



# Forensics for System Administrators

## Memory Acquisition II

**Klaus Möller**

*WP8-T1*

Webinar, 14th of December 2021

Public

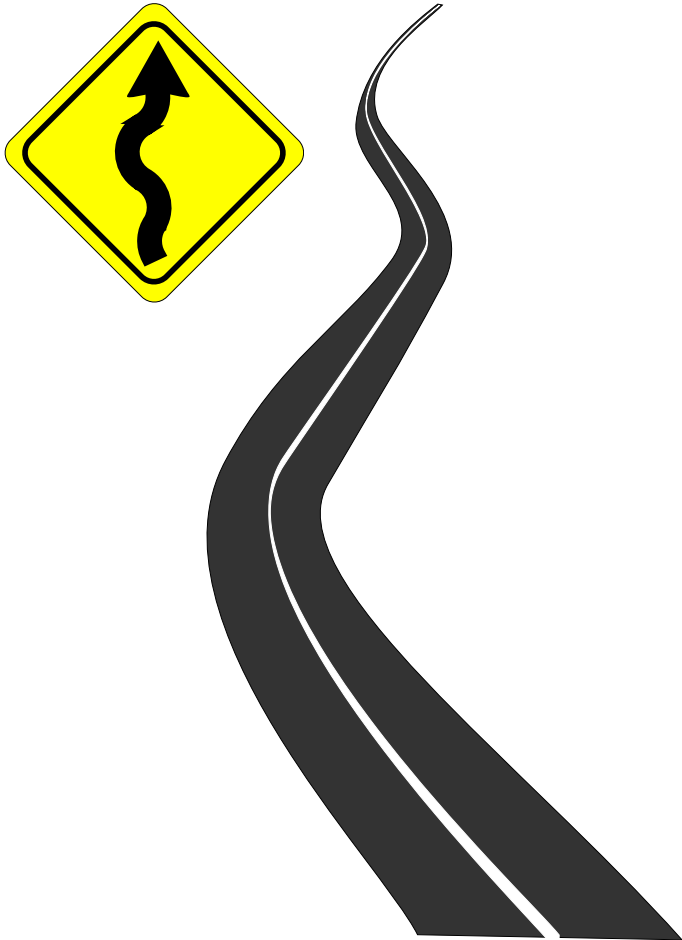
[www.geant.org](http://www.geant.org)

# Agenda - Part I



- Technical Basics
  - Virtual vs. Physical Memory
- Main Memory Dumps
  - Simple
  - Kernel Module
- System Crash dumps
  - Linux Kdump
  - Windows crash dumps

# Agenda - Part II



- Acquisition of Virtual Machine Memory
  - VMware
  - VirtualBox
  - Linux KVM/QEMU
- Swap & Hibernation
  - Linux Swap files/partitions
  - Windows pagefile, hibernation file
- Single Process Memory Dumps
  - Corefiles
  - Process Explorer

# Virtual Machine Memory

# Dump from Virtual Machine Host

- We will assume that one or more guests are compromised, but not the host
  - For memory dumps of compromised hosts - see Part I
  - Analysing compromised VM hosts is yet another story
- Pro
  - Suspend VM while dumping memory - state doesn't change during dump
  - No way for the rootkit on the guest to interfere or notice
- Still need profiles/symbol tables, etc. of the guest for later analysis

# VMware

Live  
Demo

- **vmss2core** tool
  - Input: Suspend (.vmss), snapshot (.vmsn), and full memory (.vmem) files
  - Output: core dump files for typical OS debuggers
    - WinDbg, Gdb, Solaris MDB, Mac OS X
- Howto
  - Suspend the virtual machine
  - Copy the .vmss or .vmsn file from the machines directory (need to access the host)
    - WinDbg: `vmss2core -W vmname.vmsn`
    - Linux: `vmss2core -N vmname.vmsn`
    - Mac OS X: `vmss2core -X64 vmname.vmsn`

