Windows Kernel Internals
Win32K.sys

David B. Probert, Ph.D.
Windows Kernel Development
Microsoft Corporation

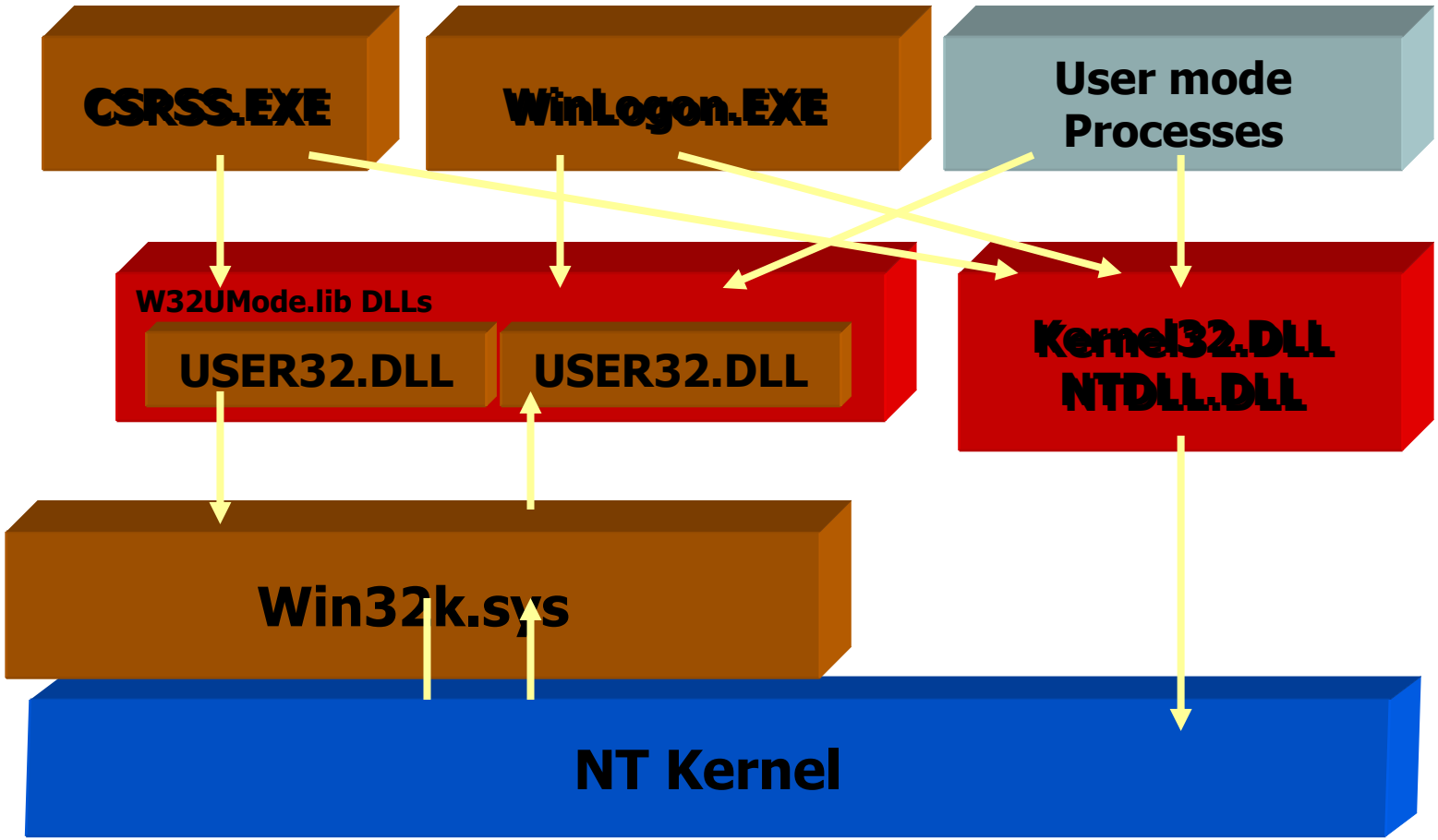
Topics

- Background
- Entry Points Architecture
- GUI Threads
- Initialization & Shutdown
- Memory Manager & Win32k
- User Mode callbacks & LPC

What is win32k?

