Windows Internals (introduction)

John Ombagi

Introduction

Overview

- Basic Operating System Concepts.
- How the Windows OS is implemented.
- Simplified inner working of this OS.
- Not a detail guide.

Expectation & Literature

- Knowledge gained can be applied in:
  - Malware analysis & reverse engineering
  - Exploit development & software development.
- Books Required:
  - Windows Internals 6 (part one and two).
  - The recent Windows Internals 7 (part one).

User mode & Kernel mode

User mode

- Allows access to non-operating system code and data only
- No access to the hardware
- Protect user application form crashing the system

Kernel mode

- Privileged mode for use by the kernel and device drivers only.
- Allows access to all system resources
- Can potentially crash system (BoD?)

Processes

What's a Process?

- It's a set of resources to execute a program.

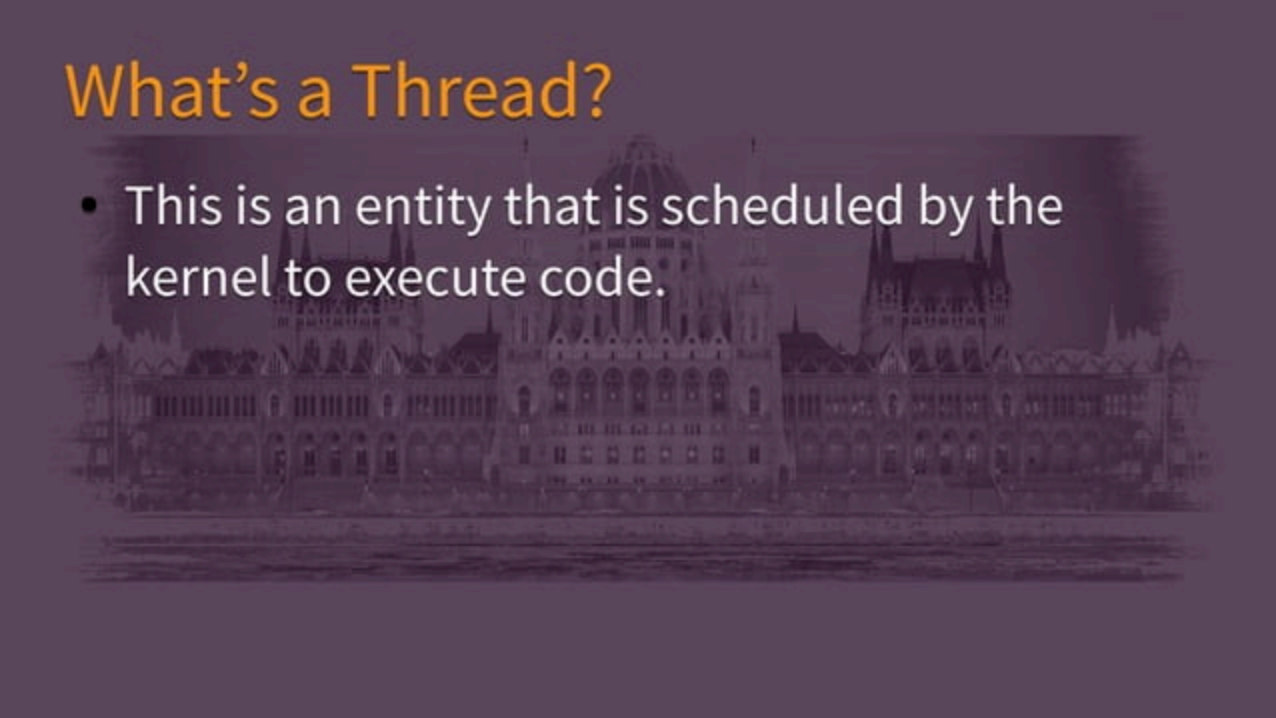
What constitute a process?

- A private Virtual Address space.
- An executable program
- A private table of handles to various kernel objects
- A security context (access token)
- One or more threads that execute code.

Threads

What's a Thread?

- This is an entity that is scheduled by the kernel to execute code.



What constitute a thread?

