Memory Forensics

An introduction

DISCLAIMER

- Speak only for myself
- These are opinions, not facts
- I could be wrong about anything
- Use at your own risk

About Me

- On corporate security team
- Analyze malware as a hobby
- Not an expert by any stretch
- Goal for talk:
  - Introduce concepts, show fun demos

Agenda

- Introduction
- Concepts
- Acquisition methods (demo!)
- Analysis (demo!)
- Wrap-up
- Links, links, links

Agenda

- Introduction
- Concepts
- Acquisition methods (demo!)
- Analysis (demo!)
- Wrap-up
- Links, links, links

Types of Forensics

- Disk/filesystem
- Network/signals
- Memory/volatile

Why Memory?

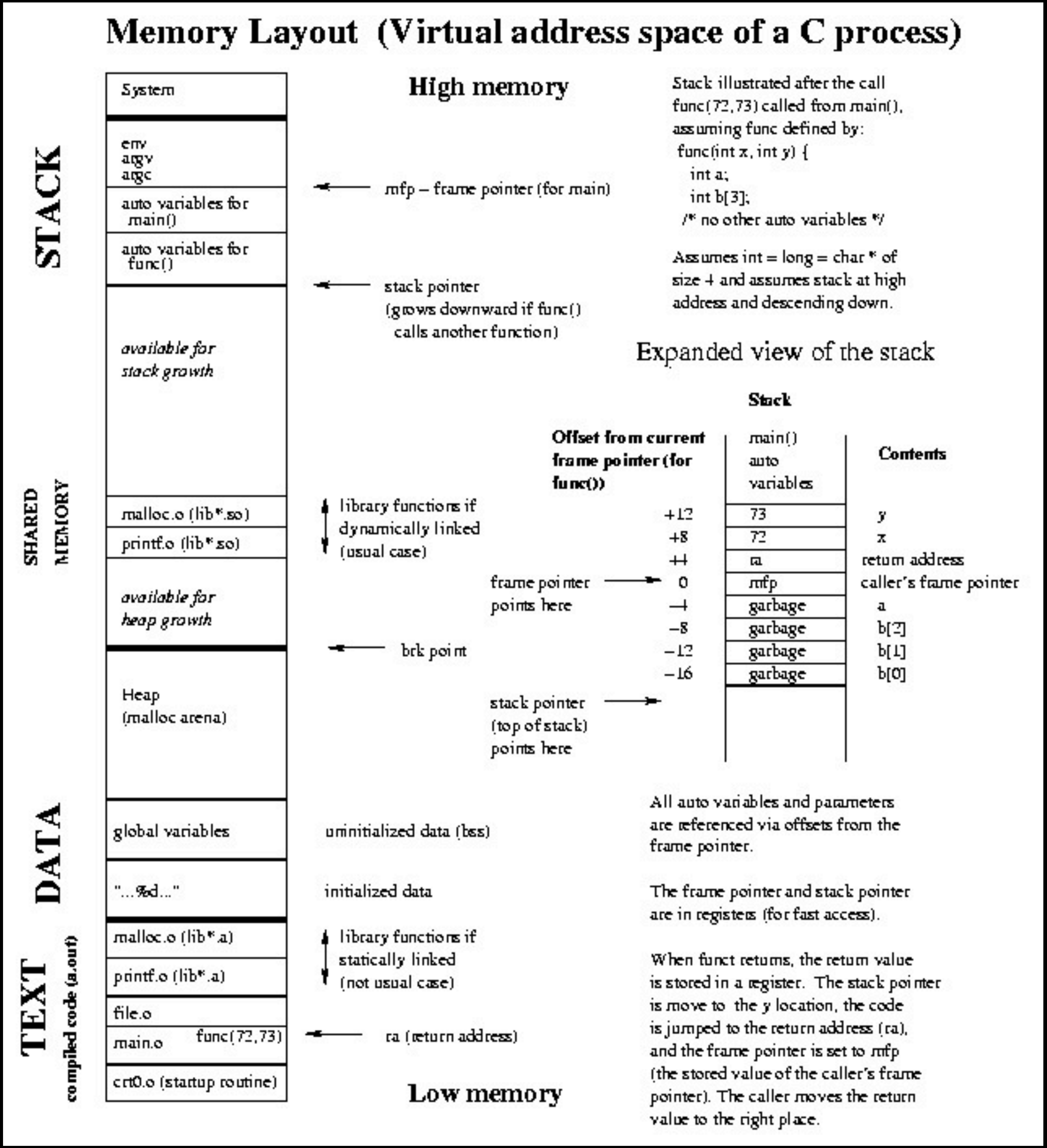
- Unpacked binary
- Observe behavior
- Encryption keys
- Memory-only malware
- Memory-only artifacts

Agenda

- Introduction
- Concepts
- Acquisition methods (demo!)
- Analysis (demo!)
- Wrap-up
- Links, links, links

What Does Memory Look Like?

- Objects: Linked lists, structs, mapped files
  - Process lists, sockets, file handles, jump tables, registry hives
- Memory pages-different access privileges
- Process space, global & local variables



Memory Layout (Virtual address space of a C process)

STACK
SHARED MEMORY
DATA
TEXT
compiled code (a.out)
System
env
argv
argc
auto variables for main()
auto variables for func()
available for stack growth
malloc.o (lib*.so)
printf.o (lib*.so)
available for heap growth
Heap (malloc arena)
global variables
"...%d..."
malloc.o (lib*.a)
printf.o (lib*.a)
file.o
main.o
func(72,73)
crt0.o (startup routine)
High memory
mfp – frame pointer (for main)
stack pointer (grows downward if func() calls another function)
library functions if dynamically linked (usual case)
brk point
uninitialized data (bss)
initialized data
library functions if statically linked (not usual case)
ra (return address)
Low memory
frame pointer points here
stack pointer (top of stack) points here

Stack illustrated after the call func(72,73) called from main(), assuming func defined by:

```
 func(int x, int y) {
   int a;
   int b[3];
  /* no other auto variables */
```

Assumes int = long = char * of size 4 and assumes stack at high address and descending down.

