Windows Internals

Module 4: System Architecture (Part 2)

Pavel Yosifovich

CTO, CodeValue

pavely@codevalue.net

http://blogs.Microsoft.co.il/blogs/pavely

pluralsight hardcore developer training

Contents

Core system files

- **Ntoskrnl.exe**
  - Executive and kernel on 64 bit systems
- **NtKrnlPa.exe**
  - Executive and kernel on 32 bit systems
- **Hal.dll**
  - Hardware Abstraction Layer
- **Win32k.sys**
  - Kernel component of the Windows subsystem
  - Handles windowing and GDI
- **NtDll.dll**
  - System support routines and Native API dispatcher to executive services
- **Kernel32.dll, user32.dll, gdi32.dll, advapi32.dll**
  - Core Windows subsystem DLLs
- **CSRSS.exe ("Client Server Runtime SubSystem")**
  - The Windows subsystem process

Demo

Core system files

Symmetric multiprocessing

- **SMP**
  - All CPUs are the same and share main memory and have equal access to peripheral devices (no master/slave)
- **Basic architecture supports up to 32/64 CPUs**
  - Windows 7 64 bit & 2008 R2 support up to 256 cores
  - Uses a new concept of a “processor group”
- **Actual number of CPUs determined by licensing and product type**
  - Multiple cores do not count towards this limit

