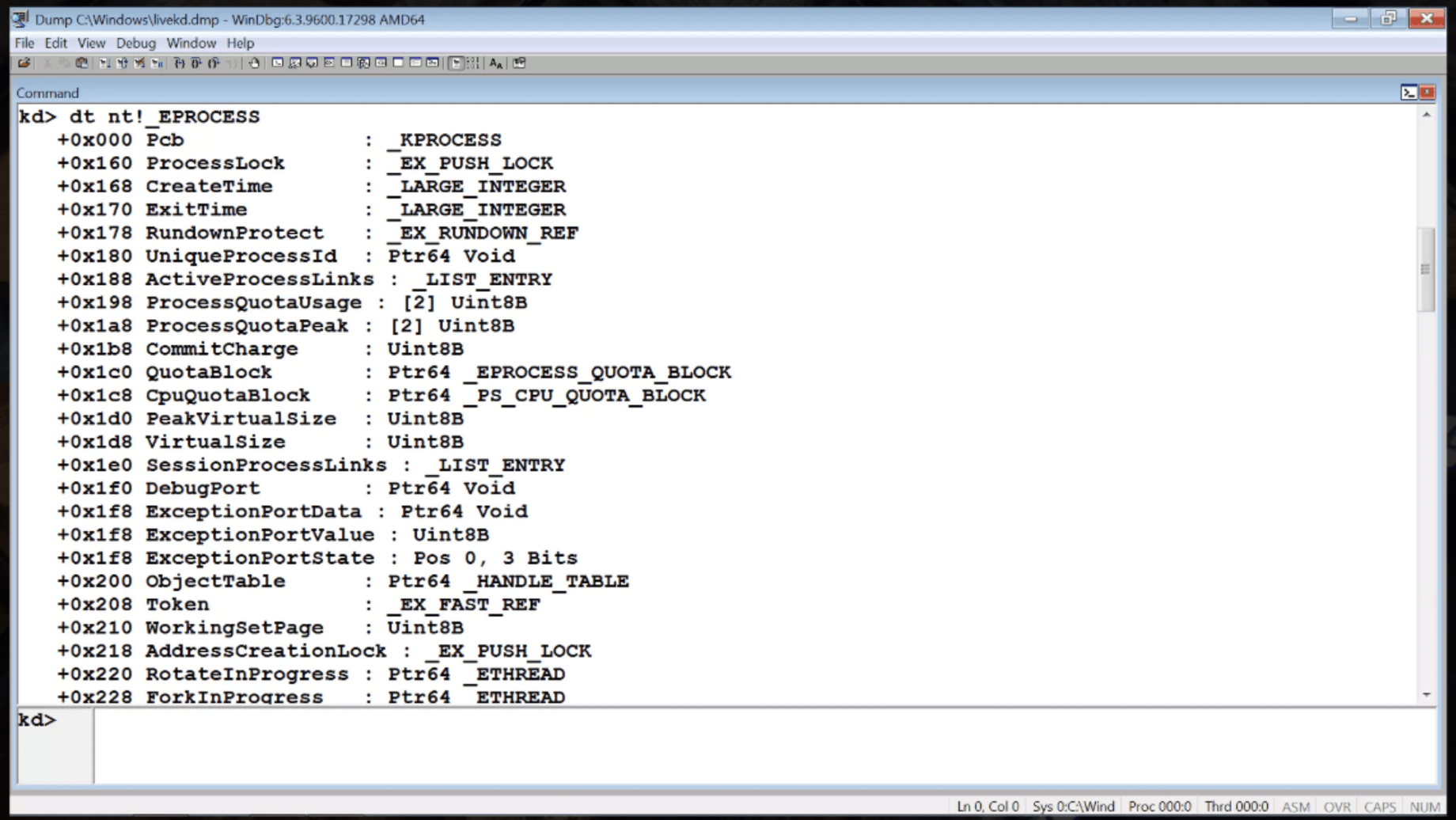
Investigating Process

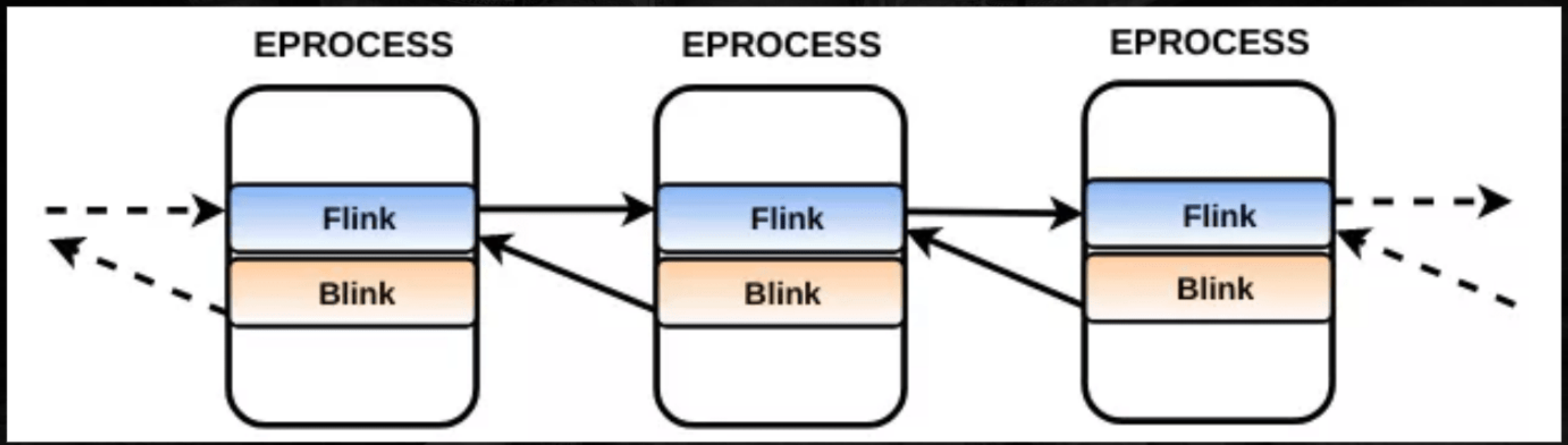
Process Overview

- Management and containment object that allows the thread to execute
- Private virtual address space (2GB/3GB on 32 bit, 8TB on 64 bit)
- Private handle table to kernel objects
- Contains one or more threads
- An executive object of type ***_EPROCESS***

Below screen shot shows the _EPROCESS structure



Operating System keeps track of active processes as double linked lists



Process Enumeration (*pslist*)

