An AFF4Map object (Supports compression and sparse).

    elf: An ELF stream. (Supports sparse image).

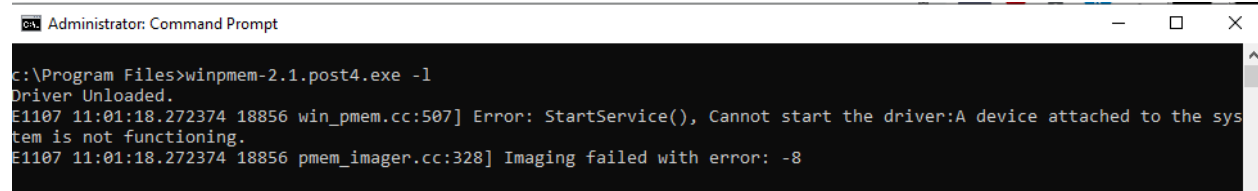
    raw: A raw padded stream. (Padded with no compression).

    If this option is used together with the --export option it specifies
    the output format of the exported stream.
```

Attempt to create .aff4 file: **Fail**

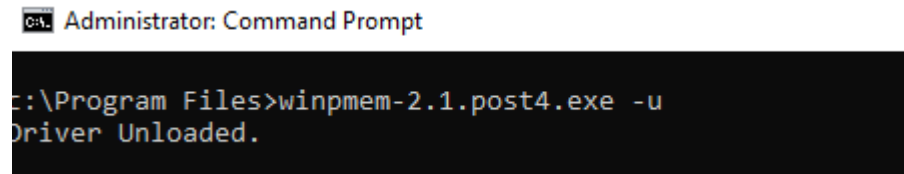
```
C:\Program Files>winpmem-2.1.post4.exe -o output.aff4
Driver Unloaded.
E1107 11:00:50.506300 2028 win_pmem.cc:507] Error: StartService(), Cannot start the driver:A device attached to the sys
tem is not functioning.
E1107 11:00:50.506300 2028 pmem_imager.cc:328] Imaging failed with error: -8
```


Attempt to load the driver: **Fail**



```
C:\Program Files>winpmem-2.1.post4.exe -l
Driver Unloaded.
E1107 11:01:18.272374 18856 win_pmem.cc:507] Error: StartService(), Cannot start the driver:A device attached to the sys
tem is not functioning.
E1107 11:01:18.272374 18856 pmem_imager.cc:328] Imaging failed with error: -8
```

Attempt to unload the driver: **Success**



```
C:\Program Files>winpmem-2.1.post4.exe -u
Driver Unloaded.
```

Background Information

Rekall is an advanced forensic and incident response framework. While it began life purely as a memory forensic framework, it has now evolved into a complete platform. Rekall implements the most advanced analysis techniques in the field, while still being developed in the open, with a free and open source license.

The Rekall framework is plugin based. This is what makes it so extensible. Developers can add many different plugins to implement different analysis techniques and produce different data.

Single Command Example:

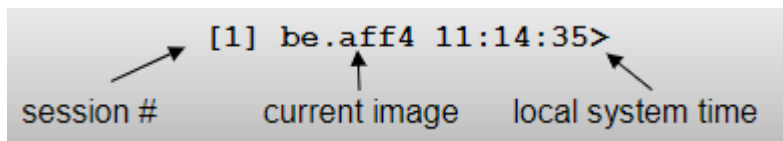
```
>Rekal -f test.aff4 pslist
```

Starting an Interactive Session:

```
>rekal -f test.aff4
```

Starting an Interactive Session(sends output to specified tool):

```
>rekal -f test.aff4 --pager=gedit
```



```
[1] be.aff4 11:14:35>
```

session # current image local system time

Key Definitions

Pslist: list all the processes on windows using a variety of methods. Since it is required by all plugins which have process selectors, this plugin will, by default, list processes using all methods.

Psxview: Find hidden processes with various process listings

Pstree: Shows the parent/child relationship between processes. This plugin prints a parent/child relationship tree by walking the task_struct.children and task_struct.sibling members.

Memdump: To dump all addressable memory in a process, use the memdump plugin. This plugin enumerates the process page tables and writes them out into an external file. An index file is also created which can be used to find the virtual address of each byte in the output file.

Connscan: Similar to the [connections](Connections.html) plugin, this plugin searches from _TCP_OBJECT structs. However, it employs pool scanning techniques.

Sockets: This module enumerates the active sockets from tcpip.sys

Cmdscan: searches the memory of csrss.exe on XP/2003/Vista/2008 and conhost.exe on Windows 7 for commands that attackers entered through a console shell (cmd.exe). This is one of the most powerful commands you can use to gain visibility into an attacker's actions on a victim system, whether they opened cmd.exe through an RDP session or proxied input/output to a command shell from a networked backdoor.

Malfind: helps find hidden or injected code/DLLs in user mode memory, based on characteristics such as VAD tag and page permissions.

Hooks_inline: Detect API hooks in process and kernel memory

Analyzing Malwares 1

Coreflood

About: Coreflood is a trojan horse and botnet created by a group of Russian hackers and released in 2010. The FBI included on its list of infected systems “approximately 17 state or local government agencies, including one police department; three airports; two defense contractors; five banks or financial institutions; approximately 30 colleges or universities; approximately 20 hospital or health care companies; and hundreds of businesses.” It is present on more than 2.3 million computers worldwide and as of May 2011 remains a threat.

To start analyzing Coreflood

```
c:\Program Files>cd rekall
```

```
c:\Program Files\Rekall>rekal --filename coreflood.vmem
```

```
-----  
The Rekall Digital Forensic/Incident Response framework 1.7.2.rc1 (Hurricane Ridge).
```

