A little journey inside Windows memory

Damien AUMAITRE
damien(at)security-labs.org
damien.aumaitre(at)sogeti.com

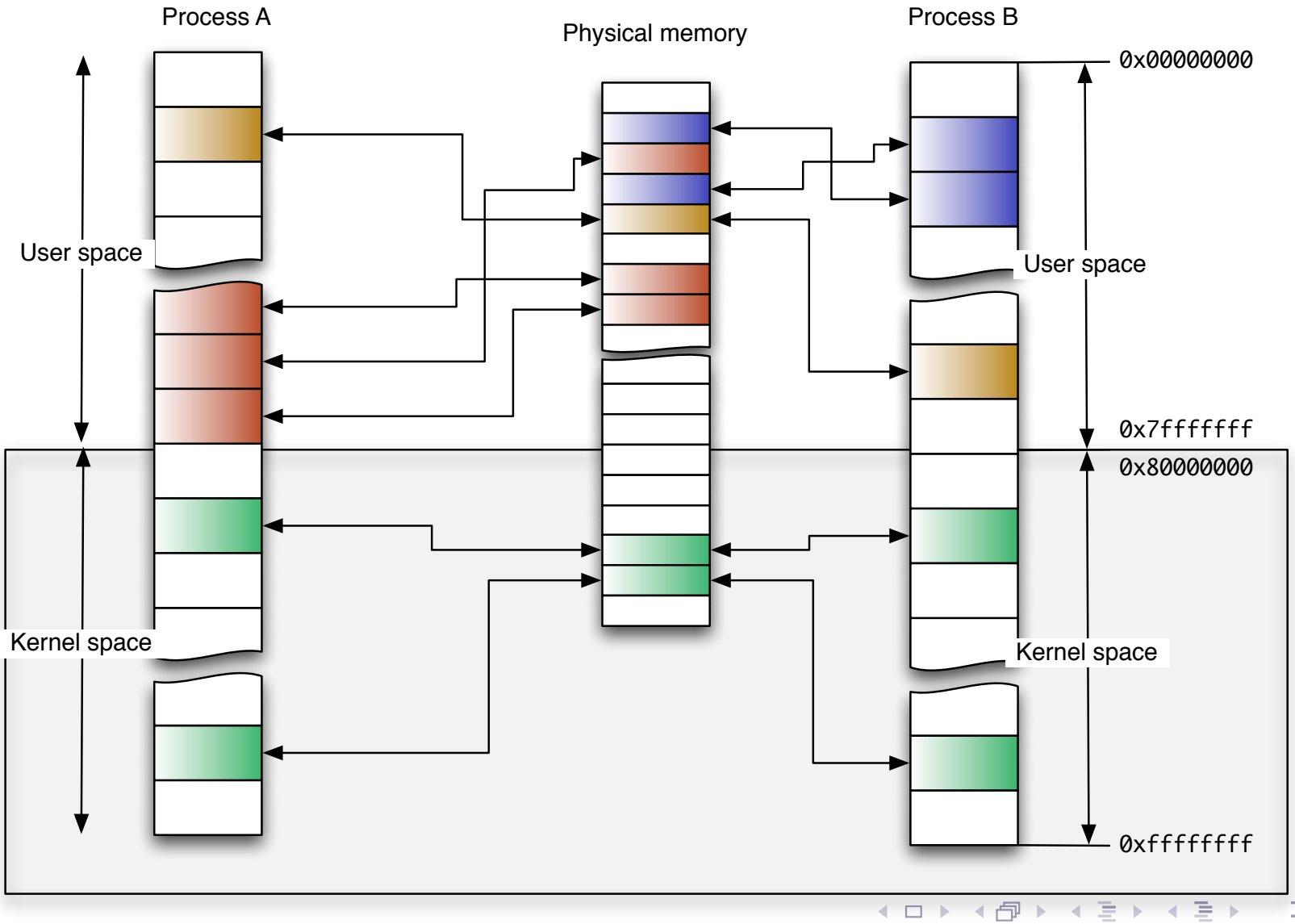
Agenda

1. Memory basics
2. How to access physical memory
3. RWX

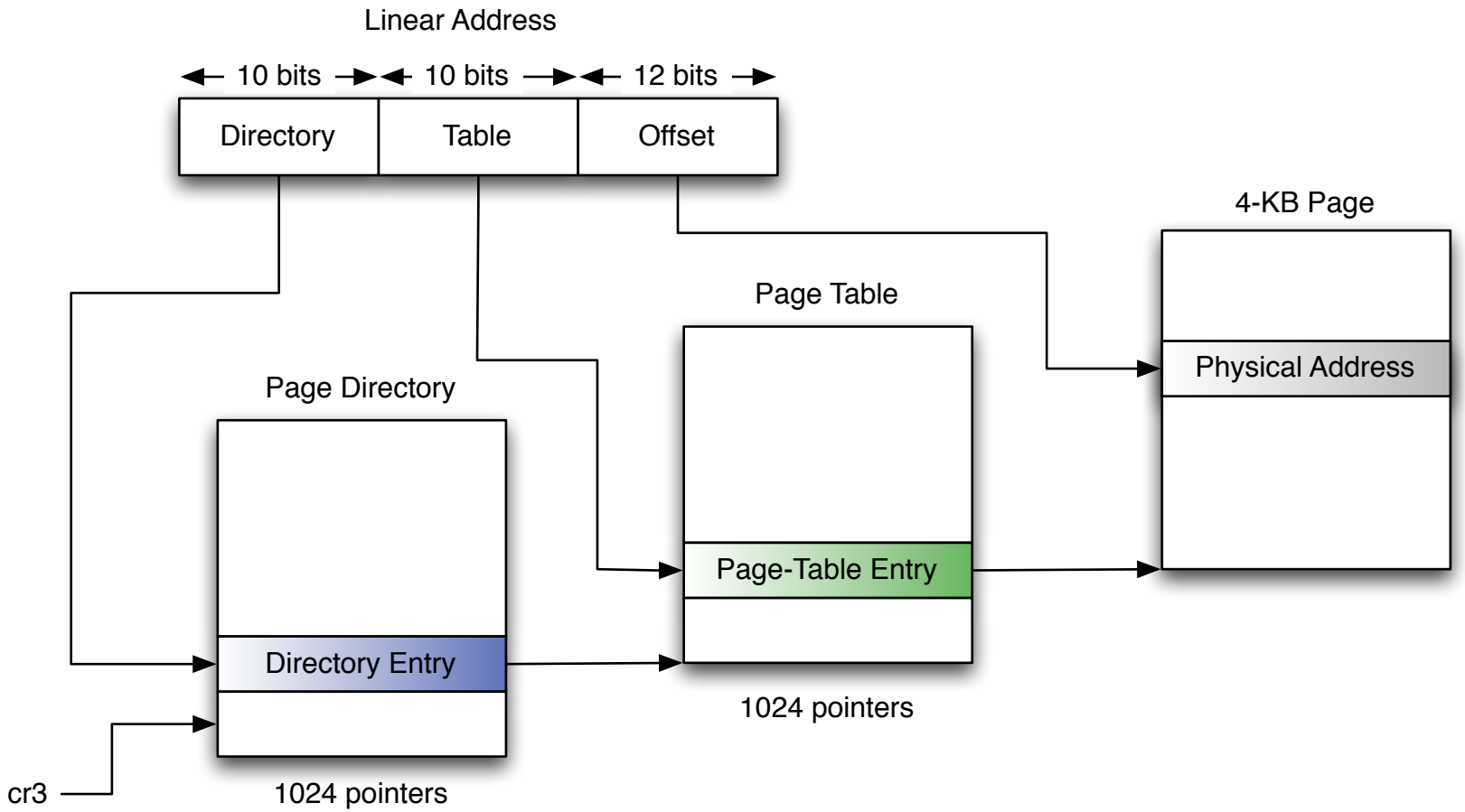
Agenda

1. Memory basics
   - Segmentation / pagination
   - Virtual memory reconstruction
2. How to access physical memory
3. RWX

Virtual address?



Virtual address?



Agenda

1. Memory basics
   - Segmentation / pagination
   - Virtual memory reconstruction
2. How to access physical memory
3. RWX

Why use physical memory?

Pros

- Only interpret data, so independent of OS API.
- Short-circuit security measures implemented by the processor or the kernel.
- Many ways to access physical memory.

Cons

- Need to reconstruct the virtual space since the OS and the processor manipulate virtual addresses.
- Need to understand OS specific structures in order to emulate OS API.

cr3 ?