- Kernel side of the Windows Sub-System
- Graphical User Interface (GUI) infrastructure of the operating system.
- Includes kernel side of:
  - Window Manager (USER)
  - Graphic Device Interfacce (GDI)
  - Dx thunks to dxg.sys (DirectX)

Win32k & the OS



W32umode.lib DLLs

- User32.dll, Imm32.dll – Window Manager
- Gdi32.dll, MsImg32.dll - GDI
- d3d8thk.dll – DirectX thunks
- CSRSS's Winsrv.dll – Console, TS, HardError.
- Twsrc_32.dll – print driver
- F3ahvoas – keyboard drivers

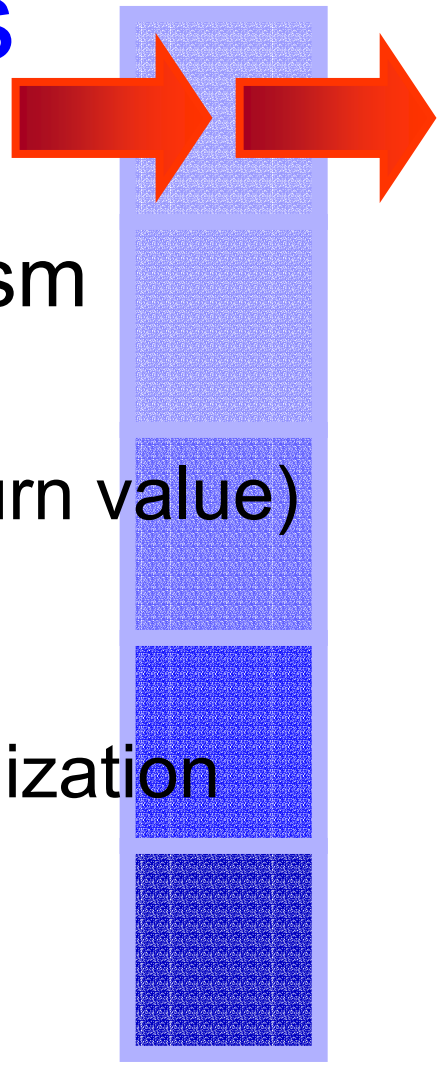
Win32k Entry Points

User Mode Stubs

- About 600 entry points.
- Nt¥windows¥core¥kmode¥services.tab
- Gensrv.exe generates usrstubs.c, used to build w32umode.lib.
- Stubs
  - Index
  - Syscall (int 2e in x86)
  - Params in the stack

Win32k Entry Points

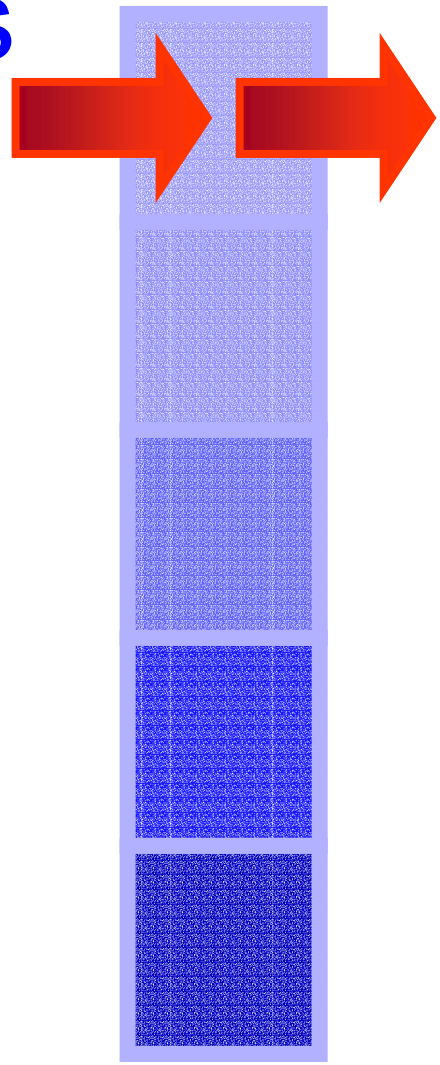
Kernel Service Table



- Gensrv.exe generates Systable.asm
- Builds three tables in win32k.sys
  - W32pServiceTable (function & Return value)
  - W32pArgument
  - TableProvided to NT kernel via KeAddSystemServiceTable on initialization
- nt¥base¥ntos¥ke¥i386¥trap.asm

Win32k Entry Points

User Mode Memory



- Must not blue screen!
- Probing – range/alignment check
- Capturing
- Try-excepts
  - Must have probed first
  - Small blocks

What is a GUI thread?