```
"We can remember it for you wholesale!"
```

```
This program is free software; you can redistribute it and/or modify it under  
the terms of the GNU General Public License.
```

```
See http://www.rekall-forensic.com/docs/Manual/tutorial.html to get started.
```

```
-----  
[1] coreflood.vmem 10:41:42>
```

We want to enumerate the processes

```

1] coreflood.vmem 10:42:22> pslist
2021-12-07 10:42:24,083:WARNING:rekall.1:Inventory for repository "http://profiles.rekall-forensic.com" seems malformed. Are you behind a captive portal or proxy? If this is a custom repository, did you forget to create an inventory? You must use the tools/profiles/build_profile_repo.py tool with the --inventory flag.
2021-12-07 10:42:24,085:WARNING:rekall.1:Repository http://profiles.rekall-forensic.com will be disabled.
-----> pslist()

```

EPROCESS	name	pid	ppid	thread_count	handle_count	session_id	wow64	process_create_time	process_exit_time
0x810b1660	System	4	0	58	183	-	False	-	-
0x80fdc368	logon.scr	124	632	1	15	0	False	2010-08-15 18:21:28Z	-
0xff25a7e0	alg.exe	216	676	6	105	0	False	2010-08-11 06:06:39Z	-
0xff3667e8	VMwareTray.exe	432	1724	1	49	0	False	2010-08-11 06:09:31Z	-
0xff374980	VMwareUser.exe	452	1724	6	189	0	False	2010-08-11 06:09:32Z	-
0x80f94588	wuauclt.exe	468	1028	4	134	0	False	2010-08-11 06:09:37Z	-
0xff2ab020	smss.exe	544	4	3	21	-	False	2010-08-11 06:06:21Z	-
0xff1ecda0	csrss.exe	608	544	10	369	0	False	2010-08-11 06:06:23Z	-
0xff1ec978	winlogon.exe	632	544	20	518	0	False	2010-08-11 06:06:23Z	-
0xff247020	services.exe	676	632	16	269	0	False	2010-08-11 06:06:24Z	-
0xff255020	lsass.exe	688	632	19	344	0	False	2010-08-11 06:06:24Z	-
0xff218230	vmacthlp.exe	844	676	1	24	0	False	2010-08-11 06:06:24Z	-
0x80ff88d8	svchost.exe	856	676	17	199	0	False	2010-08-11 06:06:24Z	-
0xff364310	wscntfy.exe	888	1028	1	27	0	False	2010-08-11 06:06:49Z	-
0xff217560	svchost.exe	936	676	10	272	0	False	2010-08-11 06:06:24Z	-
0x80fbf910	svchost.exe	1028	676	71	1341	0	False	2010-08-11 06:06:24Z	-
0xff38b5f8	TPAutoConnect.e	1084	1968	1	61	0	False	2010-08-11 06:06:52Z	-
0xff22d558	svchost.exe	1088	676	5	80	0	False	2010-08-11 06:06:25Z	-
0xff125020	cmd.exe	1136	1668	0	-	0	False	2010-08-15 18:24:00Z	2010-08-15 18:24:00Z
0xff203b80	svchost.exe	1148	676	14	208	0	False	2010-08-11 06:06:26Z	-
0xff1d7da0	spoolsv.exe	1432	676	13	135	0	False	2010-08-11 06:06:26Z	-
0xff1b8b28	vmtoolsd.exe	1668	676	5	221	0	False	2010-08-11 06:06:35Z	-
0xff3865d0	explorer.exe	1724	1708	12	341	0	False	2010-08-11 06:09:29Z	-
0xff1fdc88	VMUpgradeHelper	1788	676	4	100	0	False	2010-08-11 06:06:38Z	-
0xff143b28	TPAutoConnSvc.e	1968	676	5	100	0	False	2010-08-11 06:06:39Z	-
0xff3ad1a8	IEXPLORE.EXE	2044	1724	10	366	0	False	2010-08-15 18:11:17Z	-

Everything looks fine, but one thing that stands out is the running internet explorer. To dive further into the IEXPLORE.EXE, we check its outbound connection.

To check the outbound connection, I used connsnscan to check out previously terminated and currently active connections.

```
Out<10:45:28> Plugin: pslist (WinPsList)
[1] coreflood.vmem 10:54:45> connsnscan
-----> connsnscan()
tcpip/GUID/9546A8399BAC4717BC41758594EF0D9C2 matched offset 0x4562+0xf3ba9000=0 offset_p      local_net_address
remote_net_address      pid
-----
0xeda590 172.16.176.143:1058      65.54.81.209:80      2044
0x1079e70 172.16.176.143:1082      209.234.234.16:80      2044
0x107c888 172.16.176.143:1059      4.23.40.126:80      2044
0x108fcd8 172.16.176.143:1072      65.55.15.124:80      2044
0x10fa448 172.16.176.143:1065      65.55.253.21:80      2044
0x2214988 172.16.176.143:1092      65.54.81.14:80      2044
0x26c68a8 172.16.176.143:1074      65.55.15.243:80      2044
0x2ae4bb0 172.16.176.143:1073      65.55.15.123:80      2044
0x48b25f0 172.16.176.143:1085      65.55.149.119:80      2044
0x4a045f8 172.16.176.143:1057      65.54.81.49:80      2044
0x4a04e70 172.16.176.143:1095      69.43.160.145:80      2044
0x4a4a4a0 172.16.176.143:1084      12.120.180.24:80      2044
0x4be2558 172.16.176.143:1079      65.54.81.22:80      2044
0x5536e70 172.16.176.143:1090      65.54.81.14:80      2044
0x5802340 172.16.176.143:1062      65.55.18.18:80      2044
0x5c9e200 172.16.176.143:1067      65.54.81.14:80      2044
0x5deea30 172.16.176.143:1068      65.54.81.14:80      2044
0x6015ab0 172.16.176.143:1053      207.46.170.10:80      2044
0x605f208 172.16.176.143:1086      202.89.231.60:80      2044
0x6125538 172.16.176.143:1083      65.54.81.79:80      2044
0x623a438 172.16.176.143:1066      96.6.41.210:80      2044
0x6450720 172.16.176.143:1077      65.55.149.121:80      2044
0x64509f0 172.16.176.143:1063      64.4.18.73:80      2044
0x6497a68 172.16.176.143:1075      65.55.15.124:80      2044
0x67bd218 172.16.176.143:1070      65.54.81.209:80      2044
0x7c17be0 172.16.176.143:1060      65.55.239.161:80      2044
Out<10:54:47> Plugin: connsnscan (ConnScan)
[1] coreflood.vmem 10:54:47>
```

From what we see, it seems like the host had legitimate communications. All the communication was created by pid 2044 which is iexplorer. There is a chance that it could also be fake IP addresses registered by the hackers.

We use sockets to check the inbound connections.