- Indispensable for address translation.
- Allows you to fully obtain the process virtual space.
- Stored in _KPROCESS structure (field DirectoryTableBase).

```
typedef struct _KPROCESS  // 29 elements, 0x6C bytes (sizeof)
{
/*0x000*/     struct _DISPATCHER_HEADER Header;  // 6 elements, 0x10 bytes (sizeof)
/*0x010*/     struct _LIST_ENTRY ProfileListHead;  // 2 elements, 0x8 bytes (sizeof)
/*0x018*/     ULONG32       DirectoryTableBase[2];  // <---- the CR3
[...]
}KPROCESS, *PKPROCESS;
```

How can we find a _KPROCESS structure?

- Each _KPROCESS begins with a _DISPATCHER_HEADER structure.

```
typedef struct _DISPATCHER_HEADER     // 6 elements, 0x10 bytes (sizeof)
{
/*0x000*/     UINT8          Type;    // <---- interesting for us
/*0x001*/     UINT8          Absolute;
/*0x002*/     UINT8          Size;    // <---- interesting for us
/*0x003*/     UINT8          Inserted;
/*0x004*/     LONG32         SignalState;
/*0x008*/     struct _LIST_ENTRY WaitListHead; // 2 elements, 0x8 bytes (sizeof)
}DISPATCHER_HEADER, *PDISPATCHER_HEADER;
```

- Field Type and Size have fixed values accross OS versions.
- For example, for Windows XP SP2, Type $= 0x3$ and Size $= 0x1b$.

Result

We have a signature to retrieve a _DISPATCHER_HEADER structure inside the physical memory.

How can we find a _KPROCESS structure?