- The state of CPU registers
- Current access mode
- Two stacks, (userspace and kernel space)
- Thread Local Storage (TLS)
- Optional Security Token
- Optional message queue

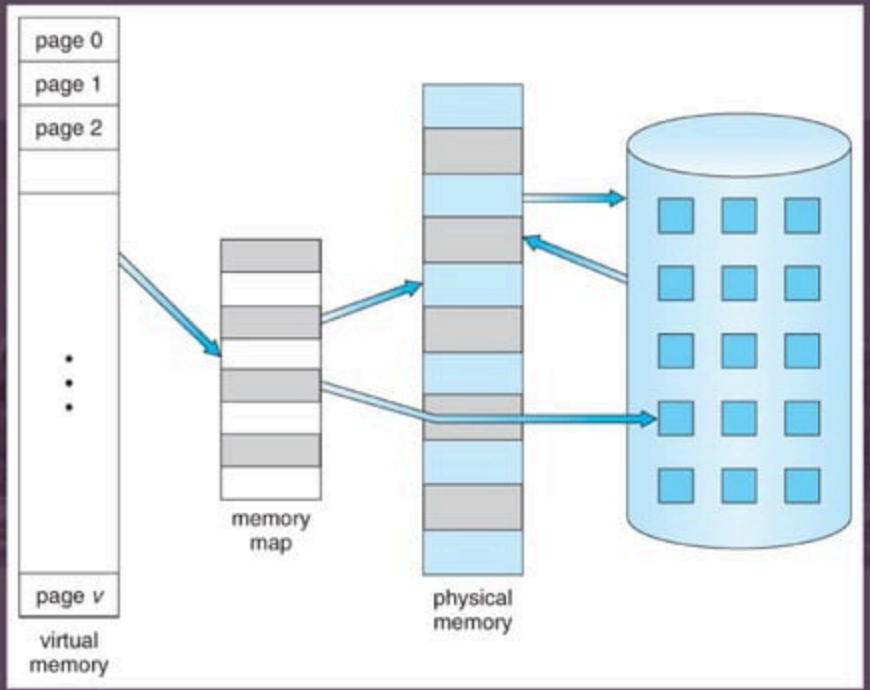
Thread State

- There are several states but the most important ones are as follows:
  - Running
  - Ready
  - Waiting

Virtual Memory

Overview

- Each process "sees" a flat linear memory.
- Virtual memory may be mapped to physical memory, but may also be stored on disk.
- Processes access memory regardless of where it actually resides.
- The memory manager handles mapping of virtual to physical page.


page 0
page 1
page 2
page v
virtual memory
memory map
physical memory

Further Reading

- Windows Internals, Sixth Edition, Part 1 (page 15)
  - Virtual Memory Mapping
  - Virtual Memory Layout

Object and Handles

Details

- Windows is an object based system.
- Objects are run time instance of static structures & reside in system memory space.
- Kernel code can obtain direct pointer to an object. In user mode, code can only obtain a handle to an object.
- Objects are reference counted.

Windows Design Goal

Overview

- Separate address space per process.
- Protected Kernel.
- Pre-emptive multitasking & multithreading.
- Internationalization support using Unicode.
- Security throughout the System.
- Integrated Networking.

Cont.

- Powerful File System (NTFS).
- Run most 16 bit Windows and DOS. applications.
- Run POSIX 1003.1 and OS/2 applications.
- Portable across processors and platforms.
- Be great client as well as server platform.

Core System Files

Executive and Kernel

- On 64 bit Systems, we have Ntoskrnl.exe
- On 32 bit Systems, we have NtKrnlPa.exe
  - Physical Address Extensions (PAE) kernel
- PAE uses paging tables to map the memory greater than 4 GB.

Hal.dll

- Stands for Hardware Abstraction Layer.
- In Windows there are several Hal and the best is chosen during installation.
- It's a layer that insulate the kernel and the drivers from the actual hardware.

Win32k.sys

- It's a Kernel mode component of the Windows subsystem.
- It handles Windowing and GDI (Graphics).

NtDll.dll

- It's the lower layer of the user mode.
- It has System support routines and Native API dispatcher to execute services.
- It provides functions to jump into Kernel mode.
- Provides other simple functions that are similar to the C runtime library.

Kernel32.dll, user32.dll, gdi32.dll, advapi32.dll etc...

- Applications don't call the Kernel or Ntdll.dll directly, NativeAPI is undocumented.
- The way to do it, is by using the official, documented Subsystem DLLs.
- They transition into Ntdll.dll if needed or bypass that for GDI/Windowing calls.

CSRSS.exe

- Client Server Runtime Subsystem
- This is the process that manages the Windows Subsystem.
- It's very important and always running
- Killing this process will result into a blue screen of death.