# VMware Workstation & ESXi

Live  
Demo

- Collection
  - Suspend the virtual machine
  - Copy the .vmss or .vmsn file from the machines directory
    - Access to the host OS required
- Processing: **vmss2core** tool
  - Input: Suspend (.vmss), snapshot (.vmsn), and full memory (.vmem) files
    - Windows: `vmss2core -W vmname.vmsn`
    - Linux: `vmss2core -N vmname.vmsn`
    - Mac OS X: `vmss2core -X64 vmname.vmsn`
  - Output: core dump files for typical OS debuggers
    - WinDbg, Gdb, Solaris MDB, Mac OS X

# VirtualBox

- Collection
  - Needs GNU binutils (for objdump)

```
$ vboxmanage debugvm "Name" dumpvmcore --filename forensic-img.elf
```

- Processing

```
$ objdump -h forensic-img.elf | egrep -w "(Idx|load1)"
```

Idx	Name	Size	VMA	LMA	File off	Algn
1	load1	40000000	0000000000000000	0000000000000000	00000720	2**0

```
$ size=0x40000000; off=0x720; head -c $(( $size+$off )) forensic-img.elf |  
tail -c +$(( $off+1 )) > forensic-img.raw
```

```
$ size=0x40000000; off=0x720; dd if=forensic-img.elf of=forensic-img.raw  
bs=1 skip=$(( $off )) count=$(( 0x$size ))
```

- The raw image can be analysed with forensic tools



# KVM/Qemu

- Collection

```
# virsh dump domain corefilepath
```

- *domain*: domain Name of the virtual guest
- *corefilepath*: path of the dump file on the host

- Processing

- Can be used like a crash dump of the guest operating system
- Qemu core dump format for Qemu backed VMs can be used directly in volatility

# Hyper-V

- Ensure the VM has checkpoints enabled,
  - Configured as Standard checkpoints
- Taking a Checkpoint
  - Hyper-V Manager (GUI)
    - VM → Checkpoint
  - PowerShell (CLI)

```
PS > Checkpoint-VM -Name VMName
```
  - Hyper-V Memory checkpoints have `.vmrs` extension
- Processing steps needed?

Checkpoints

You can configure options for checkpoints for this virtual machine.

Checkpoint Type

Enable checkpoints

Select the type of checkpoint that will be created when users choose to checkpoint this virtual machine.

Production checkpoints

Use backup technology in the guest operating system to create data-consistent checkpoints that don't include information about running applications.

Create standard checkpoints if the guest does not support creation of production checkpoints.

Take a checkpoint with full application state if it is not possible to use backup technology inside the guest operating system.

Standard checkpoints

Create application-consistent checkpoints that capture the current state of applications.

Use automatic checkpoints

Checkpoint File Location

Choose where to store the checkpoint configuration and checkpoint saved state files for this virtual machine.