Expanded view of the stack

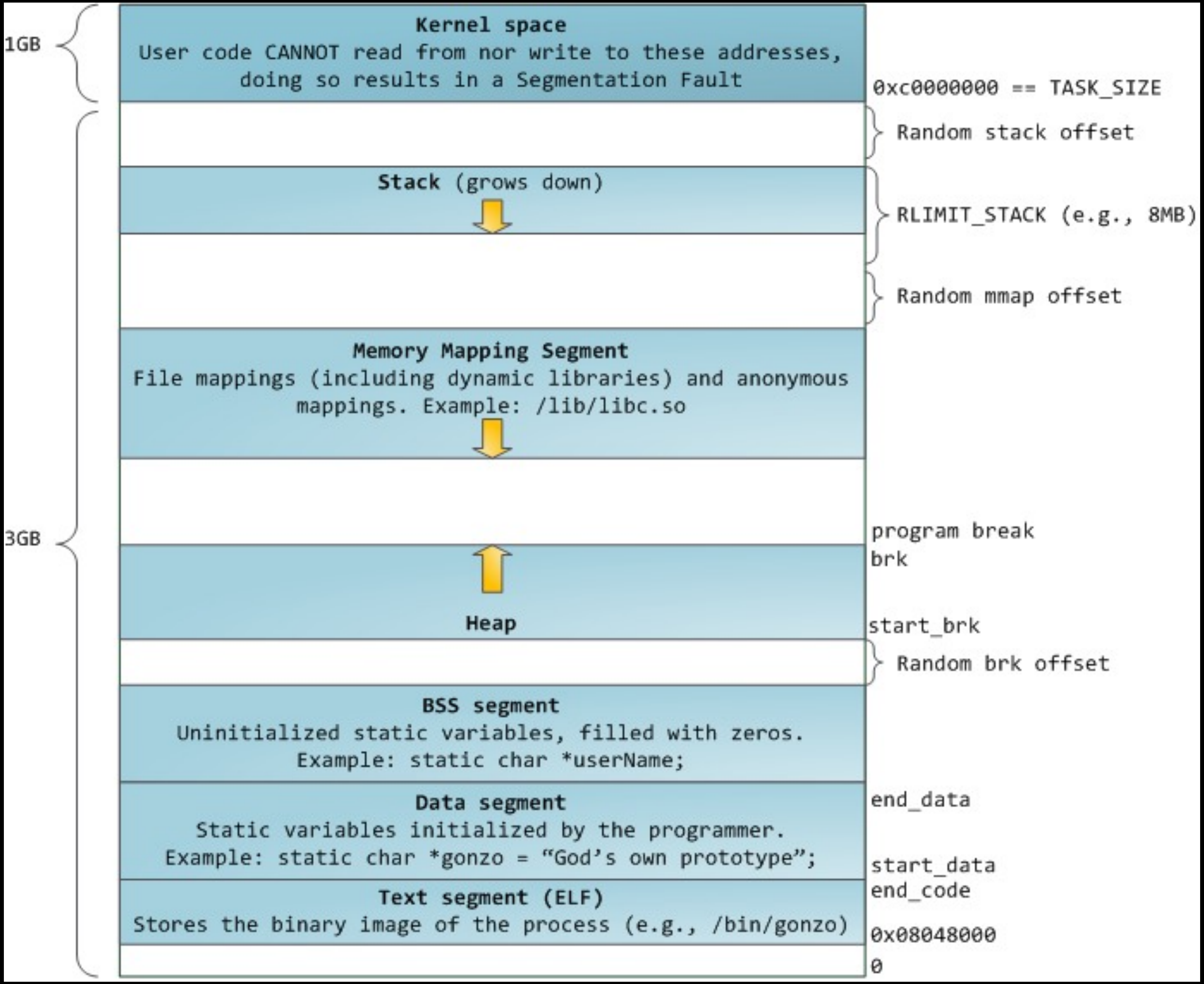
Offset from current frame pointer (for func())	Stack	Contents
	main() auto variables	
+12	73	y
+8	72	x
+4	ra	return address
0	mfp	caller's frame pointer
−4	garbage	a
−8	garbage	b[2]
−12	garbage	b[1]
−16	garbage	b[0]

All auto variables and parameters are referenced via offsets from the frame pointer.

The frame pointer and stack pointer are in registers (for fast access).

When funct returns, the return value is stored in a register. The stack pointer is move to the y location, the code is jumped to the return address (ra), and the frame pointer is set to mfp (the stored value of the caller's frame pointer). The caller moves the return value to the right place.