Method proposed by Andreas Schuster

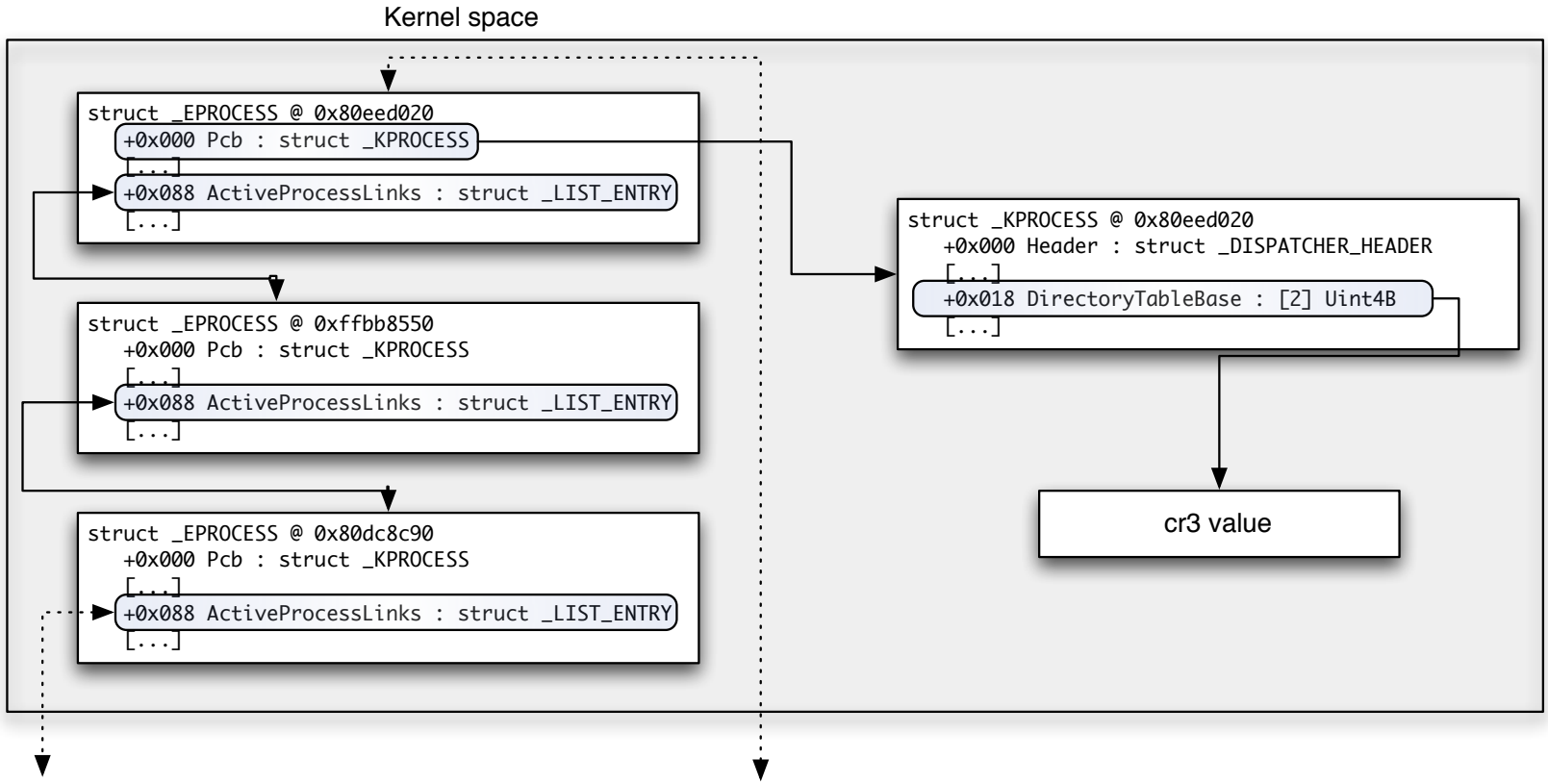
Principle

- Scan physical memory in order to localize a _DISPATCHER_HEADER structure.
- Validate candidate by checking consistency of the structure.

Processes list

- Processes are represented by _EPROCESS structures.
- Which begin with a _KPROCESS structure.
- And belong to a doubly-linked list located in kernelspace.

Virtual spaces reconstruction



Conclusion

Results

- Virtual space translation of all processes.
- Equivalence between physical memory and virtual memory.

Agenda

1. Memory basics
2. How to access physical memory
3. RWX

Agenda

1. Memory basics
2. How to access physical memory
   - Several ways
   - Zoom on FireWire
3. RWX

How to access physical memory?

Several ways:

- DMA (FireWire, PCMCIA, ExpressCard, PCI, etc.)
- VMWare
- Hibernation files with Sandman
- Coldboot attacks
- Memory dumps with forensics tools, etc.

Agenda

1. Memory basics
2. How to access physical memory
   - Several ways
   - Zoom on FireWire
3. RWX

Zoom on FireWire

FireWire ?

- Developed by Apple in the 80's and standardized by IEEE in 1995.
- Allow access to physical memory by using DMA (Direct Memory Access).

Zoom on FireWire

Memory access

- Memory access is configurated by 2 registers of the FireWire controller
- Disabled by default on Windows.
- Except for peripherals that need it.
  - For example mass-storage peripherals, like an iPod

iPod transformation

OHCI 1394 specification

- Each FireWire node has an "identity card"
- Which can be modified...

`libraw1394` library

- Userland library to manipulate FireWire bus.
- With the `raw1394_update_config_rom` function, we can alter our node identity.

iPod transformation

Method

- Dump the ROM of a connected iPod
- Replace the laptop FireWire ROM with the iPod one

iPod transformation

Before

Laptop running Linux.

```
00000000 04 04 0d ef 31 33 39 34 e0 64 a2 32 42 4f c0 00 |....1394.d.2BO..|
00000010 3c c4 44 50 00 03 03 5d 03 42 4f c0 81 00 00 02 |<.DP.....BO.....|
00000020 0c 00 83 c0 00 06 2c 2a 00 00 00 00 00 00 00 00 |......,*........|
00000030 4c 69 6e 75 78 20 2d 20 6f 68 63 69 31 33 39 34 |Linux - ohci1394|
```

iPod transformation

After

An iPod :)

```
00000000 04 04 72 86 31 33 39 34 00 ff a0 12 00 0a 27 00 |..r.1394......'.|
00000010 02 aa 6b a7 00 04 f9 3c 0c 00 83 c0 03 00 0a 27 |..k....<.......'|
00000020 81 00 00 11 d1 00 00 01 00 0e e5 a0 12 00 60 9e |..............`.|
00000030 13 01 04 83 21 00 00 01 3a 00 0a 08 3e 00 4c 10 |............>.L.|
00000040 38 00 60 9e 39 01 04 d8 3b 00 00 00 3c 0a 27 00 |8.`.9.....<.'.|
00000050 54 00 40 00 3d 00 00 03 14 0e 00 00 17 00 00 21 |T.@.............|
00000060 81 00 00 0a 00 08 96 bc 00 00 00 00 00 00 00 00 |................|
00000070 41 70 70 6c 65 20 43 6f 6d 70 75 74 65 72 2c 20 |Apple Computer,.|
00000080 49 6e 63 2e 00 00 00 00 00 04 34 e7 00 00 00 00 |Inc.......4.....|
00000090 00 00 00 00 69 50 6f 64 00 00 00 00 00 00 00 00 |....iPod........|
```

iPod transformation

Conclusion

Since Windows believes an iPod is connected, it authorizes physical memory read/write access.