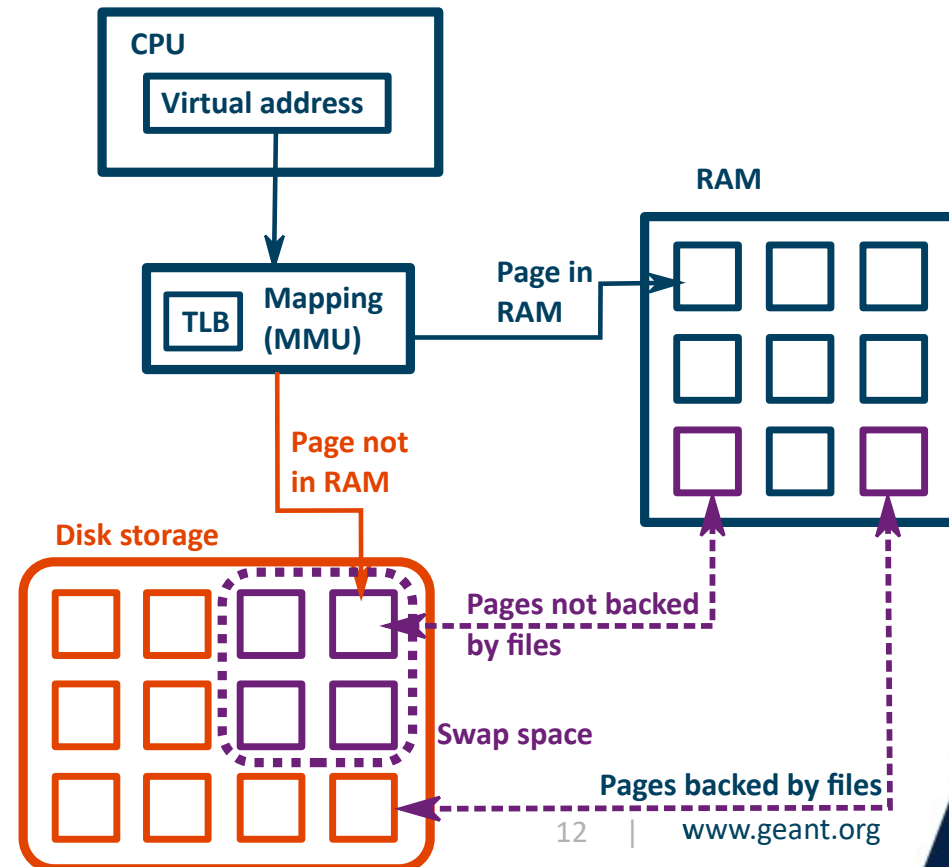
D:\CHECKPOINTS

Browse...

# Swap Space & Hibernation

# Swap Space

- When space in RAM gets filled up, pages are “*paged out*” to backing storage
- When needed (page fault), pages are “*paged in*” back to RAM
- Can be a (disk) partition
  - Linux/Solaris: Type 82
  - BSI: Type B8
- Or swap file
  - Windows:  
`%SystemRoot%\pagefile.sys`  
`%SystemRoot%\swapfile.sys`



# Linux Swap

- Swap partition can be imaged with forensic imaging tools
  - I.e. `dd`, see next webinar
- Swap files can be copied like normal files
- Both best done post-mortem
- Search for forensic artefacts
  - Passwords, URLs, encryption keys, etc.
    - `strings`
    - `yara`
    - Hex editor (check performance with gigabyte-sized files)
    - `swap_digger`

# Windows Swap Files

- Two files
  - %SystemRoot%\pagefile.sys and %SystemRoot%\swapfile.sys
  - Latter is used only for Universal Windows Platform (UWP) apps
    - Apps will be completely swapped out to this file,
    - pagefile.sys is not used for them
  - Hidden “protected operating system file”
  - Like Linux swap, they don’t have full memory copies
- Paging needs not to be disabled (enabled by default)
- Assert that pagefile is not zeroed out on shutdown

```
HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\ClearPageFileAtShutdown # 1
```

# Windows Swap File Collection

- Collection
  - Locked by Windows OS, cannot be accessed from user space
  - To copy
    - Boot into a live CD/stick, or
    - Mount the physical disk on another system, or
    - Extract the file(s) from a forensic disk image
- Processing
  - strings
  - yara
  - Hex editor (check performance with gigabyte-sized files)

# Windows Hibernation File

- *Hibernation* = “suspend to disk” (*Sleep* = “suspend to RAM”)
  - Location: %SystemRoot%\hiberfil.sys (can be configured)
  - Hidden “protected operating system file”
  - Compressed with modified LZ77 algorithm, called “Xpress”
  - Contains kernel session, device drivers, application data
    - Since Windows 8, application data is stored in swapfile.sys
  - Will be encrypted by BitLocker (or other FDE), if used
- To hibernate
  - Activate hibernation: `powercfg.exe /hibernate on`
  - Hibernate: `shutdown /h`



