MAC OS X
PHYSICAL MEMORY ANALYSIS

Matthieu Suiche – msuiche(at)moonsols.com
JSSI 2010 – Paris
WHO AM I?

Founder of MoonSols.
Microsoft MVP Enterprise Security

SandMan Framework - Windows hibernation file
Win32/64dd Utility – Windows physical memory acquisition utility
AGENDA

Intro

Who? Why? What?

Analysis
AGENDA

Intro
What?
Why?
Who?
Analysis
WHO?

Forensics Experts
Investigators
Incident Responders
Malware Analysts
..
WHY?

Information goldmine

Pros. Non-volatile memory is not enough. That’s why we need volatile memory.

Cons. Lack of research
AGENDA

Intro
Who?
Why?
What?
Analysis
OVERVIEW

- RAM
- MacBook
- Physical Memory
- Sleep
- Driver
- INFOS
- Memory Dump
MACBOOK

Actual research based on Intel Processor only (x86)

Mac OS X Leopard 10.5
Mac OS X Snow Leopard 10.6
PHYSICAL MEMORY (1/3)

Software based acquisition methods only
With our own kernel extension.
/dev/mem is disabled by default.

```c
void bcopy_phys(addr64_t from,
                addr64_t to,
                int size);
```

is your friend.
PHYSICAL MEMORY (3/3)

OS X hibernation called “Safe Sleep”
/private/var/vm/sleepimage
Compressed (WKDM) memory snapshot
Can be encrypted if secure virtual memory mechanism is used

`sudo pmset-a hibernatemode 1` to disable SVM.
ANALYSIS

No random string searching
ANALYSIS

Get kernel symbols.

Initialize kernel memory manager.

Browse kernel virtual address space.

Collect information.
INFORMATION GOLDMINE
SYMBOLS

Windows compiler stores symbols in externals files called *.PDB

Mac OS X compiler stores symbols inside a section which is part of the executable (`mach_kernel`).
SYMBOLS

__KLD, __LINKEDIT, __PRELINK and __symtab
kernel sections are destroyed as soon as the kernel (mach_kernel) is loaded by removeKernelLinker() function.

__LINKEDIT section contains variable names and offsets.
To read kernel values, we need a quick address translation formula.

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Quick translation Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>i386 Linux</td>
<td>$\text{KPA} = \text{KVA} - 0xC0000000$</td>
</tr>
<tr>
<td>Playstation 3 Linux</td>
<td>$\text{KPA} = \text{KVA} - 0xC0000000000000000$</td>
</tr>
<tr>
<td>Windows</td>
<td>$\text{KPA} = \text{KVA} &amp; 0x1FFFFFF000$</td>
</tr>
<tr>
<td>Mac OS X</td>
<td>$\text{KPA} = \text{KVA}$</td>
</tr>
</tbody>
</table>
SYMBOLS

Works **only** for the mapped executable kernel
(__text and __data sections)

Does **NOT** work for allocated buffers.

.data interesting exported variables:

Memory manager variables
MEMORY

Super interesting variables
_IDLEPDPT
_IDLEPDPPT64
_IDLEPML4
_IDLEPTD
Page Map Level 4 is also initialized on x86 version.
TRANSLATION

<table>
<thead>
<tr>
<th>PML4</th>
<th>Directory Ptr</th>
<th>Directory</th>
<th>Table</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Linear/Virtual Address

- PDE with PS=0
- PTE
- Phys. Addr.

- PML4E
- PDPT

_IdlePML4
SYMBOLS

Kernel symbols + Memory manager initialized

= ?
SYMBOLS

Kernel symbols + Memory manager initialized

= Full kernel address space is browseable (/dev/mem is now /dev/kmem)
MACHINE INFORMATION

The `version` variable contains a string with kernel version and compilation time.

The `machine_info` variable/structure contains:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>major_version</td>
<td>Major OS Version</td>
</tr>
<tr>
<td>minor_version</td>
<td>Minor OS Version</td>
</tr>
<tr>
<td>max_mem</td>
<td>Physical Memory size</td>
</tr>
<tr>
<td>physical_cpu</td>
<td>Number of physical CPU</td>
</tr>
<tr>
<td>logical_cpu</td>
<td>Number of logical CPU</td>
</tr>
</tbody>
</table>
MOUNTED FILE SYSTEM

Link-list called `mountlist`, defined by mount structure.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>f_fstypename</td>
<td>File system type</td>
</tr>
<tr>
<td>f_mntonname</td>
<td>Mounted directory</td>
</tr>
<tr>
<td>f_mntfromname</td>
<td>Mounted file system</td>
</tr>
</tbody>
</table>
**KERNEL EXTENSIONS**

The `kmod` variable is the list-head of every loaded kernel extensions defined by `kmod` structure.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>Base Address</td>
</tr>
<tr>
<td>size</td>
<td>Total Size</td>
</tr>
<tr>
<td>hdr_size</td>
<td>Header Size</td>
</tr>
<tr>
<td>name</td>
<td>Extension Name</td>
</tr>
<tr>
<td>version</td>
<td>Version</td>
</tr>
<tr>
<td>next</td>
<td>Pointer to the next entry</td>
</tr>
</tbody>
</table>
kernproc variable is list-head of every BSD processes defined by proc structure (PID, Parent PID, open files (file descriptors), children, threads, name and a pointer (p_pgrp field) to process group (pgrp structure).)

pgrp structure -> session structure (pg_session field).

session structure -> username (s_login field).
File caching is done via Unified Buffer Cache (UBC) technology under BSD Operating System. UBC is used to unifying the file system and virtual memory caches of file data, thereby providing increased system performance.
SYSCALLS

Syscalls table address is not exported

**Leopard**
As explained by Jesse D’Aguanno at BH US 2008
&sysent = &nsysent + 0x20

**Snow Leopard**
&sysent = &nsysent - ((nsysent) * sizeof(sysent))
SYSCALLS

If an offset from a Syscall entry is not in kernel symbols.
Then, this is not normal 😊

Easy & Fast
INFORMATION GOLDMINE

Information
- Processes
- Kernel Extensions
- Mounted File System
- Machine information
- Syscalls

Threads
 Memory Map
 File Descriptors
 Passwords
 Network Information
 Virtual Nodes

Open Directories
 Open Files
 Content
Web: http://www.moonsols.com
Twitter: msuiche or MoonSols
Mail: msuiche(at)moonsols.com

QUESTIONS?