http://www.cs.uleth.ca/~holzmann/C/system/memorylayout.gif

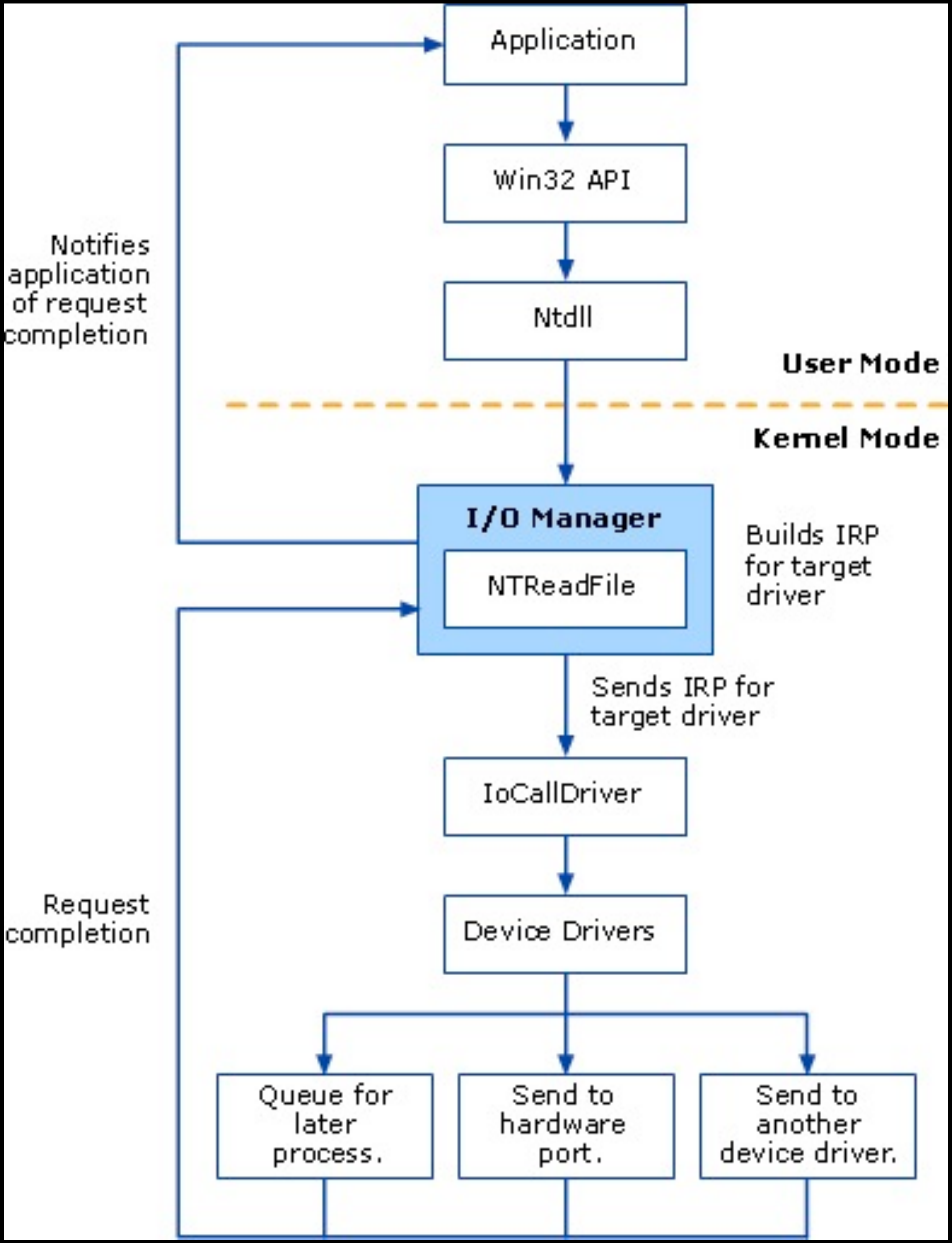


1GB
3GB
Kernel space
User code CANNOT read from nor write to these addresses, doing so results in a Segmentation Fault
0xc0000000 == TASK_SIZE
Random stack offset
Stack (grows down)
RLIMIT_STACK (e.g., 8MB)
Random mmap offset
Memory Mapping Segment
File mappings (including dynamic libraries) and anonymous mappings. Example: /lib/libc.so
program break
brk
Heap
start_brk
Random brk offset
BSS segment
Uninitialized static variables, filled with zeros.
Example: static char *userName;
Data segment
Static variables initialized by the programmer.
Example: static char *gonzo = “God’s own prototype”;
end_data
start_data
end_code
Text segment (ELF)
Stores the binary image of the process (e.g., /bin/gonzo)
0x08048000
0