```

[1] coreflood.vmem 11:04:52> sockets
> sockets()
offset_v  pid  port  proto  protocol  address  create_time
-----
0x80fd1008  4    0     47 GRE    0.0.0.0    2010-08-11 06:08:00Z
0xff158c00 2044 1052  17 UDP    127.0.0.1  2010-08-15 18:11:19Z
0xff258008  688  500  17 UDP    0.0.0.0    2010-08-11 06:06:35Z
0xff2984a0 1088 1078  17 UDP    0.0.0.0    2010-08-15 18:11:23Z
0xff367008  4    445   6 TCP    0.0.0.0    2010-08-11 06:06:17Z
0x80ffc128  936  135   6 TCP    0.0.0.0    2010-08-11 06:06:24Z
0xff225b70  688  0     255 Reserved 0.0.0.0    2010-08-11 06:06:35Z
0xff254008 1028 123   17 UDP    127.0.0.1  2010-08-15 18:24:00Z
0x80fce930 1088 1025  17 UDP    0.0.0.0    2010-08-11 06:06:38Z
0xff127d28  216 1026   6 TCP    127.0.0.1  2010-08-11 06:06:39Z
0xff3a97a0 1088 1061  17 UDP    0.0.0.0    2010-08-15 18:11:21Z
0xff12b580 1148 1900  17 UDP    127.0.0.1  2010-08-15 18:24:00Z
0xff1b8250  688 4500  17 UDP    0.0.0.0    2010-08-11 06:06:35Z
0xff382e98  4    1033  6 TCP    0.0.0.0    2010-08-11 06:08:00Z
0x80fbd40  4    445  17 UDP    0.0.0.0    2010-08-11 06:06:17Z
Out<11:04:53> Plugin: sockets (Sockets)

```

Next, let's use the malfind function to possibly find injected code inside the process memory.

Some of the outputs have been truncated for viewing ease.

```
[1] coreflood.vmem 11:07:45> malfind
-----> malfind()
***** of pid 468
Process: csrss.exe Pid: 608 Address: 0x7f6f0000
Vad Tag: Vad Protection: EXECUTE_READWRITE
Protection: 6

0x7f6f0000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..... vad_0x7f6f0000
0x7f6f0010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0x7f6f0020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0x7f6f0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....

----- vad_0x7f6f0000 -----: 0x7f6f0000
0x7f6f0000 0x0 0000 add byte ptr [eax], al
0x7f6f0002 0x2 0000 add byte ptr [eax], al
0x7f6f0004 0x4 0000 add byte ptr [eax], al
0x7f6f0006 0x6 0000 add byte ptr [eax], al
0x7f6f0008 0x8 0000 add byte ptr [eax], al
0x7f6f000a 0xa 0000 add byte ptr [eax], al
0x7f6f000c 0xc 0000 add byte ptr [eax], al
0x7f6f000e 0xe 0000 add byte ptr [eax], al
0x7f6f0010 0x10 0000 add byte ptr [eax], al
0x7f6f0012 0x12 0000 add byte ptr [eax], al
0x7f6f0014 0x14 0000 add byte ptr [eax], al
0x7f6f0016 0x16 0000 add byte ptr [eax], al
0x7f6f0018 0x18 0000 add byte ptr [eax], al
0x7f6f001a 0x1a 0000 add byte ptr [eax], al
0x7f6f001c 0x1c 0000 add byte ptr [eax], al
0x7f6f001e 0x1e 0000 add byte ptr [eax], al
0x7f6f0020 0x20 0000 add byte ptr [eax], al
0x7f6f0022 0x22 0000 add byte ptr [eax], al
0x7f6f0024 0x24 0000 add byte ptr [eax], al
0x7f6f0026 0x26 0000 add byte ptr [eax], al
0x7f6f0028 0x28 0000 add byte ptr [eax], al
0x7f6f002a 0x2a 0000 add byte ptr [eax], al
0x7f6f002c 0x2c 0000 add byte ptr [eax], al
0x7f6f002e 0x2e 0000 add byte ptr [eax], al
0x7f6f0030 0x30 0000 add byte ptr [eax], al
0x7f6f0032 0x32 0000 add byte ptr [eax], al
0x7f6f0034 0x34 0000 add byte ptr [eax], al
0x7f6f0036 0x36 0000 add byte ptr [eax], al
0x7f6f0038 0x38 0000 add byte ptr [eax], al
0x7f6f003a 0x3a 0000 add byte ptr [eax], al
0x7f6f003c 0x3c 0000 add byte ptr [eax], al
0x7f6f003e 0x3e 0000 add byte ptr [eax], al
0x7f6f0040 0x40 0000 add byte ptr [eax], al
0x7f6f0042 0x42 0000 add byte ptr [eax], al
```


Process: winlogon.exe Pid: 632 Address: 0x2c930000
Vad Tag: VadS Protection: EXECUTE_READWRITE
CommitCharge: 4, MemCommit: 1, PrivateMemory: 1, Protection: 6