pslist plugin lists the process by walking the double link list

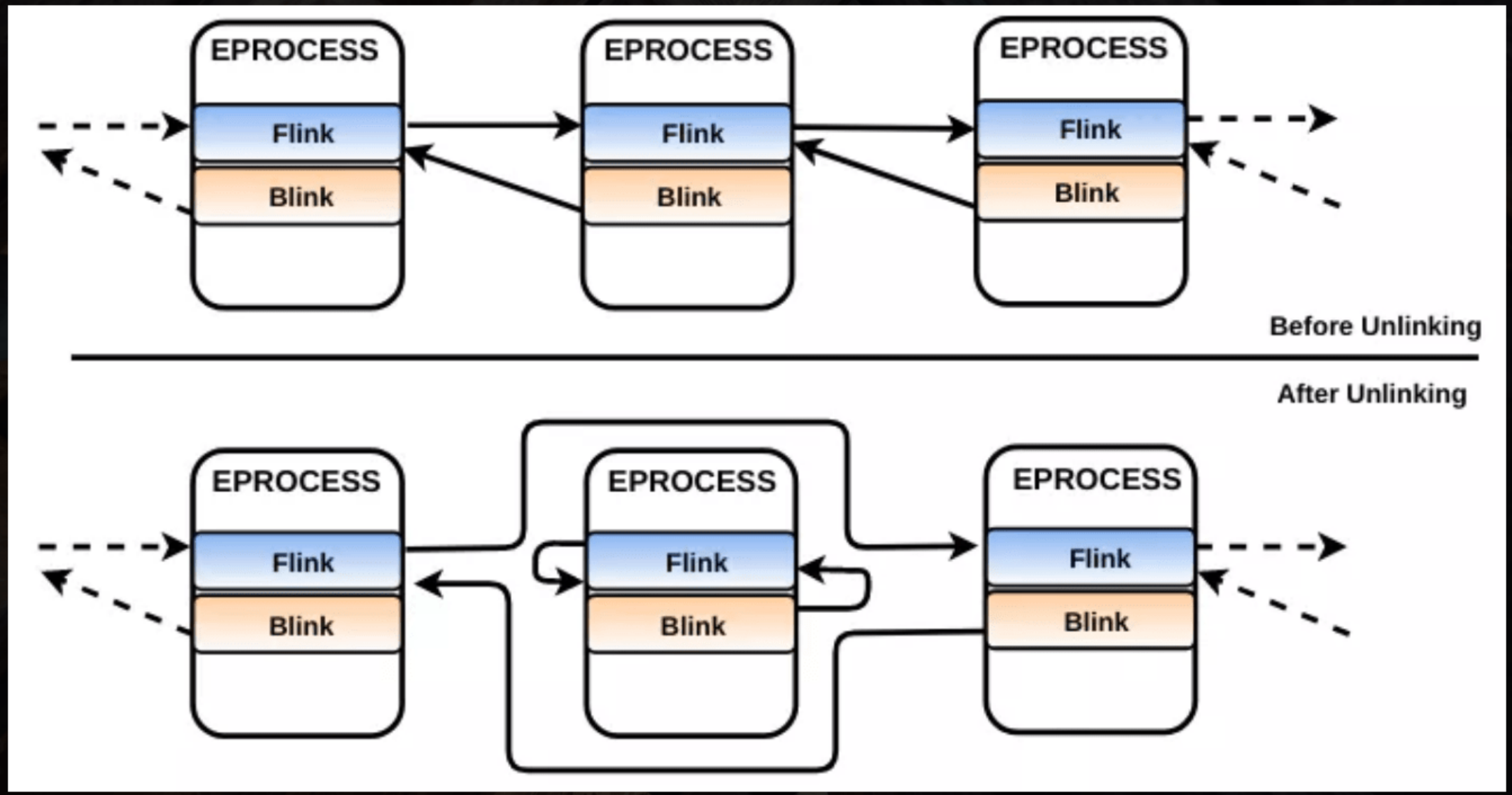
```
root@kratos:~/Volatility# python vol.py -f rat9002.vmem pslist
Volatility Foundation Volatility Framework 2.5
Offset(V)  Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start
           Exit
---------- -------------------- ------ ------ ------ -------- ------ ------ ------------------
---------- ------------------------------
0x819cc830 System                    4      0     56      255 ------      0
0x81858d08 smss.exe                588      4      3       19 ------      0 2015-01-14 09:35:48
 UTC+0000
0x818fa128 csrss.exe               636    588     11      401      0      0 2015-01-14 09:35:49
 UTC+0000
0x8153a6e8 winlogon.exe            660    588     23      520      0      0 2015-01-14 09:35:49
 UTC+0000
0x818fe020 services.exe            704    660     17      355      0      0 2015-01-14 09:35:50
 UTC+0000
0x81902128 lsass.exe               716    660     21      331      0      0 2015-01-14 09:35:50
 UTC+0000
0x818f7020 vmacthlp.exe            872    704      1       25      0      0 2015-01-14 09:35:50
 UTC+0000
0x8152b020 svchost.exe             888    704     22      196      0      0 2015-01-14 09:35:50
 UTC+0000
0x8176cda0 svchost.exe             968    704     10      221      0      0 2015-01-14 09:35:50
 UTC+0000
0x8183a398 svchost.exe            1052    704     69     1229      0      0 2015-01-14 09:35:50
```