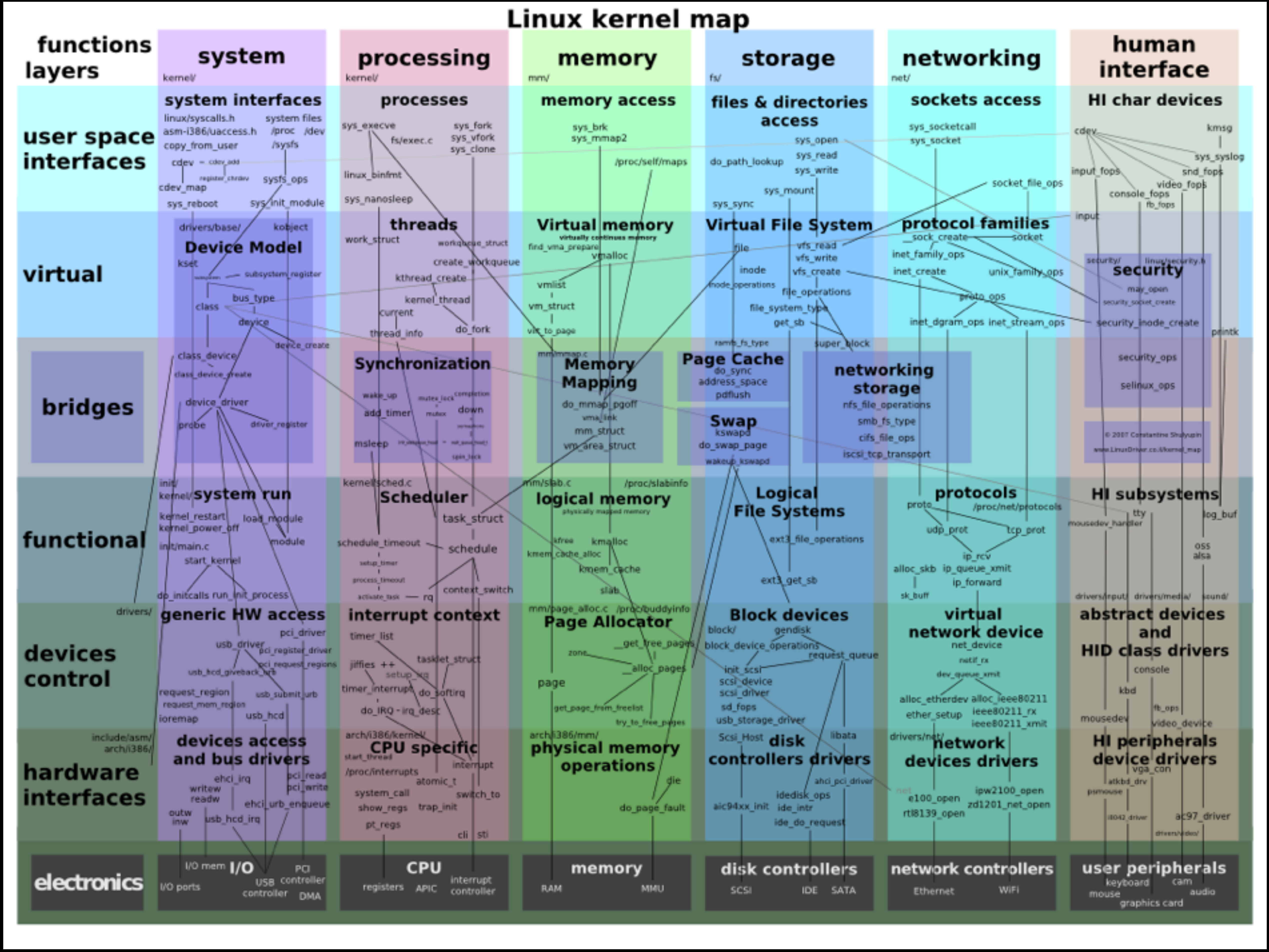
http://duartes.org/gustavo/blog/post/anatomy-of-a-program-in-memory

http://duartes.org/gustavo/blog/post/how-the-kernel-manages-your-memory

http://duartes.org/gustavo/blog/post/page-cache-the-affair-between-memory-and-files



http://technet.microsoft.com/en-us/library/cc776371(v=ws.10).aspx



http://upload.wikimedia.org/wikipedia/commons/5/5b/Linux_kernel_map.png

Sidebar...

- Security pros need deeper knowledge
  - than other tech pros
- Ex: Developer, how inputs are handled
- Ex: Sysadmin, how kernel & filesystem work

Agenda

- Introduction
- Concepts
- Acquisition methods (demo!)
- Analysis (demo!)
- Wrap-up
- Links, links, links

Software

- Access raw device
- Install custom driver/kernel module
- Swap file on disk
- Hibernation image on disk
    - hiberfil.sys (Win)
    - sleepimage (OSX)

Examples

- Memoryze & Memoryze for the Mac
- LiME
- F-Response
- FTK Imager
- DumpIt
- FastDump Pro

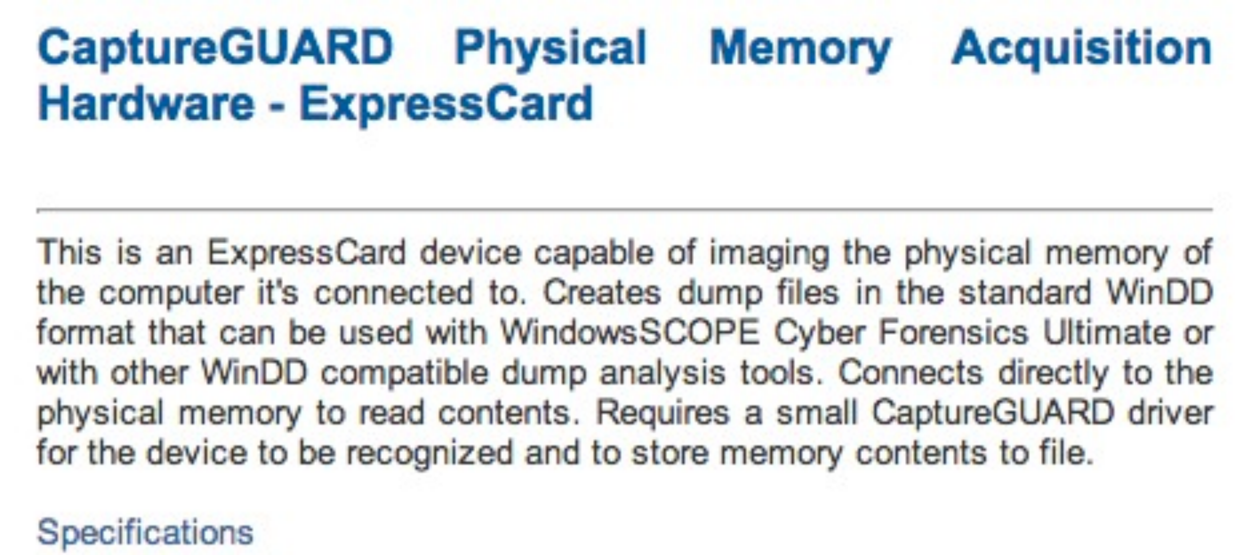
http://www.forensicswiki.org/wiki/Tools:Memory_Imaging

Direct Memory Access

"Systems may be vulnerable to a DMA attack by an external device if they have a FireWire, ExpressCard, Thunderbolt, or other expansion port that, like PCI and PCI-Express in general, hooks up attached devices directly to the physical address space."

http://en.wikipedia.org/wiki/DMA_attack

View Full-Size Image



CaptureGUARD Physical Memory Acquisition Hardware - ExpressCard

This is an ExpressCard device capable of imaging the physical memory of the computer it's connected to. Creates dump files in the standard WinDD format that can be used with WindowsSCOPE Cyber Forensics Ultimate or with other WinDD compatible dump analysis tools. Connects directly to the physical memory to read contents. Requires a small CaptureGUARD driver for the device to be recognized and to store memory contents to file.