```
0x2c930000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..... vad_0x2c930000
0x2c930010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0x2c930020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0x2c930030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

----- vad_0x2c930000 -----: 0x2c930000

0x2c930000	0x0 0000	add byte ptr [eax], al
0x2c930002	0x2 0000	add byte ptr [eax], al
0x2c930004	0x4 0000	add byte ptr [eax], al
0x2c930006	0x6 0000	add byte ptr [eax], al
0x2c930008	0x8 0000	add byte ptr [eax], al
0x2c93000a	0xa 0000	add byte ptr [eax], al
0x2c93000c	0xc 0000	add byte ptr [eax], al
0x2c93000e	0xe 0000	add byte ptr [eax], al
0x2c930010	0x10 0000	add byte ptr [eax], al
0x2c930012	0x12 0000	add byte ptr [eax], al
0x2c930014	0x14 0000	add byte ptr [eax], al
0x2c930016	0x16 0000	add byte ptr [eax], al
0x2c930018	0x18 0000	add byte ptr [eax], al
0x2c93001a	0x1a 0000	add byte ptr [eax], al
0x2c93001c	0x1c 0000	add byte ptr [eax], al
0x2c93001e	0x1e 0000	add byte ptr [eax], al
0x2c930020	0x20 0000	add byte ptr [eax], al
0x2c930022	0x22 0000	add byte ptr [eax], al
0x2c930024	0x24 0000	add byte ptr [eax], al
0x2c930026	0x26 0000	add byte ptr [eax], al
0x2c930028	0x28 0000	add byte ptr [eax], al
0x2c93002a	0x2a 0000	add byte ptr [eax], al
0x2c93002c	0x2c 0000	add byte ptr [eax], al
0x2c93002e	0x2e 0000	add byte ptr [eax], al
0x2c930030	0x30 0000	add byte ptr [eax], al
0x2c930032	0x32 0000	add byte ptr [eax], al
0x2c930034	0x34 0000	add byte ptr [eax], al
0x2c930036	0x36 0000	add byte ptr [eax], al
0x2c930038	0x38 0000	add byte ptr [eax], al
0x2c93003a	0x3a 0000	add byte ptr [eax], al
0x2c93003c	0x3c 0000	add byte ptr [eax], al
0x2c93003e	0x3e 0000	add byte ptr [eax], al
0x2c930040	0x40 0000	add byte ptr [eax], al
0x2c930042	0x42 0000	add byte ptr [eax], al
0x2c930044	0x44 0000	add byte ptr [eax], al
0x2c930046	0x46 0000	add byte ptr [eax], al

```

Process: winlogon.exe Pid: 632 Address: 0x37ec0000
Vad Tag: VadS Protection: EXECUTE_READWRITE
CommitCharge: 4, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x37ec0000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..... vad_0x37ec0000
0x37ec0010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0x37ec0020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0x37ec0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....

----- vad_0x37ec0000 -----
0x37ec0000 0x0 0000 add byte ptr [eax], al
0x37ec0002 0x2 0000 add byte ptr [eax], al
0x37ec0004 0x4 0000 add byte ptr [eax], al
0x37ec0006 0x6 0000 add byte ptr [eax], al
0x37ec0008 0x8 0000 add byte ptr [eax], al
0x37ec000a 0xa 0000 add byte ptr [eax], al
0x37ec000c 0xc 0000 add byte ptr [eax], al
0x37ec000e 0xe 0000 add byte ptr [eax], al
0x37ec0010 0x10 0000 add byte ptr [eax], al
0x37ec0012 0x12 0000 add byte ptr [eax], al
0x37ec0014 0x14 0000 add byte ptr [eax], al
0x37ec0016 0x16 0000 add byte ptr [eax], al
0x37ec0018 0x18 0000 add byte ptr [eax], al
0x37ec001a 0x1a 0000 add byte ptr [eax], al
0x37ec001c 0x1c 0000 add byte ptr [eax], al
0x37ec001e 0x1e 0000 add byte ptr [eax], al
0x37ec0020 0x20 0000 add byte ptr [eax], al
0x37ec0022 0x22 0000 add byte ptr [eax], al
0x37ec0024 0x24 0000 add byte ptr [eax], al
0x37ec0026 0x26 0000 add byte ptr [eax], al
0x37ec0028 0x28 0000 add byte ptr [eax], al
0x37ec002a 0x2a 0000 add byte ptr [eax], al
0x37ec002c 0x2c 0000 add byte ptr [eax], al
0x37ec002e 0x2e 0000 add byte ptr [eax], al
0x37ec0030 0x30 0000 add byte ptr [eax], al
0x37ec0032 0x32 0000 add byte ptr [eax], al
0x37ec0034 0x34 0000 add byte ptr [eax], al
0x37ec0036 0x36 0000 add byte ptr [eax], al
0x37ec0038 0x38 0000 add byte ptr [eax], al
0x37ec003a 0x3a 0000 add byte ptr [eax], al
0x37ec003c 0x3c 0000 add byte ptr [eax], al
0x37ec003e 0x3e 0000 add byte ptr [eax], al
0x37ec0040 0x40 0000 add byte ptr [eax], al
0x37ec0042 0x42 0000 add byte ptr [eax], al
0x37ec0044 0x44 0000 add byte ptr [eax], al

```

As you can see, the output of malfind function seems to be benign for now.