Process Enumeration (*pstree*)

pstree plugin takes the output from ***pslist*** and displays the relationship between parent and child process

```
root@kratos:~/Volatility# python vol.py -f rat9002.vmem pstree
Volatility Foundation Volatility Framework 2.5
Name                                                  Pid   PPid   Thds   Hnds Time
-------------------------------------------------- ------ ------ ------ ------ ----
 0x819cc830:System                                      4      0     56    255 1970-01-01 00:00:00 UTC+0000
. 0x81858d08:smss.exe                                 588      4      3     19 2015-01-14 09:35:48 UTC+0000
.. 0x8153a6e8:winlogon.exe                            660    588     23    520 2015-01-14 09:35:49 UTC+0000
... 0x818fe020:services.exe                           704    660     17    355 2015-01-14 09:35:50 UTC+0000
.... 0x818a4a88:alg.exe                               576    704      6    101 2015-01-14 09:36:02 UTC+0000
.... 0x818a3990:spoolsv.exe                          1388    704     14    131 2015-01-14 09:35:52 UTC+0000
.... 0x816cb020:svchost.exe                          1100    704      6     66 2015-01-14 09:35:50 UTC+0000
.... 0x8183a398:svchost.exe                          1052    704     69   1229 2015-01-14 09:35:50 UTC+0000
..... 0x818f5020:wuauclt.exe                         1088   1052      8    177 2015-01-14 09:36:46 UTC+0000
.... 0x8176cda0:svchost.exe                           968    704     10    221 2015-01-14 09:35:50 UTC+0000
.... 0x8151a7f8:vmtoolsd.exe                         1988    704      6    228 2015-01-14 09:35:58 UTC+0000
.... 0x81523da0:svchost.exe                          1144    704     16    211 2015-01-14 09:35:52 UTC+0000
.... 0x818f7020:vmacthlp.exe                          872    704      1     25 2015-01-14 09:35:50 UTC+0000
.... 0x81684da0:VMUpgradeHelper                       232    704      5     94 2015-01-14 09:36:01 UTC+0000
.... 0x8152b020:svchost.exe                           888    704     22    196 2015-01-14 09:35:50 UTC+0000
... 0x81902128:lsass.exe                              716    660     21    331 2015-01-14 09:35:50 UTC+0000
.. 0x818fa128:csrss.exe                               636    588     11    401 2015-01-14 09:35:49 UTC+0000
 0x818b5108:explorer.exe                             1608   1544     22    609 2015-01-14 09:36:13 UTC+0000
. 0x81684800:GrooveMonitor.e                         1932   1608      2    106 2015-01-14 09:36:14 UTC+0000
. 0x816166f0:VMwareTray.exe                          1824   1608      1     58 2015-01-14 09:36:14 UTC+0000
. 0x8177bda0:cmd.exe                                  444   1608      1     32 2015-03-24 18:31:55 UTC+0000
.. 0x8152bda0:rundll32.exe                           1456    444      8    112 2015-03-24 18:32:37 UTC+0000
. 0x8139ec88:ZoomIt.exe                              2004   1608      2     37 2015-01-14 09:36:14 UTC+0000
. 0x813cfda0:VMwareUser.exe                          1632   1608     10    221 2015-01-14 09:36:14 UTC+0000
. 0x814e8a70:ctfmon.exe                              2028   1608      1     71 2015-01-14 09:36:14 UTC+0000
root@kratos:~/Volatility#
```

DKOM attacks



Process Enumeration (*psscan*)

psscan plugin enumerates the processes using ***pool tag*** scanning

```
root@kratos:~/Volatility# python vol.py -f rat9002.vmem psscan
Volatility Foundation Volatility Framework 2.5
Offset(P)          Name                PID   PPID PDB        Time created                   Time exited
------------------ --------------- ------ ------ ---------- ------------------------------ ------------------
0x000000000159ec88 ZoomIt.exe        2004   1608 0x089002e0 2015-01-14 09:36:14 UTC+0000

0x00000000015cfda0 VMwareUser.exe    1632   1608 0x08900180 2015-01-14 09:36:14 UTC+0000

0x00000000016e8a70 ctfmon.exe        2028   1608 0x08900300 2015-01-14 09:36:14 UTC+0000

0x000000000171a7f8 vmtoolsd.exe      1988    704 0x089001c0 2015-01-14 09:35:58 UTC+0000

0x0000000001723da0 svchost.exe       1144    704 0x08900160 2015-01-14 09:35:52 UTC+0000

0x000000000172b020 svchost.exe        888    704 0x089000e0 2015-01-14 09:35:50 UTC+0000

0x000000000172bda0 rundll32.exe      1456    444 0x08900200 2015-03-24 18:32:37 UTC+0000

0x000000000173a6e8 winlogon.exe       660    588 0x08900060 2015-01-14 09:35:49 UTC+0000

0x00000000018166f0 VMwareTray.exe    1824   1608 0x08900280 2015-01-14 09:36:14 UTC+0000

0x0000000001884800 GrooveMonitor.e   1932   1608 0x089002c0 2015-01-14 09:36:14 UTC+0000

0x0000000001884da0 VMUpgradeHelper    232    704 0x089001e0 2015-01-14 09:36:01 UTC+0000
```