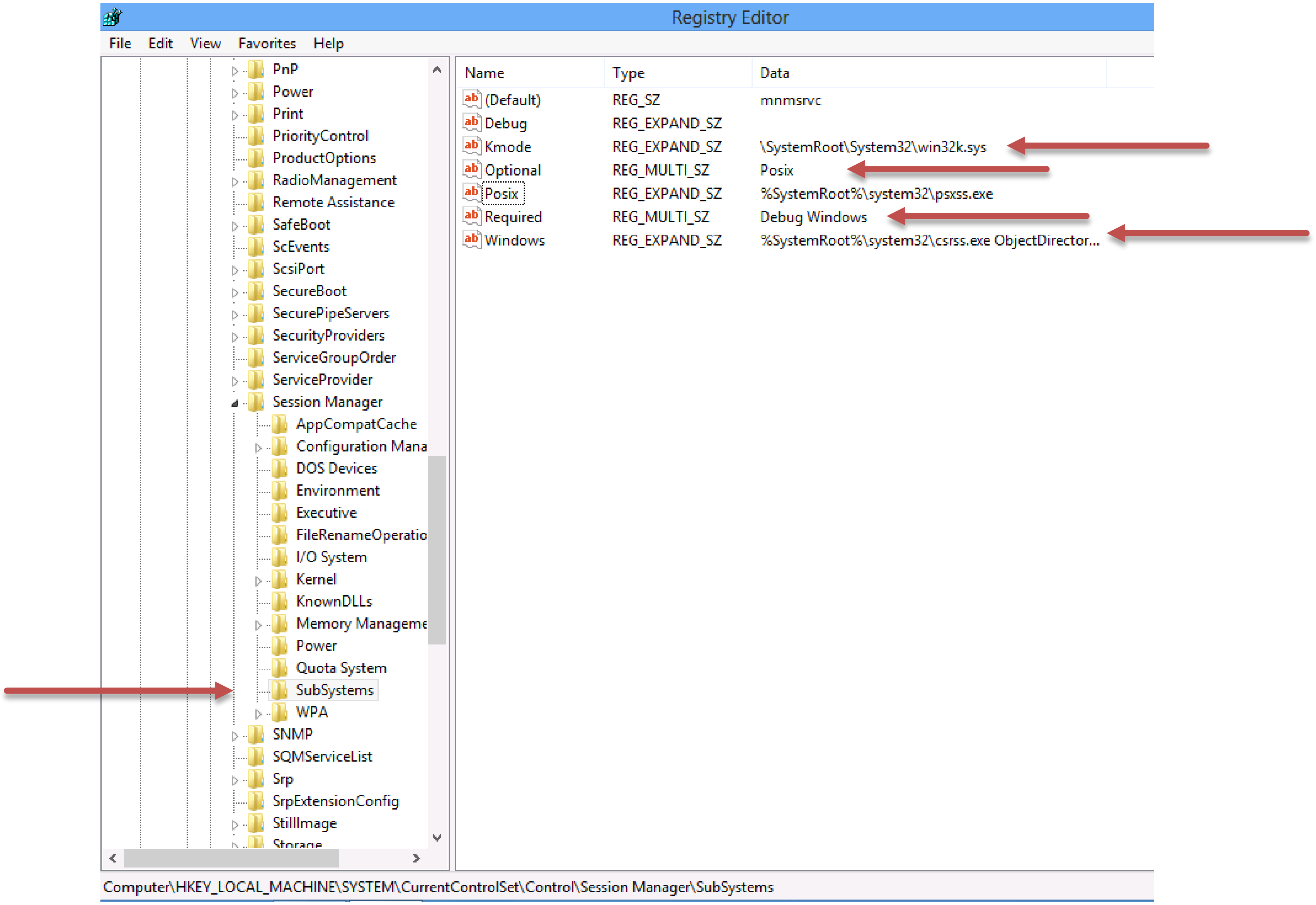
Demo

SMP

Subsystems

- **A subsystem is a special view of the OS**
  - Exposes services via subsystem DLLs
- **Original NT shipped with Win32, OS/2 and POSIX 1003.1 (POSIX-1)**
  - Windows XP dropped support for OS/2
  - An enhanced POSIX version is available with the "Services for UNIX" product
- **The Windows subsystem must always be running**
  - Owner of keyboard, mouse and display
- **Some API functions use the Advanced Local Procedure Call (ALPC) to notify CSRSS of relevant events**
- **Other subsystems configured to load on demand**
- **Subsystem information stored in registry: HKLM\System\CCS\Control\Session Manager\Subsystems**

Subsystems in the registry



Registry Editor

File Edit View Favorites Help

PnP
Power
Print
PriorityControl
ProductOptions
RadioManagement
Remote Assistance
SafeBoot
ScEvents
ScsiPort
SecureBoot
SecurePipeServers
SecurityProviders
ServiceGroupOrder
ServiceProvider
Session Manager
AppCompatCache
Configuration Mana
DOS Devices
Environment
Executive
FileRenameOperatio
I/O System
Kernel
KnownDLLs
Memory Manageme
Power
Quota System
SubSystems
WPA
SNMP
SQMServiceList
Srp
SrpExtensionConfig
StillImage
Storage

Name	Type	Data
(Default)	REG_SZ	mnmsrvc
Debug	REG_EXPAND_SZ	
Kmode	REG_EXPAND_SZ	\SystemRoot\System32\win32k.sys
Optional	REG_MULTI_SZ	Posix
Posix	REG_EXPAND_SZ	%SystemRoot%\system32\psxss.exe
Required	REG_MULTI_SZ	Debug Windows
Windows	REG_EXPAND_SZ	%SystemRoot%\system32\csrss.exe ObjectDirector...

Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\SubSystems

Subsystem DLLs

- **Every image belongs to exactly one subsystem**
  - Value stored in image PE header
    - Can view with Dependency Walker (**depends.exe**)
  - Allows the Windows Loader to make correct decisions
- **An image of a certain subsystem calls API functions exposed through the subsystem DLLs**
  - E.g. kernel32.dll, user32.dll, etc. for the Windows subsystem
- **Some images belong to no subsystem**
  - "Native" images
  - Which API functions do they call?