For more details

Adam Boileau's website : `http://storm.net.nz/projects/16`

Agenda

1. Memory basics
2. How to access physical memory
3. RWX

Agenda

1. Memory basics
2. How to access physical memory
3. RWX
   - Read: gather information
   - Write: everything is authorized
   - eXecute: Welcome to Paradise

Process Explorer 101

Context

Read-only access

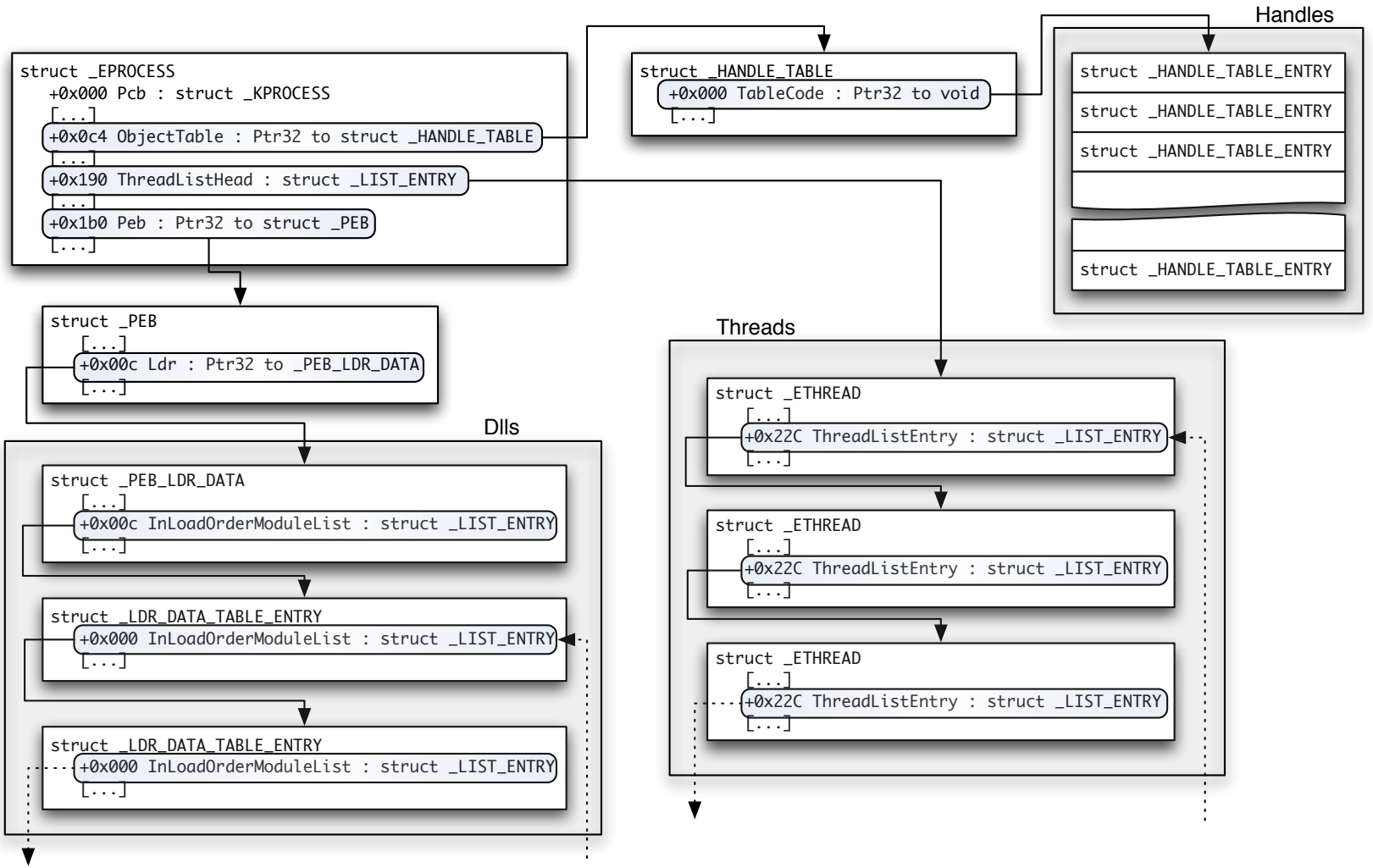
Purpose

Gather and show information relative to each process.