Process Enumeration (*psxview*)

Enumerates process using 7 different methods

```
root@kratos:~/Volatility# python vol.py -f rat9002.vmem psxview
Volatility Foundation Volatility Framework 2.5
Offset(P)  Name                    PID pslist psscan thrdproc pspcid csrss session deskthrd ExitTime
---------- -------------------- ------ ------ ------ -------- ------ ----- ------- -------- --------
0x01884800 GrooveMonitor.e        1932 True   True   True     True   True  True    True
0x01aa4a88 alg.exe                 576 True   True   True     True   True  True    True
0x01723da0 svchost.exe            1144 True   True   True     True   True  True    True
0x018cb020 svchost.exe            1100 True   True   True     True   True  True    True
0x01a3a398 svchost.exe            1052 True   True   True     True   True  True    True
0x01afe020 services.exe            704 True   True   True     True   True  True    True
0x018166f0 VMwareTray.exe         1824 True   True   True     True   True  True    True
0x01b02128 lsass.exe               716 True   True   True     True   True  True    True
0x0172bda0 rundll32.exe           1456 True   True   True     True   True  True    True
0x0159ec88 ZoomIt.exe             2004 True   True   True     True   True  True    True
0x01884da0 VMUpgradeHelper         232 True   True   True     True   True  True    True
0x01af5020 wuauclt.exe            1088 True   True   True     True   True  True    True
0x015cfda0 VMwareUser.exe         1632 True   True   True     True   True  True    True
0x01af7020 vmacthlp.exe            872 True   True   True     True   True  True    True
0x0196cda0 svchost.exe             968 True   True   True     True   True  True    True
0x0172b020 svchost.exe             888 True   True   True     True   True  True    True
0x01aa3990 spoolsv.exe            1388 True   True   True     True   True  True    True
0x0173a6e8 winlogon.exe            660 True   True   True     True   True  True    True
0x0197bda0 cmd.exe                 444 True   True   True     True   True  True    True
0x016e8a70 ctfmon.exe             2028 True   True   True     True   True  True    True
0x0171a7f8 vmtoolsd.exe           1988 True   True   True     True   True  True    True
0x01ab5108 explorer.exe           1608 True   True   True     True   True  True    True
0x01bcc830 System                    4 True   True   True     True   False False   False
0x01a58d08 smss.exe                588 True   True   True     True   False False   False
```

Lab 12: The Case of Perseus

A Top executive in your organization suspects that his system is infected after opening an attachment that came in via email. You are the incident responder handling this incident, you have collected the memory image (***perseus.vmem***). Investigate the memory image and answer the below questions:

- Do you see any process that looks suspicious?
- What is the name of the suspicious process?
- What is the process id of the suspicious process?
- Are there more than one suspicious process?
- How are these suspicious processes related?
- How is the attacker trying to blend in with legitimate processes?