Specifications

http://www.windowsscope.com

http://www.breaknenter.org/projects/inception/

http://digitalfire.ucd.ie/?page_id=430

http://macfwdump.sourceforge.net/

Cold-boot

“The attack relies on the data remanence property of DRAM and SRAM to retrieve memory contents which remain readable in the seconds to minutes after power has been removed.”

http://en.wikipedia.org/wiki/Cold_boot_attack

https://citp.princeton.edu/research/memory/

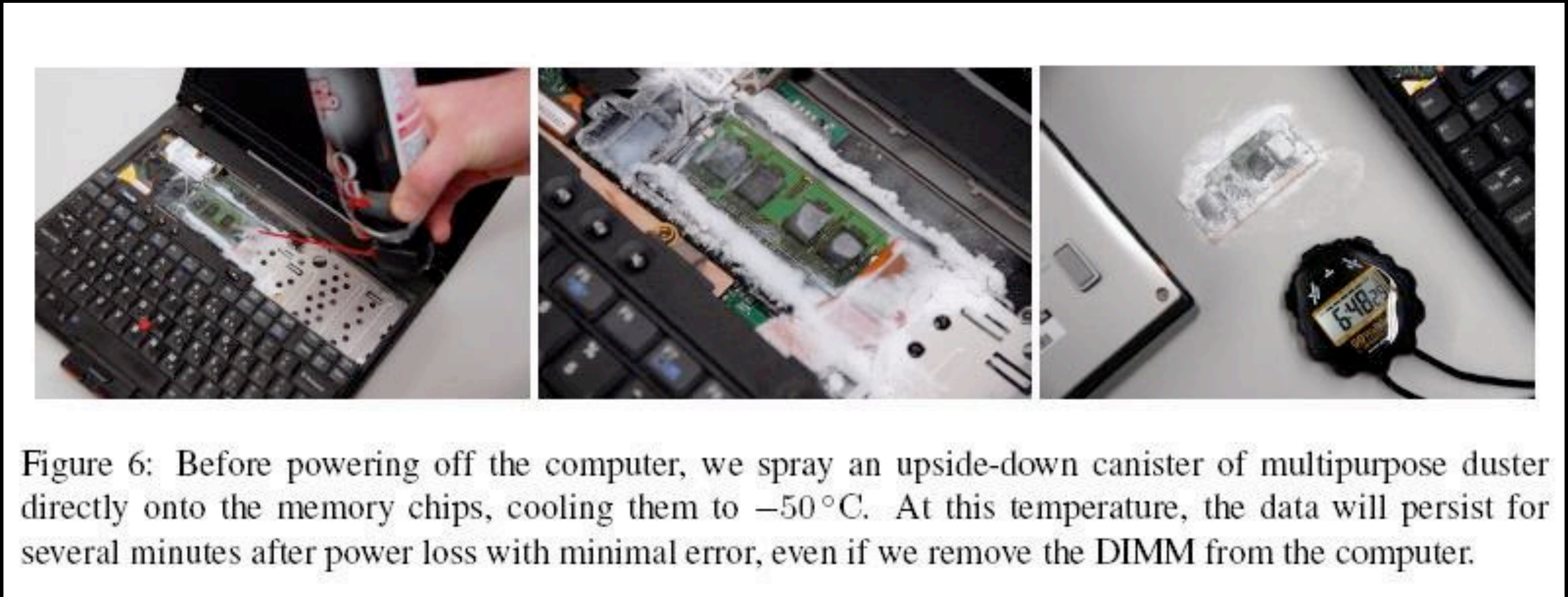
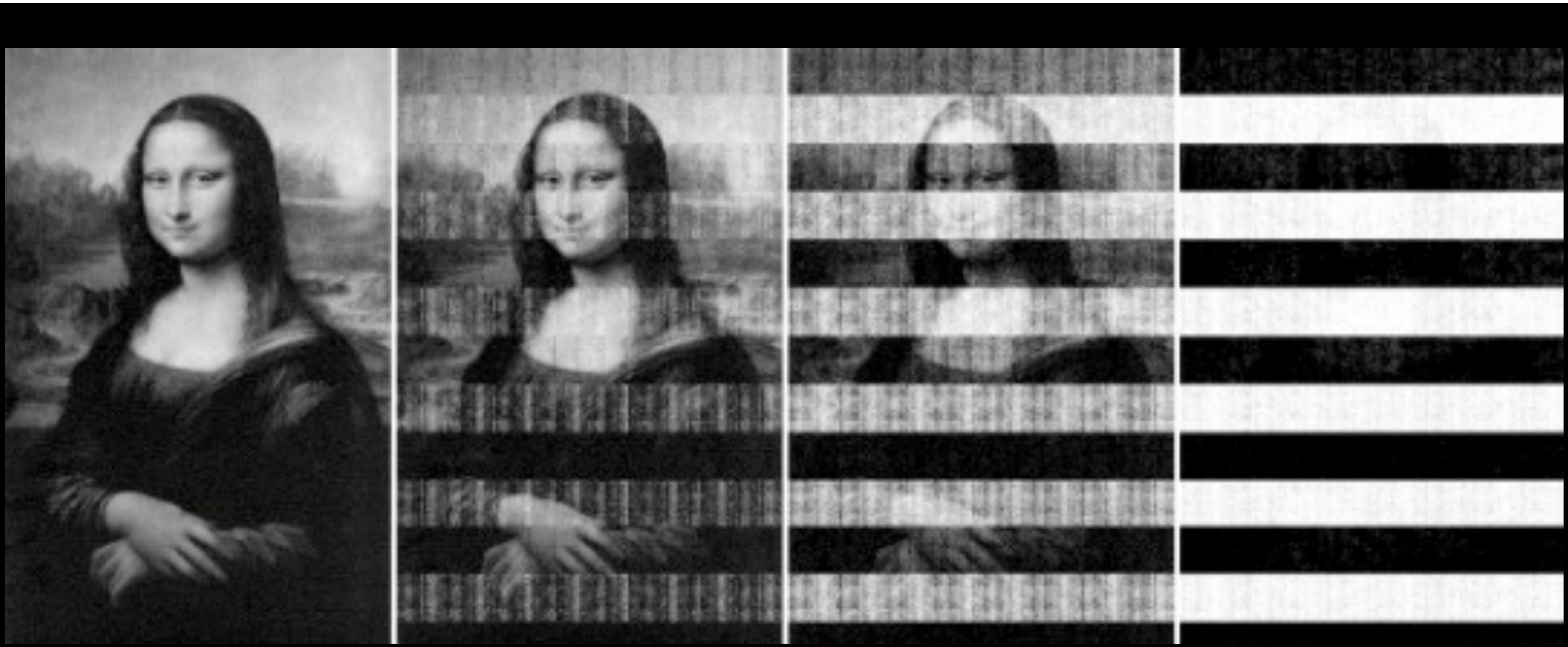