# Windows Hibernation File Collection

- Collection
  - Copy file through forensic disk imaging (see next webinar)
- Processing
  - volatility

```
vol.py imagecopy -f hiberfil.sys -O hiber.img --profile=WinXXXX
```

- WinXXXX: Need correct Windows profile
- hibr2bin

```
hibr2bin /I c:\hiberfil.sys /o z:\hiber.img
```

# Single Process Memory Dumps

# Single Process Dumps

- Taking the image of a whole system sometimes may be too much
- Need only to investigate one (or a few) process(es)
- Lack of privileges to dump a whole system, i.e. just ordinary users
- Not allowed to take an image of the whole system, for confidentiality or privacy reasons

# Linux Processes: `/proc/$pid/mem`

Live Demo

- Naive approach: `cat /proc/$pid/mem`

```
:~> echo $$  
21349  
:~> cat /proc/21349/mem > testmemdump  
cat: /proc/21349/mem: Input/output error  
:~> ls -l /proc/21349/mem  
-rw----- 1 moeller users 0 Nov 25 11:06 /proc/21349/mem  
:~> id  
uid=1000(moeller) gid=100(users) ...
```

- What happened?
  - `/proc/$pid/mem` contains only the process' mapped pages
  - Accessing an unmapped page yields EIO (I/O error)

# Linux Processes: Copy with `/proc/$pid/maps`



- How to make sure that only mapped pages are dumped?
- Utilize `/proc/$pid/maps` to know the mapped regions

```
:~> cat /proc/self/maps
55d00e807000-55d00e809000 r--p 000f1000 00:2f 2038838 /bin/bash
55d00e809000-55d00e80d000 rw-p 000f3000 00:2f 2038838 /bin/bash
55d00e80d000-55d00e81b000 rw-p 00000000 00:00 0
55d00f325000-55d00f8d2000 rw-p 00000000 00:00 0 [heap]
7fb0163e2000-7fb016408000 r-xp 00000000 00:2f 3055433 /lib64/libtinfo.so.6.1
...
7fb017043000-7fb017044000 r--p 00025000 00:2f 3033470 /lib64/ld-2.26.so
7fb017044000-7fb017045000 rw-p 00026000 00:2f 3033470 /lib64/ld-2.26.so
7fb017045000-7fb017046000 rw-p 00000000 00:00 0
7fff9af16000-7fff9af37000 rw-p 00000000 00:00 0 [stack]
7fff9af8a000-7fff9af8e000 r--p 00000000 00:00 0 [vvar]
7fff9af8e000-7fff9af90000 r-xp 00000000 00:00 0 [vdso]
ffffffffffffff600000-ffffffffffffff601000 --xp 00000000 00:00 0 [vsyscall]
```

# Linux Processes: Dump Tools

Live  
Demo

- Several Scripts and Programs for that
  - Python: memdump.py  
(<https://gist.github.com/Dbof/b9244cfc607cf2d33438826bee6f5056>)
  - Shell: See next page
- GDB/gcore
- Sysinternals Procdump-for-linux
  - <https://github.com/Sysinternals/ProcDump-for-Linux>
- Manually with kill (and some more stuff)

# Linux Process Dump from Shell



- `dd if=/proc/$pid/mem of=/path/to/dump bs=1 skip=$((0x$col1)) count=$((0x$end - 0x$start))`
  - If getting seek errors, use method below
    - May be ignored though
- `xxd -s 0x$start -l=$((0x$end - 0x$start)) /proc/$pid/mem`
  - Convert back to binary with `xxd -r` (or use `dd` in the first place)
- ***\$start/\$end***: Column 1/2 from process memory map file
- Practical tips
  - Avoid `$$` or `/proc/self` expansion - may evaluate to the id of the process doing the dump, not the process to be dumped
  - Use `kill -SIGSTOP` to freeze process while dumping, unfreeze with `kill -SIGCONT`