Investigating Process Handles and Registry

Objects and Handles

- Object Manager is responsible for creating and manipulating objects
- Objects are runtime instances of static structures

  Examples: process, mutex, event, desktop, file
- Objects Reside in the Kernel Memory
- User Mode Process can obtain a handle to an Object.
- Kernel code can either obtain handle or obtain a direct pointer to an object
- Objects are reference counted

Enumerating Handles

handles plugin displays all the handles from the handle table, with ***-p*** option you can filter on specific process and with ***-t*** option you can filter based on specific handle types

```
root@kratos:~/Volatility# python vol.py -f rat9002.vmem handles -p 1456 -t File,Mutant,Key --silent
Volatility Foundation Volatility Framework 2.5
Offset(V)     Pid     Handle     Access Type            Details
---------- ------ ---------- ---------- --------------- -------
0x81845f90   1456       0x7c   0x100020 File            \Device\HarddiskVolume1\WINDOWS\system32
0x818ff9d8   1456      0x650   0x1f01ff File            \Device\Tcp
0x8152e420   1456      0x664   0x1f01ff File            \Device\Tcp
0x818fa028   1456      0x668   0x1f01ff File            \Device\Afd\Endpoint
0x81778c68   1456      0x6a8   0x1f01ff File            \Device\RawIp\0
0x81777c38   1456      0x6ac   0x1f01ff File            \Device\Afd\Endpoint
0x8174c0e0   1456      0x6e0   0x1f0001 Mutant          MSCTF.Shared.MUTEX.MHG
0xe18bc020   1456      0x6f4    0xf003f Key             MACHINE\SYSTEM\CONTROLSET001\SERVICES\WINSOCK2\P
ARAMETERS\NAMESPACE_CATALOG5
0xe1585190   1456      0x6fc    0xf003f Key             MACHINE\SYSTEM\CONTROLSET001\SERVICES\WINSOCK2\P
ARAMETERS\PROTOCOL_CATALOG9
0x815a1e60   1456      0x714   0x1f0001 Mutant          CTF.TimListCache.FMPDefaultS-1-5-21-1078081533-1
292428093-1417001333-500MUTEX.DefaultS-1-5-21-1078081533-1292428093-1417001333-500
0x813e0b90   1456      0x718   0x120001 Mutant          ShimCacheMutex
0x815fc310   1456      0x71c   0x1f0001 Mutant          CTF.TMD.MutexDefaultS-1-5-21-1078081533-12924280
93-1417001333-500
0x815fc360   1456      0x720   0x1f0001 Mutant          CTF.Layouts.MutexDefaultS-1-5-21-1078081533-1292
428093-1417001333-500
0x815fc3b0   1456      0x724   0x1f0001 Mutant          CTF.Asm.MutexDefaultS-1-5-21-1078081533-12924280
93-1417001333-500
0x8153a440   1456      0x728   0x1f0001 Mutant          CTF.Compart.MutexDefaultS-1-5-21-1078081533-1292
428093-1417001333-500
0x8153a490   1456      0x72c   0x1f0001 Mutant          CTF.LBES.MutexDefaultS-1-5-21-1078081533-1292428
093-1417001333-500
```

Mutex

- Mutex ("mutual exclusion") also called as Mutant

- Used as a locking mechanism to synchronize access to a resource on the system. "For example, to prevent two threads from writing to shared memory at the same time, each thread waits for ownership of a mutex object before executing the code that accesses the memory. After writing to the shared memory, the thread releases the mutex object."

- A mutex is usually used by malware to mark its presence and avoid the infection by different instances of the same malware.