Figure 6: Before powering off the computer, we spray an upside-down canister of multipurpose duster directly onto the memory chips, cooling them to $-50\,^{\circ}\mathrm{C}$. At this temperature, the data will persist for several minutes after power loss with minimal error, even if we remove the DIMM from the computer.



http://osarena.net/hacks-guides/tresor-profilaxte-to-linux-sas-apo-tis-cold-boot-epithesis.html

DEMO (click me!)

Build LiME
Create Volatility profile
Dump memory over TCP
Find bash history

Important Notes!

- Don’t build LiME or mem profile on victim!
  - Use virtual machine with same OS/kernel
  - Build module & profile ahead of time
    - if you can (speed up response)
- Requires gcc, gdb, make, etc

Agenda

- Introduction
- Concepts
- Acquisition methods (demo!)
- Analysis (demo!)
- Wrap-up
- Links, links, links

Suspicious Signs

- Handles to other processes
- Missing from one or more process list
- Has injected sections
- Holds suspicious mutex

DKOM

- Direct Kernel Object Manipulation
  - Unlink process from _EPROCESS list
  - CSRSS process also has handles
    - and internal list

Ligh, M.H., Adair, S., Hartstein, B., & Richard, M. (2011) Malware Analyst's Cookbook and DVD. Indianapolis: Wiley.