Next, we call a IAT hook on the iexplore by `hooks_iat 2044`

```

[1] coreflood.vmem 10:41:09> hooks_iat 2044
-----> hooks_iat(2044)
      source          target          target_func
-----
Process IEXPLORE.EXE (2044)
-----

```

From the function call, I expected to see multiple calls on the hooks further cementing the findings of malware analysis of Coreflood trojan.

Analyzing Malwares 2

R2D2

About: R2D2 is a malicious program belonging to the Crysis/Dharma ransomware family. Systems infected with this malware have their data encrypted and users receive ransom demands for decryption. When encryption is underway, all affected files are renamed according to the following pattern: original filename, unique ID (generated individually for each victim), cyber criminals' email address and ".R2D2" extension. For example, a file such as "1.jpg" would appear as something similar to "1.jpg.id-1E857D00.[1024back@tuta.io].R2D2" following encryption. After this process is complete, a text file ("FILES ENCRYPTED.txt") is created on the desktop and a pop-up window is displayed.

To start analyzing R2D2

```
c:\Program Files\Rekall>rekall --filename 0zapftis.vmem

-----
The Rekall Digital Forensic/Incident Response framework 1.7.2.rc1 (Hurricane Ridge).

"We can remember it for you wholesale!"

This program is free software; you can redistribute it and/or modify it under
the terms of the GNU General Public License.

See http://www.rekall-forensic.com/docs/Manual/tutorial.html to get started.
-----
[1] 0zapftis.vmem 10:52:33>
```

Some extra information about R2D2 using imageinfo function

```
[1] 0zapftis.vmem 10:53:09> imageinfo
2021-12-08 10:53:10,526:WARNING:rekall.1:Inventory for repository "http://profiles.rekall-forensic.com" seems malformed
Are you behind a captive portal or proxy? If this is a custom repository, did you forget to create an inventory? You m
st use the tools/profiles/build_profile_repo.py tool with the --inventory flag.
2021-12-08 10:53:10,528:WARNING:rekall.1:Repository http://profiles.rekall-forensic.com will be disabled.
-----> imageinfo()
-----
key          value
-----
Kernel DTB   0x319000
NT Build     2600.xpsp_sp2_rtm.040803-21580000 \
NT Build Ex  -
Signed Drivers
Time (UTC)   2011-10-10 17:06:54Z
Time (Local) 2011-10-10 13:06:54-0400
Sec Since Boot 98,21875
NtSystemRoot C:\WINDOWS
```

Lets enumerate the processes by calling **pslist**

```
[1] 0zapftis.vmem 10:54:30> pslist
-----> pslist()
EPROCESS      name                pid  ppid  thread_count handle_count session_id wow64  process_create_time  proc
ess_exit_time
-----
0x819cc830 System                4    0      55        162      - False -
0x816d63d0 VMwareTray.exe       184  1956     1         28      0 False 2011-10-10 17:04:41Z -
0x8180b478 VMwareUser.exe       192  1956     6         83      0 False 2011-10-10 17:04:41Z -
0x818233c8 reader_sl.exe        228  1956     2         26      0 False 2011-10-10 17:04:41Z -
0x815e7be0 wuauclt.exe           400  964     8        173      0 False 2011-10-10 17:04:46Z -
0x81945020 smss.exe             536   4      3         21      - False 2011-10-10 17:03:56Z -
0x817a34b0 cmd.exe              544  1956     1         30      0 False 2011-10-10 17:06:42Z -
0x816c6020 csrss.exe            608  536    11        355      0 False 2011-10-10 17:03:58Z -
0x813a9020 winlogon.exe         632  536    24        533      0 False 2011-10-10 17:03:58Z -
0x816da020 services.exe         676  632    16        261      0 False 2011-10-10 17:03:58Z -
0x813c4020 lsass.exe            688  632    23        336      0 False 2011-10-10 17:03:58Z -
0x81772ca8 vmacthlp.exe         832  676     1         24      0 False 2011-10-10 17:03:59Z -
0x8167e9d0 svchost.exe          848  676    20        194      0 False 2011-10-10 17:03:59Z -
0x817757f0 svchost.exe          916  676     9        217      0 False 2011-10-10 17:03:59Z -
0x816c6da0 svchost.exe          964  676    63       1058      0 False 2011-10-10 17:03:59Z -
0x815daca8 svchost.exe         1020  676     5         58      0 False 2011-10-10 17:03:59Z -
0x813aeda0 svchost.exe         1148  676    12        187      0 False 2011-10-10 17:04:00Z -
0x817937e0 spoolsv.exe          1260  676    13        140      0 False 2011-10-10 17:04:00Z -
0x81754990 VMwareService.e     1444  676     3        145      0 False 2011-10-10 17:04:00Z -
0x8136c5a0 alg.exe              1616  676     7         99      0 False 2011-10-10 17:04:01Z -
0x815c4da0 wscntfy.exe          1920  964     1         27      0 False 2011-10-10 17:04:39Z -
0x813bcd0 explorer.exe          1956 1884    18        322      0 False 2011-10-10 17:04:39Z -
Out<10:54:31> Plugin: pslist (WinPsList)
[1] 0zapftis.vmem 10:54:31>
```

Right off of the pslist, there are two processes that stand out. **Reader_sl.exe** and **cmd.exe**