- A mutex can be used as a unique indicator to identify the malware

Detecting Malware Presence using Mutex

In the below screenshot ***Extreme RAT*** creates a mutex to mark its presence

```
root@kratos:~/Volatility# python vol.py -f xrat.vmem handles -p 1560 -t Mutant --silent
Volatility Foundation Volatility Framework 2.5
Offset(V)     Pid     Handle     Access Type            Details
---------- ------ ---------- ---------- --------------- -------
0x816f46e8   1560       0x64   0x1f0001 Mutant          DDrawWindowListMutex
0x817c3f28   1560       0x68   0x1f0001 Mutant          DDrawDriverObjectListMutex
0x816f1218   1560       0x6c   0x110000 Mutant          __DDrawExclMode__
0x818bfc68   1560       0x70   0x110000 Mutant          __DDrawCheckExclMode
0x8141bde8   1560       0xe8   0x1f0001 Mutant          oZ694XMhk6yxgbTA0
0x81693748   1560       0xec   0x120001 Mutant          ShimCacheMutex
0x817c3ab0   1560      0x160   0x100000 Mutant          _!MSFTHISTORY!_
0x8129cbb0   1560      0x170   0x1f0001 Mutant          c:!documents and settings!administrato
r!local settings!temporary internet files!content.ie5!
0x8143dc50   1560      0x17c   0x1f0001 Mutant          c:!documents and settings!administrato
r!cookies!
0x816c5c38   1560      0x188   0x1f0001 Mutant          c:!documents and settings!administrato
r!local settings!history!history.ie5!
0x817c25d8   1560      0x190   0x100000 Mutant          WininetStartupMutex
0x817c2518   1560      0x1ac   0x100000 Mutant          WininetProxyRegistryMutex
0x814307f8   1560      0x1b0   0x1f0001 Mutant          WininetConnectionMutex
0x817a3248   1560      0x1d4   0x1f0001 Mutant          ZoneAttributeCacheCounterMutex
0x817a3248   1560      0x1d8   0x1f0001 Mutant          ZoneAttributeCacheCounterMutex
0x81513f38   1560      0x218   0x100000 Mutant          RasPbFile
root@kratos:~/Volatility#
```

Registry

- The registry contains various settings and configuration for Windows
- The registry can be used to determine malware persistence
- Volatility's ***printkey*** plugin can be used to print Registry key, subkeys and its values
- Malware normally modifies one of the startup registry keys maintain persistence

Common Startup Registry Keys

- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce
- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run
- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
- HKCU\Software\Microsoft\Windows NT\CurrentVersion\Windows
- HKCU\Software\Microsoft\Windows NT\CurrentVersion\Windows\Run
- HKCU\Software\Microsoft\Windows\CurrentVersion\Run
- HKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce

Detecting Malware Persistence

Malicious executable "system.exe" is added in the Run registry key, this will ensure that malware will run every time system starts

```
root@kratos:~/Volatility# python vol.py -f xrat.vmem printkey -K "Software\Microsoft\Windows\CurrentVersion\Run"
Volatility Foundation Volatility Framework 2.5
Legend: (S) = Stable   (V) = Volatile

--------------------------
Registry: \Device\HarddiskVolume1\Documents and Settings\Administrator\NTUSER.DAT
Key name: Run (S)
Last updated: 2016-04-30 17:41:35 UTC+0000

Subkeys:

Values:
REG_SZ        ZoomIt          : (S) C:\softwares\ZoomIt\ZoomIt.exe
REG_SZ        ctfmon.exe      : (S) C:\WINDOWS\system32\ctfmon.exe
REG_EXPAND_SZ HKCU            : (S) C:\WINDOWS\InstallDir\system.exe
root@kratos:~/Volatility#
```

Detecting Malware Persistence

Malicious executable "***system.exe***" is added in the ***Run*** registry key, this will ensure that malware will run every time system starts

```
root@kratos:~/Volatility# python vol.py -f xrat.vmem printkey -K "Software\Microsoft\Windows\Cu
rrentVersion\Run"
Volatility Foundation Volatility Framework 2.5
Legend: (S) = Stable   (V) = Volatile

---------------------------
```

```
Registry: \Device\HarddiskVolume1\Documents and Settings\Administrator\NTUSER.DAT
Key name: Run (S)
Last updated: 2016-04-30 17:41:35 UTC+0000

Subkeys:

Values:
REG_SZ        ZoomIt          : (S) C:\softwares\ZoomIt\ZoomIt.exe
REG_SZ        ctfmon.exe      : (S) C:\WINDOWS\system32\ctfmon.exe
REG_EXPAND_SZ HKCU            : (S) C:\WINDOWS\InstallDir\system.exe
root@kratos:~/Volatility#
```

Lab 13: The Case of Sypbot

While you are reading an article on malware called 'spybot' you find that spybot creates a mutex (mutant) "***krnel***" to mark its presence. Analyze the memory image (spybot.vmem) and answer the below questions:

- Is the host infected with spybot?
- What is the malicious process id?
- What is the name of the malicious process?
- Which other process is related to the malicious process and what is its pid?