The Native API

- **Implemented by NTDLL.DLL**
  - Used by subsystem DLLs and “native” images
  - Undocumented interface
  - Lowest layer of user mode code
- **Contains**
  - Various support functions
  - Dispatcher to kernel services
    - Most of them accessible using Windows API “wrappers”

Demo

Subsystem DLLs and NTDLL

System Processes

- Idle process
- System process
- Session Manager (Smss.Exe)
- Windows subsystem (Csrss.Exe)
- Logon process (Winlogon.Exe)
- Service control manager (SCM) (Services.Exe)
- Local security authentication server (Lsass.Exe)
- Local session manager (Lsm.exe)

Idle Process

- Always has a PID of 0
- Not a real process (does not run any executable image)
- One thread per CPU (core)
- Accounts for idle time

System Process

- **Has a fixed PID (4)**
- **Represents the kernel address space and resources**
- **Hosts system threads**
  - Threads created by the kernel and device drivers
  - Execute code in system space only
  - Created using the **PsCreateSystemThread** kernel API (documented in the WDK)
  - Allocate memory from the system pools

Demo

Idle and System processes

Session Manager

- **Running the image \windows\system32\smss.exe**
- **The first user mode process created by the system**
- **Main tasks**
  - Creating system environment variables
  - Launches the subsystem processes (normally just csrss.exe)
  - Launches itself in other sessions
    - That instance loads WINLOGON and CSRSS in that session
    - Then terminates
- **Finally**
  - Waits forever for csrss.exe instances to terminate
    - If any of them dies, crashes the system
  - Waits for subsystem creation requests
  - Waits for terminal services session creation requests

Winlogon

- **Running the image \windows\system32\winlogon.exe**
- **Handles interactive logons and logoffs**
- **If terminated, logs off the user session**
- **Notified of a user request by the Secure Attention Sequence (SAS), typically Ctrl+Alt+Del**
- **Authenticates the user by presenting a username / password dialog (through LogonUI.exe)**
  - Can be replaced
- **Sends captured username and password to LSASS**
  - If successfully authenticated, initiates the user's session

LSASS

- Running the image \windows\system32\Lsass.exe
- Calls the appropriate authentication package
- Upon successful authentication, creates a token representing the user’s security profile
- Returns information to Winlogon

Service Control Manager (SCM)

- **Running the image \windows\system32\services.exe**
- **Responsible for starting, stopping and interacting with service processes**
- **Services**
  - Similar to UNIX “daemon processes”
  - Normal Windows executables, that interact with the SCM
  - Can be started automatically when the system starts up without an interactive logon
  - Can run under “special” accounts
    - LocalSystem, NetworkService, LocalService

Local Session Manager

- **Introduced in Windows Vita**
  - Running the image **\windows\system32\lsm.exe**
- **In windows 8, turned into a service**
  - Implemented in **\windows\system32\lsm.dll**
  - Hosted in a standard svchost.exe
- **Manages terminal sessions on the local machine**
- **Communicates requests to SMSS**