Lets see the hierarchy of these processes using **pstree**

```
-----> pstree()
EPROCESS      ppid  thd_count hnd_count  create_time
-----
0x819cc830 System (4)                0      55      162 -
. 0x81945020 smss.exe (536)          4      3      21 2011-10-10 17:03:56Z
.. 0x816c6020 csrss.exe (608)       536    11     355 2011-10-10 17:03:58Z
... 0x813a9020 winlogon.exe (632)   536    24     533 2011-10-10 17:03:58Z
.... 0x816da020 services.exe (676)  632    16     261 2011-10-10 17:03:58Z
..... 0x81772ca8 vmacthlp.exe (832)  676     1      24 2011-10-10 17:03:59Z
..... 0x8167e9d0 svchost.exe (848)   676    20     194 2011-10-10 17:03:59Z
..... 0x817757f0 svchost.exe (916)   676     9     217 2011-10-10 17:03:59Z
..... 0x816c6da0 svchost.exe (964)   676    63    1058 2011-10-10 17:03:59Z
..... 0x815e7be0 wuauclt.exe (400)   964     8     173 2011-10-10 17:04:46Z
..... 0x815c4da0 wscntfy.exe (1920)  964     1      27 2011-10-10 17:04:39Z
.... 0x815daca8 svchost.exe (1020)   676     5      58 2011-10-10 17:03:59Z
.... 0x813aeda0 svchost.exe (1148)   676    12     187 2011-10-10 17:04:00Z
.... 0x817937e0 spoolsv.exe (1260)   676    13     140 2011-10-10 17:04:00Z
.... 0x81754990 VMwareService.e (1444)  676     3     145 2011-10-10 17:04:00Z
.... 0x8136c5a0 alg.exe (1616)       676     7      99 2011-10-10 17:04:01Z
... 0x813c4020 lsass.exe (688)       632    23     336 2011-10-10 17:03:58Z
. 0x813bcd0 explorer.exe (1956)     1884    18     322 2011-10-10 17:04:39Z
. 0x816d63d0 VMwareTray.exe (184)    1956     1      28 2011-10-10 17:04:41Z
. 0x8180b478 VMwareUser.exe (192)    1956     6      83 2011-10-10 17:04:41Z
. 0x818233c8 reader_sl.exe (228)     1956     2      26 2011-10-10 17:04:41Z
. 0x817a34b0 cmd.exe (544)           1956     1      30 2011-10-10 17:06:42Z
```

From pstree, we can see that explore.exe is starting reader_sl.exe and cmd.exe. Everything seems benign for now.

Lets run **cmdscan** to search the memory process to gain visibility on possible attackers.

```
[1] 0zapftis.vmem 11:00:43> cmdscan
-----> cmdscan()
*****
CommandProcess: csrss.exe Pid: 608
CommandHistory: 0x11132d8 Application: cmd.exe Flags: Allocated, Reset
CommandCount: 2 LastAdded: 1 LastDisplayed: 1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x4c4
Cmd Address Text
-----
0 0x004e1eb8 sc query malwar
1 0x011135e8 sc query malware
```

It shows that csrss.exe is attempting to start a service called malware.

Lets run **connscan** to see if there are any suspicious connections to outside IP addresses.

```
1] 0zapftis.vmem 11:03:09> connscan
-----> connscan()
tcpip/GUID/9546A8399BAC4717BC41758594EF0D9C2 matched offset 0x4562+0xf11c0000=0 offset_p local_net_address
remote_net_address pid
-----
0x1a25a50 0.0.0.0:1026 172.16.98.1:6666 1956
```

From the function, it is evident that process 1956 is trying to make a connection to 172.16.98.1.

Lets run **dlllist** on cmd.exe by typing **dlllist proc_regex="cmd.exe"**

```
0x816118d8 4 445 17 UDP 0.0.0.0 2011-10-10 17:03:55Z
Out<11:11:06> Plugin: sockets (Sockets)
1] 0zapftis.vmem 11:13:03> dlllist proc_regex="cmd.exe"
-----> dlllist(proc_regex="cmd.exe")
base size reason dll_path
-----
cmd.exe pid: 544
Command line : "C:\WINDOWS\system32\cmd.exe"
Service Pack 2
-----
0x4ad00000 0x61000 65535 C:\WINDOWS\system32\cmd.exe
0x7c900000 0xb0000 65535 C:\WINDOWS\system32\ntdll.dll
0x7c800000 0xf4000 65535 C:\WINDOWS\system32\kernel32.dll
0x77c10000 0x58000 65535 C:\WINDOWS\system32\msvcrt.dll
0x77d40000 0x90000 65535 C:\WINDOWS\system32\USER32.dll
0x77f10000 0x46000 65535 C:\WINDOWS\system32\GDI32.dll
0x5cb70000 0x26000 1 C:\WINDOWS\system32\ShimEng.dll
0x6f880000 0x1ca000 1 C:\WINDOWS\AppPatch\AcGenral.DLL
0x77dd0000 0x9b000 23 C:\WINDOWS\system32\ADVAPI32.dll
0x77e70000 0x91000 11 C:\WINDOWS\system32\RPCRT4.dll
0x76b40000 0x2d000 2 C:\WINDOWS\system32\WINMM.dll
0x774e0000 0x13c000 2 C:\WINDOWS\system32\ole32.dll
0x77120000 0x8c000 1 C:\WINDOWS\system32\OLEAUT32.dll
0x77be0000 0x15000 1 C:\WINDOWS\system32\MSACM32.dll
0x77c00000 0x8000 3 C:\WINDOWS\system32\VERSION.dll
0x7c9c0000 0x814000 1 C:\WINDOWS\system32\SHELL32.dll
0x77f60000 0x76000 3 C:\WINDOWS\system32\SHLWAPI.dll
0x769c0000 0xb3000 1 C:\WINDOWS\system32\USERENV.dll
0x5ad70000 0x38000 1 C:\WINDOWS\system32\UxTheme.dll
0x10000000 0x59000 1 C:\WINDOWS\system32\mf42u1.dll
0x71ab0000 0x17000 2 C:\WINDOWS\system32\WS2_32.dll
0x71aa0000 0x8000 1 C:\WINDOWS\system32\WS2HELP.dll
0x71f60000 0x8000 1 C:\WINDOWS\system32\snmpapi.dll
0x773d0000 0x102000 1 C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.2600.2180_x-ww_a84f1ff9\comctl32.dll
0x5d090000 0x97000 1 C:\WINDOWS\system32\comctl32.dll
0x77b40000 0x22000 1 C:\WINDOWS\system32\Apphelp.dll
Out<11:13:03> Plugin: dlllist (WinDllList)
[1] 0zapftis.vmem 11:13:03>
```

There is a dll that seemed eye-catching which was the mfc42ul.dll. After googling the dll, it is indeed a malicious dll.