# Linux Processes: Core dumps



- GDB/gcore
  - `gcore -a -o $prefix $pid`
  - File will be named `$prefix.$pid`, otherwise `gcore.$pid`
  - May not work if process blocks ptrace
  - From within a gdb: `generate-core-file`

- kill

```
# ulimit -S -c unlimited # unset soft limit on core dumps
# cat /proc/sys/kernel/core_pattern # where is dump is written to
# kill -SIGSEGV $pid # see signal(7)
```

- Does not work if process ignores or handles signals
- Does not work with setuid/gid binaries
  - Unless `kernel_suid_dumpable = 2 (or 3)` (**Beware** ☠)



# Windows Processes: procexp

- Sysinternals *Process Explorer* (GUI)
  - Use *full dump*
  - See also: *VMMMap*

Live Demo

VMMMap - Sysinternals: www.sysinternals.com

Process: sqlwriter.exe  
PID: 3216

Committed: 34.896 K

Private Bytes: 1.796 K

Working Set: 7.252 K

Type	Size	Committed	Private	Total WS
Total	4.275.984 K	34.896 K	1.796 K	7.252 K
Free	137.434.677.824 K			
Heap	1.384 K	540 K	476 K	412 K
Image	27.284 K	27.284 K	584 K	6.016 K
Managed Heap				
Mapped File	4.484 K	4.484 K		180 K
Page Table	400 K	400 K	400 K	400 K
Private Data	4.230.364 K	56 K	56 K	48 K
Shareable	5.588 K	1.852 K		168 K
Stack	4.096 K	280 K	280 K	28 K
Unusable	2.384 K			

Address	Type	Size	Committed	Private	Total WS	Private
00000007FFE0000	Private Data	4 K	4 K	4 K	4 K	4 K
00000007FFE5000	Private Data	4 K	4 K	4 K	4 K	4 K
000000C813400000	Private Data	2.048 K	20 K	20 K	20 K	20 K
000000C813600000	Thread Stack	2.048 K	140 K	140 K	16 K	1 K
000000C813C00000	Thread Stack	2.048 K	140 K	140 K	12 K	1 K
000002C6D2B00000	Heap (Shareable)	64 K	64 K		4 K	
000002C6D2DC0000	Mapped File	12 K	12 K		12 K	

Process Explorer - Sysinternals: www.sysinternals.com [WINDEV2110EVAL\User] (Administrator)

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
AggregatorHost.exe		1.060 K	3.996 K	3852		
svchost.exe		11.524 K	23.348 K	3088	Host Process for Windows S...	Microsoft Corporation
svchost.exe		1.248 K	4.308 K	3108	Host Process for Windows S...	Microsoft Corporation
svchost.exe		2.376 K	9.804 K	3124	Host Process for Windows S...	Microsoft Corporation
svchost.exe		4.100 K	20.984 K	3152	Host Process for Windows S...	Microsoft Corporation
svchost.exe		8.440 K	20.604 K	3176	Host Process for Windows S...	Microsoft Corporation
sqlwriter.exe		1.052 K	34.492 K	3216	SQL Server VSS Writer - 64 Bit	Microsoft Corporation
vmtoolsd.exe		19.364 K	3256	3256	VMware Tools Core Service	VMware, Inc.
VGAuthService.exe		8.512 K	3264	3264	VMware Guest Authenticatio...	VMware, Inc.
vm3dservice.exe		4.824 K	3312	3312	VMware SVGA Helper Service	VMware, Inc.
vm3dservice.exe		5.348 K	3700	3700	VMware SVGA Helper Service	VMware, Inc.
MsMpEng.exe		118.348 K	3548	3548	Antimalware Service Execut...	Microsoft Corporation
dllhost.exe		16.832 K	4144	4144	COM Surrogate	Microsoft Corporation
msdtc.exe		9.076 K	4384	4384	Microsoft Distributed Transa...	Microsoft Corporation
svchost.exe		21.004 K	4100	4100	Host Process for Windows S...	Microsoft Corporation
svchost.exe		9.152 K	4876	4876	Host Process for Windows S...	Microsoft Corporation
svchost.exe		18.904 K	4628	4628	Host Process for Windows S...	Microsoft Corporation
svchost.exe		16.004 K	5060	5060	Host Process for Windows S...	Microsoft Corporation
svchost.exe		27.128 K	4264	4264	Host Process for Windows S...	Microsoft Corporation
svchost.exe		2.764 K	14.228 K	6136	Host Process for Windows S...	Microsoft Corporation
svchost.exe		1.868 K	8.092 K	5768	Host Process for Windows S...	Microsoft Corporation
ctfmon.exe		19.980 K	5468	5468	CTF Loader	Microsoft Corporation
svchost.exe		14.520 K	5596	5596	Host Process for Windows S...	Microsoft Corporation
svchost.exe		18.432 K	5624	5624	Host Process for Windows S...	Microsoft Corporation
svchost.exe		29.144 K	6060	6060	Host Process for Windows S...	Microsoft Corporation
svchost.exe		14.228 K	6136	6136	Host Process for Windows S...	Microsoft Corporation
svchost.exe		1.868 K	8.092 K	5768	Host Process for Windows S...	Microsoft Corporation