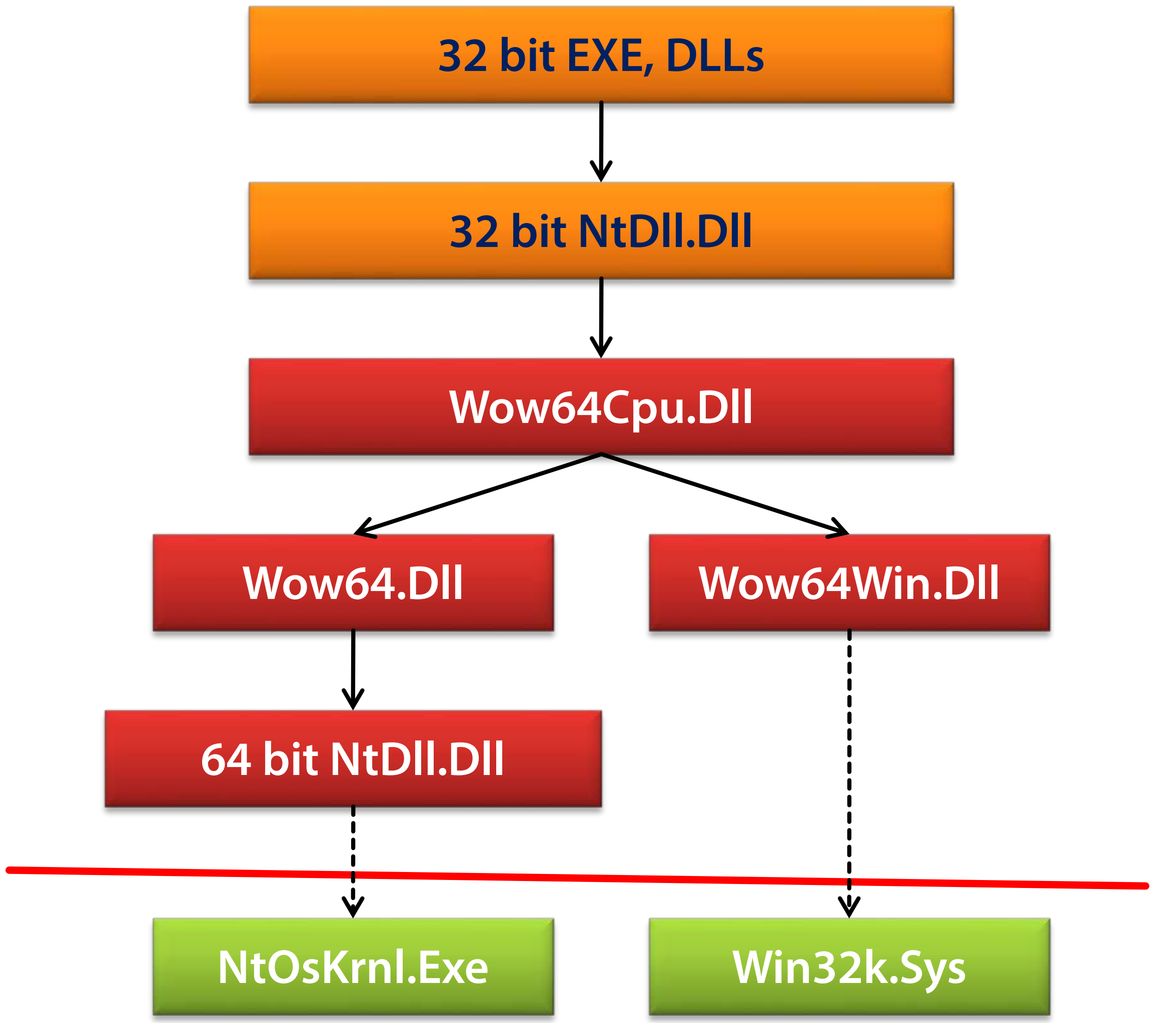
Demo

Services

Wow64

- **Allows execution of Win32 binaries on 64-bit Windows**
  - Wow64 intercepts system calls from the 32-bit application
    - Converts 32-bit data structure into 64-bit aligned structures
    - Issues the native 64-bit system call
    - Returns any data from the 64-bit system call
- **The IsWow64Process function can tell whether a process is running under Wow64**
- **Address space is 2GB or 4GB (if image is linked with the LARGEADDRESSAWARE flag)**
- **Device drivers must be native 64 bit**
- **File system**
  - \windows\system32 contains 64 bit images
  - \windows\syswow64 contains 32 bit images

Wow64 Architecture



Wow64 Restrictions

- **A 64 bit process cannot load a 32 bit DLL and vice versa**
  - Except resource-only DLLs, which can be loaded cross-architecture
- **Some APIs are not supported by Wow64 processes**
  - **E.g. ReadFileScatter**, **WriteFileGather**, AWE functions

File System Redirection

- **System directories names have not changed in 64 bit Windows (e.g. \Windows\System32 contains native 64 bit images)**
- **32 bit applications must use their own directories**
  - \Windows\System32 maps to \Windows\Syswow64
  - 32 bit apps installed in \Program Files (x86)
  - 64 bit apps installed in \Program Files
- **Some directories are not redirected**

Registry Redirection

- Components trying to register as 32 bit and 64 bit will clash
- 32 bit components are redirected to the Wow64 registry node (Wow6432Node)
  - HKEY_LOCAL_MACHINE\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** – open a 64 bit key
  - **KEY_WOW64_32KEY** – open a 32 bit key

Demo

Wow64

Summary

- A Process executes under a specific subsystem
- The primary subsystem is the Windows subsystem
- NTDLL is the gateway to kernel mode
- Wow64 allows running 32 bit processes on 64 bit systems transparently