What is needed?

- Processes and threads lists.
- Opened handles, loaded libraries.

Process Explorer 101



Process Explorer 101

DEMO

Regedit 101

Context

Same as Process Explorer 101.

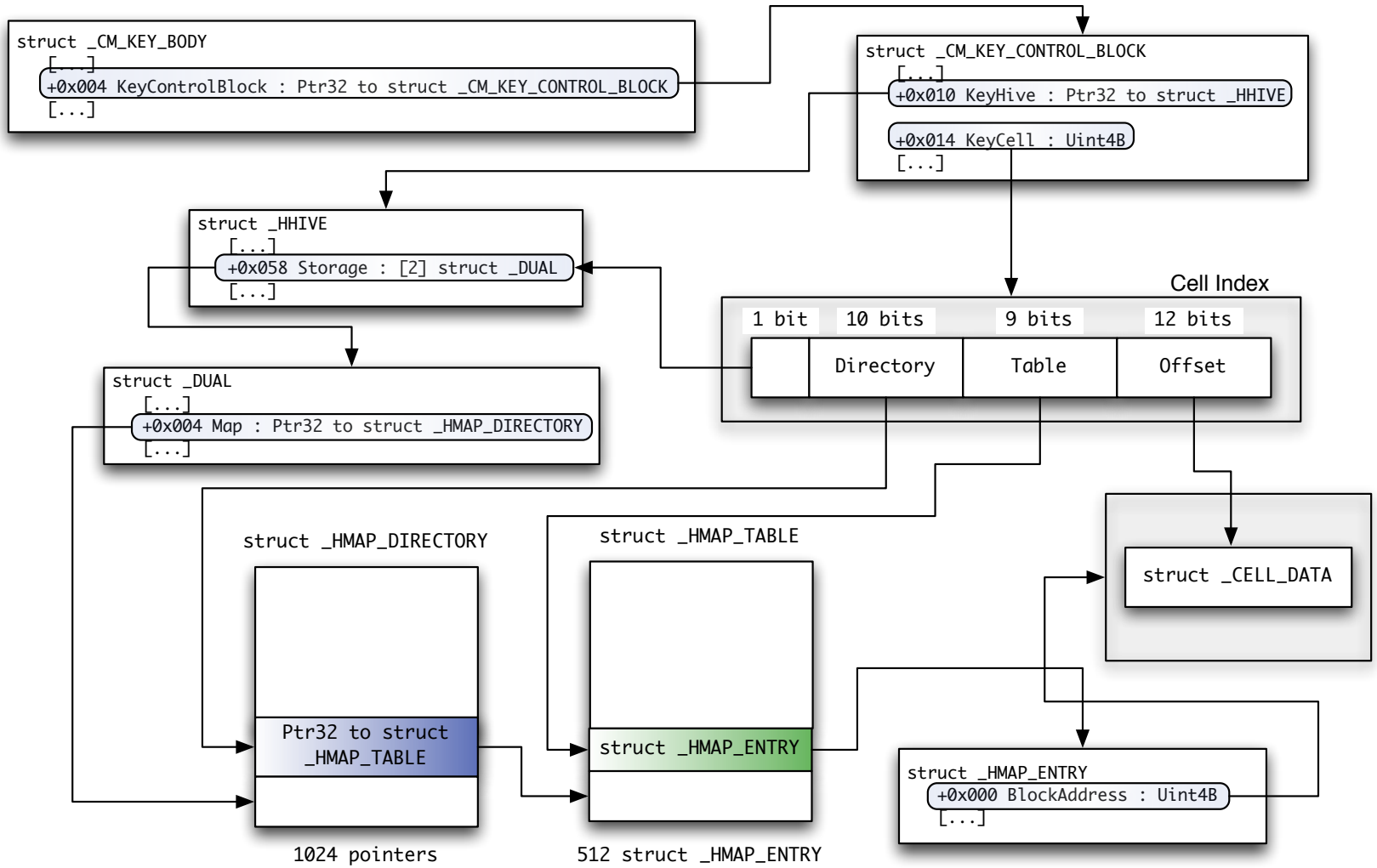
Purpose

Clone regedit.