Window

- Set Affinity...
- Set Priority
- Kill Process (Del)
- Kill Process Tree (Shift+Del)
- Restart
- Suspend
- Debug
- Create Dump**
  - Create Mindump...
  - Create Full Dump...**
- Check VirusTotal
- Properties...
- Search Online... (Ctrl+M)

CPU Usage: 3.71% Commit Charge: 38.31% Processes: 155

# Windows Processes: procdump

Live  
Demo

- Sysinternals *Procdump* (CLI)

```
PS> procdump.exe -ma $pid  
PS> procdump.exe -ma $process_name
```

- Can specify conditions on when to dump
  - In case process is for unresponsive, crashing or consuming excessive resources
- Also possible to take multiple dumps several seconds apart
  - For monitoring resource usage or changes in processes internal state



# Wrapping Up

[www.geant.org](http://www.geant.org)

# What have you learned?

- Memory dumps can be taken from VMs easily
  - However, some processing is necessary for forensic analysis
  - Big plus: No access to guest OS needed
- Hibernation file yields full memory contents
- Swap files/partitions do not
  - But some useful information can still be gained
- Single process memory dumps are easy too
  - Procdump can be used on both Linux and Windows

# Thank you

Any questions?

Next Webinar: *Persistent Storage Acquisition I*

*December 18<sup>th</sup>, 2021*

[www.geant.org](http://www.geant.org)



# References

- VirtualBox
  - <https://www.andreafortuna.org/2017/06/23/how-to-extract-a-ram-dump-from-a-running-virtualbox-machine/>
  - <https://gist.github.com/ssnkhan/60a2ab79480bd966876aac5d2c6c2e68>
- Hibernation file
  - Hiber2bin: <https://github.com/comaeio/Hibr2Bin>
  - Volatility Hibernation file address space  
<https://github.com/volatilityfoundation/volatility/wiki/Hiber-Address-Space>
  - Windows hibernation file for fun 'n' profit,  
[https://www.blackhat.com/presentations/bh-usa-08/Suiche/BH\\_US\\_08\\_Suiche\\_Windows\\_hibernation.pdf](https://www.blackhat.com/presentations/bh-usa-08/Suiche/BH_US_08_Suiche_Windows_hibernation.pdf)
- Linux swap\_digger: [https://github.com/sevagas/swap\\_digger](https://github.com/sevagas/swap_digger)

## References: Books on Forensics

- Michael Hale Ligh, et al: *The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory*, John Wiley & Sons, Inc. 2014, ISBN: 978-1-118-82509-9
- Bruce Nikkel: *Pactical Forensic Imaging*, No Starch Press Inc. 2016, ISBN-13: 978-1-59327-793-2
- Harlan Carvey: *Windows Forensic Analysis*, Syngress Publishing Inc. 2009