This entry is classified as malware, spyware, adware, or other potentially unwanted software.

If the description states that it is malware, you should immediately run a trusted anti-virus and anti-spyware tool.

Item Details

Type:	AppInit_DLLs
Filename:	%SYSDIR%\mfc42ul.dll
Description:	Backdoor:Win32/R2d2.A

This confirms that we hunted down R2D2 malware.

Analyzing Malwares 3

Cridex

About: Cridex malware, also known as Cridex or W32.Cridex, is a malicious computer worm that spreads to computers by copying itself to removable disks. On each computer it infects, it opens a backdoor and downloads malicious software to the hard disk. The malicious software gathers personal information on the compromised machine, including web session and banking data, and transmits it to a third-party. Cridex-infected machines can also become botnet slaves, participating in behavior such as DDoS attacks.

To start analyzing Cridex

```
c:\Program Files\Rekall>rekall --filename cridex.vmem

-----
The Rekall Digital Forensic/Incident Response framework 1.7.2.rc1 (Hurricane Ridge).

"We can remember it for you wholesale!"

This program is free software; you can redistribute it and/or modify it under
the terms of the GNU General Public License.

See http://www.rekall-forensic.com/docs/Manual/tutorial.html to get started.
-----
[1] cridex.vmem 11:34:33>
```

Some additional information about Cridex can be found using **imageinfo**

```
[1] cridex.vmem 11:35:30> imageinfo
2021-12-08 11:35:32,170:WARNING:rekall.1:Inventory for repository "http://profiles.rekall-forensic.com" seems malformed.
Are you behind a captive portal or proxy? If this is a custom repository, did you forget to create an inventory? You must
use the tools/profiles/build_profile_repo.py tool with the --inventory flag.
2021-12-08 11:35:32,171:WARNING:rekall.1:Repository http://profiles.rekall-forensic.com will be disabled.
-----> imageinfo()
-----
key          value
-----
Kernel DTB   0x2fe000
NT Build     2600.xpsp.080413-2111s 0xc0000000 \
NT Build Ex  -
Signed Drivers
Time (UTC)    2012-07-22 02:45:08Z
Time (Local)  2012-07-21 22:45:08-0400
Sec Since Boot 164.890625
NtSystemRoot  C:\WINDOWS
```


Lets enumerate the processes by calling **pslist** to see all the processes in an infected system

_EPROCESS	name	pid	ppid	thread_count	handle_count	session_id	wow64	process_create_time	process_exit_time
0x823c89c8	System	4	0	53	240	-	False	-	-
0x822f1020	smss.exe	368	4	3	19	-	False	2012-07-22 02:42:31Z	-
0x822a0598	csrss.exe	584	368	9	326	0	False	2012-07-22 02:42:32Z	-
0x82298700	winlogon.exe	608	368	23	519	0	False	2012-07-22 02:42:32Z	-
0x81e2ab28	services.exe	652	608	16	243	0	False	2012-07-22 02:42:32Z	-
0x81e2a3b8	lsass.exe	664	608	24	330	0	False	2012-07-22 02:42:32Z	-
0x820e8da0	alg.exe	788	652	7	104	0	False	2012-07-22 02:43:01Z	-
0x82311360	svchost.exe	824	652	20	194	0	False	2012-07-22 02:42:33Z	-
0x81e29ab8	svchost.exe	908	652	9	226	0	False	2012-07-22 02:42:33Z	-
0x823001d0	svchost.exe	1004	652	64	1118	0	False	2012-07-22 02:42:33Z	-
0x821dfda0	svchost.exe	1056	652	5	60	0	False	2012-07-22 02:42:33Z	-
0x821fcda0	wuauc.lt.exe	1136	1004	8	173	0	False	2012-07-22 02:43:46Z	-
0x82295650	svchost.exe	1220	652	15	197	0	False	2012-07-22 02:42:35Z	-
0x821dea70	explorer.exe	1484	1464	17	415	0	False	2012-07-22 02:42:36Z	-
0x81eb17b8	spoolsv.exe	1512	652	14	113	0	False	2012-07-22 02:42:36Z	-
0x8205bda0	wuauc.lt.exe	1588	1004	5	132	0	False	2012-07-22 02:44:01Z	-
0x81e7bda0	reader_sl.exe	1640	1484	5	39	0	False	2012-07-22 02:42:36Z	-