- Thread are non-GUI when created
- Converted on first call to win32k.sys
  - Bigger Stack
  - Win32k.sys notified of creation and destruction
  - Converts process to GUI
- How to recognize a GUI thread:
  - KTHREAD->Win32Thread pointer.
  - In user mode TEB->Win32ThreadInfo
  - Programmatically – IsGuiThread(fConvert) – Whistler only

Convertion to GUI Thread

- nt¥base¥ntos¥ke¥i386¥trap.asm
- PsConvertToGuiThread
  - MmCreateKernelStack & KeSwitchKernelStack
  - KTHREAD->ServiceTable initialized to ntkrnlmp!KeServiceDescriptorTable, replaced with ntkrnlmp!KeServiceDescriptorTableShadow
  - Call PspW32ProcessCallout
  - Call PspW32ThreadCallout

GUI Initialization

- SMSS.EXE – Identifies Session components
  - HKLM¥SYSTEM¥CurrentControlSet¥Control¥Session Manager¥subsystem
    - kmode (win32k.sys)
    - Windows (CSRSS.EXE)
  - “Initial command” (Hardcoded to Winlogon.exe)
- Csrss and Winlogon loaded via RtlCreateUserProcess
- Win32k loaded via MmLoadSystemImage (NtSetSystemInformation)

Initialization

Win32k!DriverEntry

- KeAddSystemServiceTable
- PsEstablishWin32Callouts
- MmPageEntireDriver
- InitializeGre
- Win32UserInitialize
- Returns Win32KDriverUnload

Intialization

PsEstablishWin32Callouts

- W32pProcessCallout
- W32pThreadCallout
- UserGlobalAtomTableCallout
- UserPowerEventCallout
- UserPowerStateCallout
- UserJobCallout
- NtGdiFlushUserBatch

Initialization

Winsrv!UserServerDllInitialization

- CSRSS is expect to create first GUI thread.
- Win32k!NtUserInitialize
  - InitVideo
  - Font initialization
- Notification thread. – NLS (registry cache), Power, Media (CD), Net changes (TS)
- ApiPort thread (LPC)

Initialization

Winlogon

- Creates IO Windowstation (Winsta0)
  - Desktop thread
  - Raw Input Thread (RIT).
- RegisterLogonProcess.
- Creates LogonDesktop, which causes USER to create disconnect desktop.
- Creates default desktop.
- Launches Services.exe and svchost.exe -- more windowstations.

Shutdown

- ExitWindows
  - Win32k notifies Winlogon which makes actual ExitWindows call
  - Csrss notifies and shuts down processes.
- Non TS – Winlogon calls NtShutdownSystem
- TS needs to unload win32k.sys and exit CSRSS and WINLOGON
- InitiateWin32kCleanup
- TerminateProcess CSRSS
- Win32KDriverUnload

MM & Win32k

Paging

- Needed to support multiple sessions
- MmPageEntireDriver
  - Treated like a user mode process – code shared, data per instance – the whole thing is pageable.
- Pool
  - Session pool – pageable
  - Non paged – required for kevent, ktimers, Ob objects, etc

MM & Win32k

Address Spaces

A000.0000 -> A080.0000: 8MB of win32k.sys and other images

A080.0000 -> A0c0.0000: 4MB of private Mm per-session data and working set list information

A0c0.0000 -> A200.0000: 20MB of session views for win32k desktop heaps and any other mapped files

A200.0000 -> A300.0000: 16MB of session paged pool to support win32k pool allocations

A300.000 -> A400.0000: 16MB of systemwide-global mapped views

MM & Win32k

Configurable Address Spaces

- HKLM¥CurrentControlSet¥Control¥Session Manager¥Memory Management¥SessionViewSize = DWORD n

  where n is the size of the SESSION view space in MB (default is 20mb if no key).

- HKLM¥CurrentControlSet¥Control¥Session Manager¥Memory Management¥SessionPoolSize = DWORD n

  where n is the size of the SESSION pool space in MB. (default is 16mb if no key).

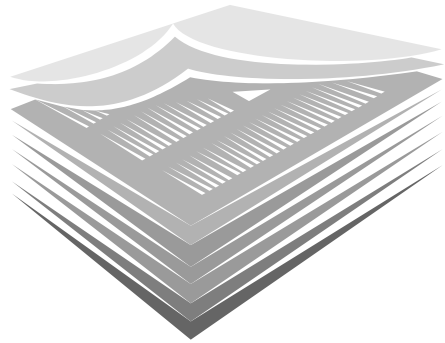
MM & win32k

Views

- Mapping kernel mode memory into user address space.
  - Read only (usually).
- Used to avoid kernel transitions
- MmMapViewOfSection & MmUnmapViewOfSection

MM & Win32k

Stacks



- Bigger than regular threads.
- Kernel stacks are fixed size – stack fault == blue screen.
- Upon entering the kernel (or before calling back to user mode), MM guarantees 12K of stack are resident.
- Can grow up to 64K (possibly will be changed to 32K)
- In win64 stack and backstore (for registers). 88K and 56K.

User mode callbacks

- KeUserModeCallback:
  - api index,
  - pointer to params & size
  - pointer to pointer to ret data and pointer to size of ret data (user mode memory)
- NtCurrentPeb()->KernelCallbackTable loaded at initialization time.

LPCing

- CSRSS ApiPort.
- LpcRequestPort (CSRSS context)
- LpcRequestWaitReplyPort (Any other context)
- Must release critical sections!

Discussion