# References: Operating System Internals

- Pavel Yosifovich et al: *Windows Internals, Part 1 (System architecture), 7<sup>th</sup> Ed.*, Microsoft Press 2017, ISBN-13: 978-0735684188
- Allievi Andrea et al: *Windows Internals, Part 2 (Developer Reference), 7<sup>th</sup> Ed.*, Microsoft Press 2021, ISBN-13: 978-0135462409
- Robert Love: *Linux Kernel Development 3<sup>rd</sup> Ed.*, Addison-Wesley Professional 2010, ISBN-13: 978-0672329463
- Robert Love: *Linux System Programming: Talking Directly to The Kernel And C Library, 2<sup>nd</sup> Ed.*, O'Reilly 2013, ISBN-13 : 978-1449339531
- The FreeBSD Documentation Project: FreeBSD Handbook, <https://docs.freebsd.org/en/books/handbook/>
- The FreeBSD Documentation Project: FreeBSD Developers' Handbook, <https://docs.freebsd.org/en/books/developers-handbook/>
- The FreeBSD Documentation Project: FreeBSD Architecture Handbook, <https://docs.freebsd.org/en/books/arch-handbook/>
- Marshall Kirk McKusick et al.: *The Design and Implementation of the FreeBSD Operating System: Edition 2*, Addison-Wesley Professional 2014, ISBN-13: 978-0321968975



# References: Images und Testcases

- Computer Forensic Reference Data Sets (CFReDS)  
<http://www.cfreds.nist.gov/>
- Digital Forensics Tool Testing Images  
<http://dftt.sourceforge.net/>
- Digital Forensics Research Workshop (DFRWS)  
<http://www.dfrws.org/>
- Honeynet Project Challenges  
<https://www.honeynet.org/challenges>

# References: Memory Imaging Tools (Open Source)

- Microsoft AVML: <https://github.com/microsoft/avml>
- Volatility LiME: <https://github.com/504ensicsLabs/LiME>
  - Schlafwandlers kcore\_dump  
<https://schlafwandler.github.io/posts/dumping-/proc/kcore/>
- Hal Pomeranz automation script for AVML/LiME:  
<https://github.com/halpomeranz/lmg>
- Velocidex Pmem Suite (lin|win|osx)pmem:  
<https://winpmem.velocidex.com/>
- Moonsols mdd (v 1.3, 2013, for very old Windows versions):  
<https://sourceforge.net/projects/mdd/>

# Sample Forensic Distributions

- SIFT (SAS Investigative Forensic Toolkit): <https://www.sans.org/tools/sift-workstation/>
- CAINE (Computer Aided Investigative Environment): <https://www.caine-live.net/>
- GRML Forensic: <https://grml-forensic.org/>
- ALT Linux Rescue: <https://en.altlinux.org/Rescue>
- BlackArch: <https://blackarch.org/>
- BackBox: <https://www.backbox.org/>
- KALI (formerly Backtrack): <https://www.kali.org/downloads/>
- Matriux: <http://www.matriux.com/>
- Safe Boot Disk (Windows based): [https://www.forensicsoft.com/help/SAFE\\_Boot1-1/](https://www.forensicsoft.com/help/SAFE_Boot1-1/)

## References: Standards

- US NIST Special Publication 800-86 *Guide to Integrating Forensic Techniques into Incident Response*, 2006, <https://doi.org/10.6028/NIST.SP.800-86>
- ENISA *Trainings for Cybersecurity Specialists*, <https://www.enisa.europa.eu/topics/trainings-for-cybersecurity-specialists/online-training-material?tab=articles>
- IETF RFC 3227 *Guidelines for Evidence Collection and Archiving*, <https://tools.ietf.org/html/rfc3227>