Lets see the hierarchy of these processes using **pstree**

_EPROCESS	ppid	thd_count	hnd_count	create_time
0x823c89c8 System (4)	0	53	240	-
. 0x822f1020 smss.exe (368)	4	3	19	2012-07-22 02:42:31Z
.. 0x822a0598 csrss.exe (584)	368	9	326	2012-07-22 02:42:32Z
.. 0x82298700 winlogon.exe (608)	368	23	519	2012-07-22 02:42:32Z
... 0x81e2ab28 services.exe (652)	608	16	243	2012-07-22 02:42:32Z
.... 0x820e8da0 alg.exe (788)	652	7	104	2012-07-22 02:43:01Z
.... 0x82311360 svchost.exe (824)	652	20	194	2012-07-22 02:42:33Z
.... 0x81e29ab8 svchost.exe (908)	652	9	226	2012-07-22 02:42:33Z
.... 0x823001d0 svchost.exe (1004)	652	64	1118	2012-07-22 02:42:33Z
..... 0x821fcda0 wuauc.lt.exe (1136)	1004	8	173	2012-07-22 02:43:46Z
..... 0x8205bda0 wuauc.lt.exe (1588)	1004	5	132	2012-07-22 02:44:01Z
.... 0x821dfda0 svchost.exe (1056)	652	5	60	2012-07-22 02:42:33Z
.... 0x82295650 svchost.exe (1220)	652	15	197	2012-07-22 02:42:35Z
.... 0x81eb17b8 spoolsv.exe (1512)	652	14	113	2012-07-22 02:42:36Z
... 0x81e2a3b8 lsass.exe (664)	608	24	330	2012-07-22 02:42:32Z
. 0x821dea70 explorer.exe (1484)	1464	17	415	2012-07-22 02:42:36Z
. 0x81e7bda0 reader_sl.exe (1640)	1484	5	39	2012-07-22 02:42:36Z

Reader_sl.exe stands out because it has a parent id of 1484 which is a **explorer.exe**

Lets use **psxview** to see any hidden processes

_EPROCESS	name	pid	PsActiveProcessHead	CSRSS	PspCidTable	Sessions	Handles	PSScan	Thrdproc
0x823c89c8	System	4	True	False	True	False	False	True	True
0x822f1020	smss.exe	368	True	False	True	False	True	True	True
0x822a0598	csrss.exe	584	True	False	True	True	True	True	True
0x82298700	winlogon.exe	608	True	True	True	True	True	True	True
0x81e2ab28	services.exe	652	True	True	True	True	True	True	True
0x81e2a3b8	lsass.exe	664	True	True	True	True	True	True	True
0x820e8da0	alg.exe	788	True	True	True	True	True	True	True
0x82311360	svchost.exe	824	True	True	True	True	True	True	True
0x81e29ab8	svchost.exe	908	True	True	True	True	True	True	True
0x823001d0	svchost.exe	1004	True	True	True	True	True	True	True
0x821dfda0	svchost.exe	1056	True	True	True	True	True	True	True
0x821fcda0	wuauclt.exe	1136	True	True	True	True	True	True	True
0x82295650	svchost.exe	1220	True	True	True	True	True	True	True
0x821dea70	explorer.exe	1484	True	True	True	True	True	True	True
0x81eb17b8	spoolsv.exe	1512	True	True	True	True	True	True	True
0x8205bda0	wuauclt.exe	1588	True	True	True	True	True	True	True
0x81e7bda0	reader_sl.exe	1640	True	True	True	True	True	True	True

So far, the output seems benign.

Next, we want to check for open sockets and tcp connections. To do those, we will use sockets and connscan.

Connscan

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

There are two tcp connections from 172.16.112.128 to 41.168.5.140 and 125.19.103.198 by 1484(explorer.exe)

Sockets

Offset(V)	PID	Port	Proto	Protocol	Address	Create Time
0x81ddb780	664	500	17	UDP	0.0.0.0	
0x82240d08	1484	1038	6	TCP	0.0.0.0	
0x81dd7618	1220	1900	17	UDP	172.16.112.128	
0x82125610	788	1028	6	TCP	127.0.0.1	
0x8219cc08	4	445	6	TCP	0.0.0.0	
0x81ec23b0	908	135	6	TCP	0.0.0.0	
0x82276878	4	139	6	TCP	172.16.112.128	
0x82277460	4	137	17	UDP	172.16.112.128	
0x81e76620	1004	123	17	UDP	127.0.0.1	
0x82172808	664	0	255	Reserved	0.0.0.0	
0x81e3f460	4	138	17	UDP	172.16.112.128	
0x821f0630	1004	123	17	UDP	172.16.112.128	
0x822cd2b0	1220	1900	17	UDP	127.0.0.1	
0x82172c50	664	4500	17	UDP	0.0.0.0	
0x821f0d00	4	445	17	UDP	0.0.0.0	

We can see that one of the TCP connections from 1484 is still open.

We can see two open TCP connections to 2 different external IP addresses which bring up suspicion.

Lets create a dmp file using

```
[1] cridex.vmem 12:20:42> memdump 1640, dump_dir="."
```

After we open the dmp file in a notepad and search for 41.168.5.140 which is from the output of the conscan, we see

Accept: */*

User-Agent: Mozilla/5.0 (Windows; U; MSIE 7.0; Windows NT 6.0; en-US)

PVñ¿)Gz| E û]@ €-Û-p€)“Œºe]ppÀPDpI8 POST /zb/v_01_a/in/ HTTP/1.1

Host: 41.168.5.140:8080

Content-Length: 229

Connection: Keep-Alive

Cache-Control: no-cache

Showing that reader_sl.exe is sending data to 41.168.5.140 via http post.

This concludes the findings of cridex malware.

References

- <https://sansorg.egnyte.com/dl/L4hx0pM9y3>
- <https://holdmybeersecurity.com/2017/07/29/rekall-memory-analysis-framework-for-windows-linux-and-mac-osx/>
- <https://readthedocs.org/projects/rekall/downloads/pdf/latest/>
- <http://blog.rekall-forensic.com/2015/09/installing-rekall-on-windows.html>
- <https://samsclass.info/121/proj/rekall-fail.htm>
- <https://class.malware.re/2020/01/28/analysis-exercise.html>

- <https://capoo.tistory.com/9>
- <https://github.com/volatilityfoundation/volatility/wiki/Memory-Samples>