What is needed?

Hives and registry keys.

Regedit 101


struct _CM_KEY_BODY
[...]
+0x004 KeyControlBlock : Ptr32 to struct _CM_KEY_CONTROL_BLOCK
[...]
struct _CM_KEY_CONTROL_BLOCK
[...]
+0x010 KeyHive : Ptr32 to struct _HHIVE
+0x014 KeyCell : Uint4B
[...]
struct _HHIVE
[...]
+0x058 Storage : [2] struct _DUAL
[...]
Cell Index
1 bit
10 bits
9 bits
12 bits
Directory
Table
Offset
struct _DUAL
[...]
+0x004 Map : Ptr32 to struct _HMAP_DIRECTORY
[...]
struct _HMAP_DIRECTORY
struct _HMAP_TABLE
struct _CELL_DATA
Ptr32 to struct _HMAP_TABLE
struct _HMAP_ENTRY
struct _HMAP_ENTRY
+0x000 BlockAddress : Uint4B
[...]
1024 pointers
512 struct _HMAP_ENTRY

Regedit 101

DEMO

Agenda

1. Memory basics
2. How to access physical memory
3. RWX
   - Read: gather information
   - Write: everything is authorized
   - eXecute: Welcome to Paradise

Login without password?

Context

- Read/write access

Several ways:

- Adam Boileau's `winlockpwn` or...
- 2-bytes patch in registry :)

Login without password?

DEMO

Privilege escalation

- Each process owns a security token.
- Security token belongs to kernel memory.
- But we can access kernel memory :)

Privilege escalation

DEMO

Agenda

1. Memory basics
2. How to access physical memory
3. RWX
   - Read: gather information
   - Write: everything is authorized
   - eXecute: Welcome to Paradise

Arbitrary code execution

Context

- Read/write access
- But no execute access...

A solution

- Functions pointers hooking

Arbitrary code execution

Which pointers?

- `_KUSER_SHARED_DATA` structure
- `SystemCall` field
- Called before each system call

Where to store the payload?

- The `_KUSER_SHARED_DATA` structure occupies only 334 bytes on a 4K-page...

Arbitrary code execution

DEMO

Arbitrary code execution

How it works ?

- Each process belongs to a desktop.
- Only one desktop can interact with a user.
- For an interactive user, 3 desktops `Default`, `Disconnect` et `Winlogon`
- With `CreateProcess`, we can specify the desktop
- We can spawn a cmd in `Winlogon` desktop.
- Thus we have a pre-authentication SYSTEM shell :)

What if DEP is enabled?

- _KUSER_SHARED_DATA is not executable.
- Per process DEP control with _KEXECUTE_OPTIONS.
- Stored inside the _KPROCESS structure.

```
typedef struct _KEXECUTE_OPTIONS // 7 elements, 0x1 bytes (sizeof)
{
/*0x000*/     UINT8          ExecuteDisable : 1;          // 0 BitPosition
/*0x000*/     UINT8          ExecuteEnable : 1;           // 1 BitPosition
/*0x000*/     UINT8          DisableThunkEmulation : 1;   // 2 BitPosition
/*0x000*/     UINT8          Permanent : 1;               // 3 BitPosition
/*0x000*/     UINT8          ExecuteDispatchEnable : 1;   // 4 BitPosition
/*0x000*/     UINT8          ImageDispatchEnable : 1;     // 5 BitPosition
/*0x000*/     UINT8          Spare : 2;                   // 6 BitPosition
}KEXECUTE_OPTIONS, *PKEXECUTE_OPTIONS;
```

Conclusion

iPod 101

- Physical access = root

- We can reconstruct a high-level view of the operating system with only physical memory.
- Many applications: forensics, debug, intrusion.

Questions ?

- Thanks for your attention
- Questions ?

Bibliography

- Adam Boileau: `http://storm.net.nz/projects/16`
- Andreas Schuster: `http://computer.forensikblog.de/en/`
- Sandman: `http://sandman.msuiche.net/`
- Coldboot attacks: `http://citp.princeton.edu/memory/`