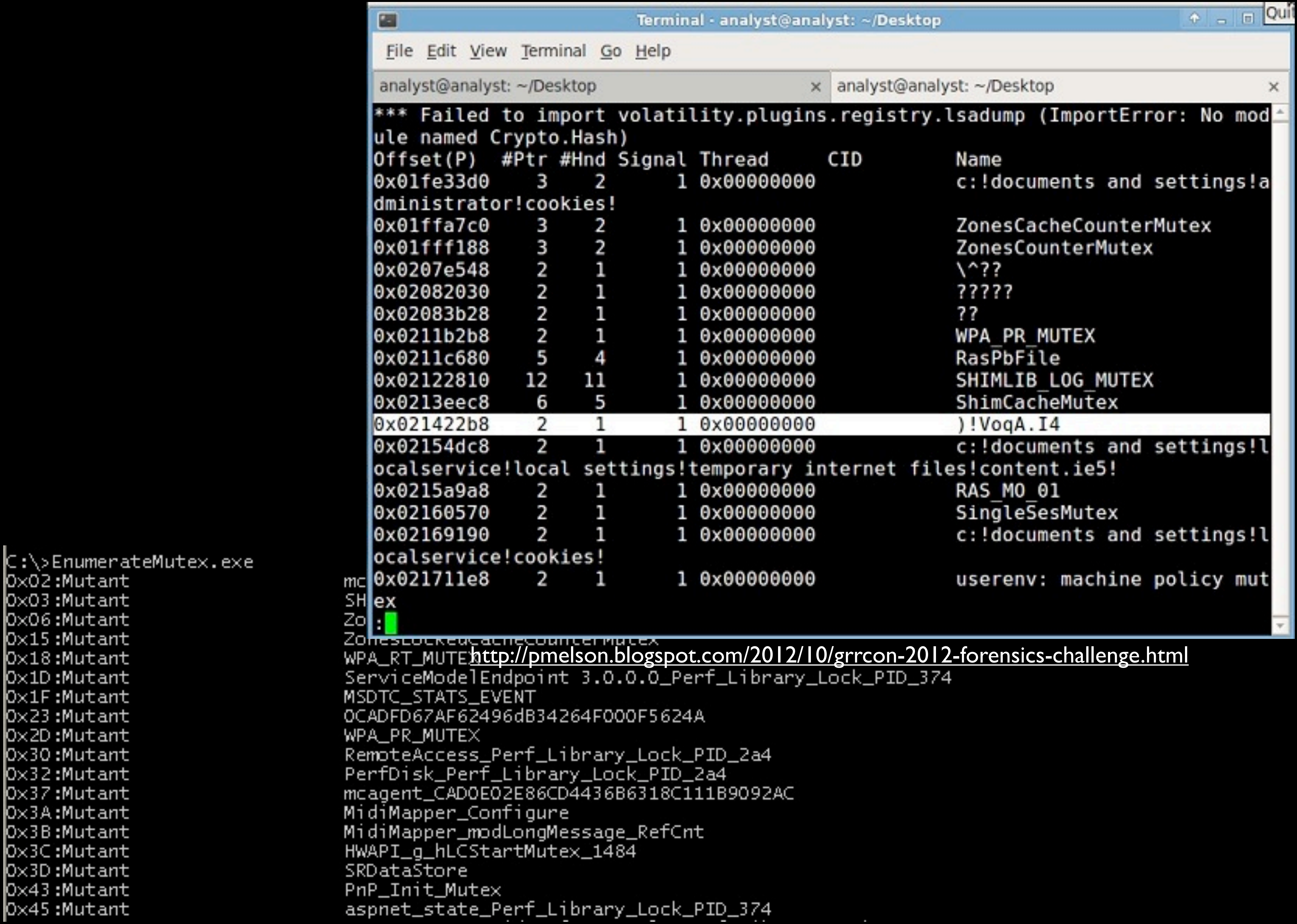
Process Injection

- Process Environment Block
  - Command line & arguments
  - Three lists of the loaded DLLs
    - Could unlink list, but VAD has map
      - Tampering w/VAD requires rootkit

Ligh, M.H., Adair, S., Hartstein, B., & Richard, M. (2011) Malware Analyst's Cookbook and DVD. Indianapolis: Wiley.

Misc

- Process hollowing (similar to injection)
  - Start legit binary in suspended thread
    - Replace the image, resume thread
- Mutex
  - Ensure only one copy of malware runs
    - or avoid concurrency w/specific prog


Terminal - analyst@analyst: ~/Desktop
File Edit View Terminal Go Help
analyst@analyst: ~/Desktop
analyst@analyst: ~/Desktop
Quit

```
*** Failed to import volatility.plugins.registry.lsadump (ImportError: No mod
ule named Crypto.Hash)
Offset(P)  #Ptr #Hnd Signal Thread     CID        Name
0x01fe33d0    3    2      1 0x00000000            c:!documents and settings!a
dministrator!cookies!
0x01ffa7c0    3    2      1 0x00000000            ZonesCacheCounterMutex
0x01fff188    3    2      1 0x00000000            ZonesCounterMutex
0x0207e548    2    1      1 0x00000000            \^??
0x02082030    2    1      1 0x00000000            ?????
0x02083b28    2    1      1 0x00000000            ??
0x0211b2b8    2    1      1 0x00000000            WPA_PR_MUTEX
0x0211c680    5    4      1 0x00000000            RasPbFile
0x02122810   12   11      1 0x00000000            SHIMLIB_LOG_MUTEX
0x0213eec8    6    5      1 0x00000000            ShimCacheMutex
0x021422b8    2    1      1 0x00000000            )!VoqA.I4
0x02154dc8    2    1      1 0x00000000            c:!documents and settings!l
ocalservice!local settings!temporary internet files!content.ie5!
0x0215a9a8    2    1      1 0x00000000            RAS_MO_01
0x02160570    2    1      1 0x00000000            SingleSesMutex
0x02169190    2    1      1 0x00000000            c:!documents and settings!l
ocalservice!cookies!
0x021711e8    2    1      1 0x00000000            userenv: machine policy mut
ex
:
```

http://pmelson.blogspot.com/2012/10/grrcon-2012-forensics-challenge.html

```
C:\>EnumerateMutex.exe
0x02:Mutant                mc
0x03:Mutant                SH
0x06:Mutant                Zo
0x15:Mutant                ZonesLockedCacheCounterMutex
0x18:Mutant                WPA_RT_MUTEX
0x1D:Mutant                ServiceModelEndpoint 3.0.0.0_Perf_Library_Lock_PID_374
0x1F:Mutant                MSDTC_STATS_EVENT
0x23:Mutant                0CADFD67AF62496dB34264F000F5624A
0x2D:Mutant                WPA_PR_MUTEX
0x30:Mutant                RemoteAccess_Perf_Library_Lock_PID_2a4
0x32:Mutant                PerfDisk_Perf_Library_Lock_PID_2a4
0x37:Mutant                mcagent_CAD0E02E86CD4436B6318C111B9092AC
0x3A:Mutant                MidiMapper_Configure
0x3B:Mutant                MidiMapper_modLongMessage_RefCnt
0x3C:Mutant                HWAPI_g_hLCStartMutex_1484
0x3D:Mutant                SRDataStore
0x43:Mutant                PnP_Init_Mutex
0x45:Mutant                aspnet_state_Perf_Library_Lock_PID_374
```

http://labs.alienvault.com/labs/index.php/2009/malware-exploring-mutex-objects/

DEMO (click me!)

Collect artifacts to net share
Import artifacts to Redline
Discover injected memory
Locate events in timeline

(Not shown: Creating the collector)

Agenda

- Introduction
- Concepts
- Acquisition methods (demo!)
- Analysis (demo!)
- Wrap-up
- Links, links, links

Wrap-up

- Memory forensics offer unique advantages
- Concealment techniques leave a trail
- Tools can help, but knowledge is required
  - Study system internals
- Many free tools & guides exist
  - Barrier to entry is low!

