Info (--info)

```
python vol.py --info
```

```
> python vol.py --info | find "Linux"
$ python vol.py --info | grep -i Linux
```

Identify the correct profile (imageinfo)

Windows:

```
python vol.py --file dumps/winxp.mem imageinfo
```

Prints an high level summary of the memory sample.

Linux:

```
$ strings dumps/linux90.lime | grep -i 'MESSAGE=Linux version' | uniq
```

Registry

hivescan

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 hivescan
```

Finds the physical addresses of Registry hives in memory

hivelist

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 hivelist
```

Locates the virtual addresses of registry hives in memory, and the full paths to the corresponding hive on disk

printkey

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 printkey -o 0x8ee66740 -K
“Microsoft\Windows NT\CurrentVersion”
```

Displays the subkeys, values, data, and data types contained within a specified registry key.

hivedump

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 hivedump -o 0x8ee66740
```

Recursively lists all subkeys in a hive.

hashdump

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 hashdump -y 0x8b21c008 -s
0x9aad6148
```

Extracts and decrypts cached domain credentials stored in the registry

Processes (Windows)

pslist [-P]

```
python vol.py --file dumps/winxp.mem --profile=WinXPSP2x86 pslist
python vol.py --file dumps/winxp.mem --profile=WinXPSP2x86 pslist -P
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 pslist
```

Lists the processes of the system.

psscan

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 psscan
```

This can find processes that previously terminated (inactive) and processes that have been hidden or unlinked by a rootkit.

The DTB (Directory Table Base) is what Volatility uses to translate virtual addresses to physical addresses.

pstree

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 pstree
```

Prints a parent/child relationship tree.

psxview

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 psxview
```

Gives a cross-reference of processes based on multiple sources.

dlllist

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 dlllist
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 dlllist --pid=3152
```

Displays process's loaded DLLs.

dlldump

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 dlldump --pid=3152 --
base=0x04a330000 -D dumps/
```

Extracts a DLL from a process's memory space and dump it to disk for analysis.

handles

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 handles --pid=3152 --object-
type=process
```

Displays the open handles in a process. This applies to files, registry keys, mutexes, named pipes, events, window stations, desktops, threads, and all other types of securable executive objects.

getsids

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 getsids
```

Views the SIDs (Security Identifiers) associated with a process.

cmdscan

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 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).

consoles

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 consoles
```

Finds commands that attackers typed into cmd.exe or executed via backdoors.

Processes (Linux)

linux_pslist

```
python vol.py --file=dumps/linux91.lime --profile=LinuxUbuntu91x64 linux_pslist
python vol.py --file=dumps/linux91.dmp --profile=LinuxUbuntu91x64 linux_pslist
```

Prints the list of active processes.

linux_pstree

```
python vol.py --file=dumps/linux90.lime --profile=LinuxUbuntu90x64 linux_pstree
```

Prints a parent/child relationship tree.

linux_psaux

```
python vol.py --file=dumps/linux90.lime --profile=LinuxUbuntu90x64 linux_psaux
```

Enumerates processes and it can show the command-line arguments.

linux_psxview

```
python vol.py --file=dumps/linux90.lime --profile=LinuxUbuntu90x64 linux_psxview
```

Gives a cross-reference of processes based on multiple sources.

Process Memory (Windows)

memmap

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 memmap --pid=3152 > dumps/3152.txt
```

Shows exactly which pages are memory resident, given a specific process DTB.

memdump

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 memdump --pid=3152 -D dumps/
```

Extracts all memory resident pages in a process into an individual file.

procdump

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 procdump --pid=3152 -D dumps/
```

Dumps a process's executable.

Evtlogs (x86 and x64 Windows XP and Windows 2003 Server only)

```
python vol.py --file dumps/imipenem --profile=WinXPSP2x86 evtlogs -D dumps/
```

Extracts and parses binary event logs from memory.

iehistory

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 iehistory
```

Recovers fragments of IE history index.dat cache files.

Process Memory (Linux)

linux_memmap

```
python vol.py --file=dumps/linux91.dmp --profile=LinuxUbuntu91x64 linux_memmap
```

Prints the list of allocated and memory-resident (non-swapped) pages in a process. The virtual and physical addresses are shown.

linux_proc_maps

```
python vol.py --file=dumps/linux91.dmp --profile=LinuxUbuntu91x64 linux_proc_maps -p 4695
```

Prints details of process memory, including heaps, stacks, and shared libraries.

linux_bash

```
python vol.py --file=dumps/linux91.dmp --profile=LinuxUbuntu91x64 linux_bash
```

Recovers bash history from memory.

Networking (Windows)

Connections (x86 and x64 Windows XP and Windows 2003 Server only)

```
python vol.py --file dumps/winxp.mem --profile=WinXPSP2x86 connections
```

Lists TCP connections that were active at the time of the memory acquisition.

connscan (x86 and x64 Windows XP and Windows 2003 Server only)

```
python vol.py --file dumps/winxp.mem --profile=WinXPSP2x86 connscan
```

This can find artifacts from previous connections that have since been terminated, in addition to the active ones.

sockets (x86 and x64 Windows XP and Windows 2003 Server only)

```
python vol.py --file dumps/winxp.mem --profile=WinXPSP2x86 sockets
```

Lists sockets for any protocol: IP Addresses, Ports and Transport protocols (TCP, UDP and raw).

sockscan (x86 and x64 Windows XP and Windows 2003 Server only)

```
python vol.py --file dumps/winxp.mem --profile=WinXPSP2x86 sockscan
```

As with **connscan**, this can pick up residual data and artifacts from previous sockets.

netscan (32- and 64-bit Windows Vista, Windows 2008 Server and Windows 7)

```
python vol.py --file dumps/win7.dmp --profile=Win7SP1x86 netscan
```

To scan for network artifacts in 32- and 64-bit Windows Vista, Windows 2008 Server and Windows 7 memory dumps.

Networking (Linux)

linux_arp

```
python vol.py --file=dumps/linux91.dmp --profile=LinuxUbuntu91x64 linux_arp
```

Prints the ARP table. The most important data in an ARP table is the MAC and IP address pairs of the devices on the network. It also contains other valuable information, such as the specific interface a MAC address is connected to.

linux_ifconfig

```
python vol.py --file=dumps/linux91.dmp --profile=LinuxUbuntu91x64 linux_ifconfig
```

Prints the active interface information, including IPs, interface name, MAC address, and whether the NIC is in promiscuous mode or not (sniffing).

linux_netstat

```
python vol.py --file=dumps/linux91.dmp --profile=LinuxUbuntu91x64 linux_netstat
```

Displays the contents of various network-related data structures and active connections. Lists the Protocol (TCP, UDP, UNIX), State, I-Node, Daemon and Path.