Symmetric Multiprocessing

SMP

- All CPUs are the same and share main memory and have equal access to peripheral devices.
- No Master or Slave CPUs.
- Basic architecture supports up to 32/64 CPUs.

SMP cont.

- It used bitmask, which was the size of the machine WORD.
- However starting on Windows 7 64 bit and 2008 R2 support up to 256 cores.

  A Processor Group can contain up to 64 Processors, there are four possible processor groups in a 256 core system.

Subsystems

Overview

- It exposes service via subsystem DLLS.
- These subsystem DLLs expose the API of that subsystem in a way that is fitting to that subsystem.
- The Windows subsystem must always be running.
- It’s loaded automatically during boots up

Subsytems cont..

- Information about Subsystems are stored in the following registry key:
  - *HKLM\System\CCS\Control\Session Manger\Subsystems*
- Subsystems DLLs are the ones exposing a particular API.

Cont.

- An image of a certain subsystem calls API functions exposed through the subsystem DLLs.
- Some processes start up before the Windows Subsystem is up.
- Most dispatcher to kernel services are using Windows API "wrappers".

System Processes

Idle Process.

- it has an id a PID of 0 which is not a real process.
- It's just one thread per CPU (core).
- It's a count for Idle time.

System Process

- It's a real process and has a fixed PID of 4.
- It represent stuff going on in the Kernel
- It execute code in system space only.
- Created by *PsCreateSystemThread* kernel API
- The threads don't need a particular process to maintain running (on as System is alive)

Session manger (Smss.exe)

- It runs the *\windows\system32\smss.exe*
- it's the first user mode process created by System (the kernel, part of the boot process)
- Because at this time the windows Subsystem is not loaded yet, it uses the Native API provided by the NtDll.dll.

Session manger contd...

- Creating system environment variables.
- launching the subsystem processes
- It also launches itself in other sessions
- Waits for csrss.exe instances to terminate.
- Waits for subsystem creation request.
- Waits for terminal services session creation requests.

Windows Subsystem (Csrss.exe)

- It runs \windows\system32\csrss.exe
- Provides the user mode side of the Win32 subsystem.
- CSRSS is mainly responsible for Win32 console handling and GUI shutdown.
- It is critical to system operation

Logon Process (Winlogon.exe)

- It runs \windows\system32\winlogon.exe
  - This handles interactive logons and logoffs. If terminated, logs off the user session
  - It's also responsible for capturing of Secure Attention Sequence (SAS) - Ctrl + Alt + Del
  - Presents username / password dialog (through LogonUI.exe).

Local Sec. Auth. Server (Lsass.exe)

- Running the \windows\system32\lsass.exe
- It calls the appropriate authentication packages.
- Upon successful authentication, creates a token representing the user's security profile.
- Returns information to Winlogon.

Service Control Manager (SCM)

- Running \windows\system32\services.exe
  - Responsible for starting, stopping and interacting with service processes.
  - SERVICE : Similar to 'Unix' daemon processes; Not running in Kernel mode in any way; Normal Windows executable that interact with SCM; Can run under "special" accounts (LocalSystem, NetworkService, LocalService).

Local Session Manager (Lsm.exe)

- A helper Introduced in Windows Vista that does some managing of the local session and provides information to smss.exe if needed.
- Running image \windows\system32\lsm.exe
- In windows 8, turned into a service.
- Manages terminal sessions on the local machine

Wow64 (Windows in Windows 64)

Wow64 allows execution of Win32 (32 bit exes) binaries on 64-bit Windows.

- The isWow64Process function can tell whether a process is running under Wow64.
- Filesystem:
  - \windows\system32 contains 64 bit images
  - \windows\syswow64 containd 32 bit images

Wow64-Arch

- Wow64 Restrictions:
  - A 64 bit process cannot load a 32 bit DLLs and vice versa except resource-only DLLs.
- Filesystem Redirection:
  - \Windows\System32 maps to \Windows\Syswow64.
  - \Program Files (x86) & \Program Files

- Registry Redirection:
  - COM components trying to register as 32 bit and 64 bit will crush.
  - 32 bit components are redirected to the Wow64 registry node (Wow6432Node)
    - HKYE_LOCAL_MACHINES\Software
    - HKEY_CLASSES_ROOT
    - HKEY_CURRENT_USER\Software\Classes
  - New flags for Registry APIs allow access to the 64 bit or 32 bit nodes
    - KEY_WOW64_64KEY & KEY_WOW64_32KEY

End of Part One.

John Ombagi
jayombagi@gmail.com