Pop Quiz

Pop Quiz

- Name one interface for DMA attack

Pop Quiz

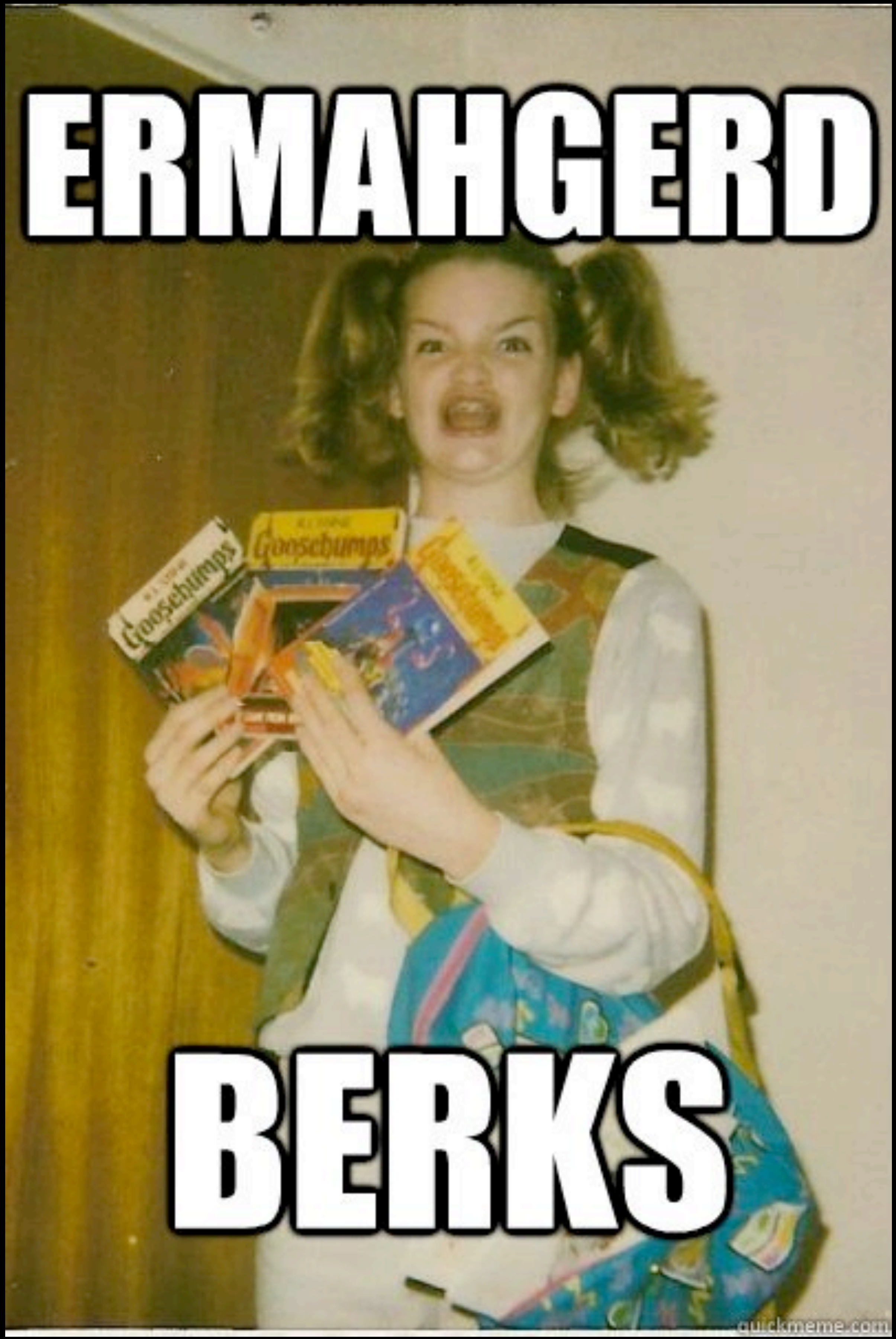
- Name one interface for DMA attack
- What does DKOM stand for?

Pop Quiz

- Name one interface for DMA attack
- What does DKOM stand for?
- Name a software memory acquisition tool

Agenda

- Introduction
- Concepts
- Acquisition methods (demo!)
- Analysis (demo!)
- Wrap-up
- Links, links, links



N BLERGS

http://www.quickmeme.com/meme/3otxsn/

Malware Analyst's Cookbook and DVD
http://www.malwarecookbook.com/

SecurityXploded
http://securityxploded.com/malware-memory-forensics.php

Forensics Wiki
http://www.forensicswiki.org/

Gustavo Duarte
http://duartes.org/gustavo/blog/

Windows Incident Response
http://windowsir.blogspot.com/

Journey Into Incident Response
http://journeyintoir.blogspot.com/

contagio malware dump
http://contagiodump.blogspot.com/

Practical Malware Analysis
http://practicalmalwareanalysis.com/

DigitalFIRE
http://digitalfire.ucd.ie/

Memory Forensics
http://memoryforensics.blogspot.com/

APTish Attack via Metasploit
http://www.sysforensics.org/

Linux Sleuthing
http://linuxsleuthing.blogspot.com/

DeepEnd Research
http://www.deependresearch.org/

SEMPERSECURUS
http://sempersecurus.blogspot.com/

http://www.webdesignhot.com/free-vector-graphics/electric-tools-vector-set/

Memoryze

http://www.mandiant.com/resources/download/memoryze

Memoryze for the Mac

http://www.mandiant.com/resources/download/mac-memoryze

LiME

https://code.google.com/p/lime-forensics/

Inception

http://www.breaknenter.org/projects/inception/

Volatility

https://www.volatilesystems.com/default/volatility

Redline

http://www.mandiant.com/resources/download/redline

Yara

http://code.google.com/p/yara-project/

Cuckoo Sandbox

http://www.cuckoosandbox.org/

Thanks!

Brian Keefer

http://rants.effu.se

https://twitter.com/chort0

https://alpha.app.net/chort

http://www.SMTPS.net

chort0 on Freenode

Slides: http://www.SMTPS.net/pub/presentations/CCSF_Mem_Forensics.pdf