

Windows Operating System Family - Concepts & Tools

Roadmap for This Lecture

- High-level Overview on Windows Concepts
 - Design goals of NT
 - Processes, Threads
 - Virtual Memory, Protection
 - Objects and Handles
 - Security
- Key monitoring tools
 - Windows is thoroughly instrumented
- Extra resources at <http://www.sysinternals.com>

Requirements and Design Goals for the original Windows NT project

- Provide a true 32-bit, preemptive, reentrant, virtual memory operating system
- Run on multiple hardware architectures and platforms
- Run and scale well on symmetric multiprocessing systems (in addition to uni-processor machines)
- Be a great distributed computing platform (Client & Server)
- Run most existing 16-bit MS-DOS and Microsoft Windows 3.1 applications
- Meet government requirements for POSIX 1003.1 compliance
- Meet government and industry requirements for operating system security
- Be easily adaptable to the global market by supporting Unicode

In this course, the term Windows refers to Windows 2000, XP, Server 2003, Vista and Win 7.

Goals (contd.)

- **Extensibility**
 - Code must be able to grow and change as market requirements change.
- **Portability**
 - The system must be able to run on multiple hardware architectures and must be able to move with relative ease to new ones as market demands dictate.
- **Reliability and Robustness**
 - Protection against internal malfunction and external tampering.
 - Applications should not be able to harm the OS or other running applications.
- **Compatibility**
 - User interface and APIs should be compatible with older versions of Windows as well as older operating systems such as MS-DOS.
 - It should also interoperate well with UNIX, OS/2, and NetWare.
- **Performance**
 - Within the constraints of the other design goals, the system should be as fast and responsive as possible on each hardware platform.

Portability

- HAL (Hardware Abstraction Layer):
 - support for x86 (initial), MIPS (initial), Alpha AXP, PowerPC (NT 3.51), Itanium (Windows XP/2003)
 - Machine-specific functions located in HAL
- Layered design:
 - architecture-specific functions located in kernel
- Windows kernel components are primarily written in C:
 - OS executive, utilities, drivers
 - UI and graphics subsystem - written in C++
 - HW-specific/performance-sensitive parts:
written in assembly lang: int trap handler, context switching

Windows API & Subsystems

- Windows API (application programming interface):
 - Common programming interface to Windows NT/2000/XP/2003, Windows 95/98/ME and Windows CE
 - OS implement (different) subsets of the API
 - MSDN: <http://msdn.microsoft.com>
- Windows supports multiple subsystems (APIs):
 - Windows (primary), POSIX, OS/2
 - User space application access OS functionality via subsystems
- Subsystems define APIs, processes, and file system semantics
 - OS/2 used to be primary subsystem for Windows NT

64-bit vs. 32-bit Windows APIs

- Pointers and types derived from pointer, e.g. handles, are 64-bit long
 - A few others go 64, e.g. WPARAM, LPARAM, LRESULT, SIZE_T
 - Rest are the same, e.g., 32-bit INT, DWORD, LONG
- Only five replacement APIs!
 - Four for Window/Class Data
 - Replaced by Polymorphic (`_ptr`) versions
 - Updated constants used by these APIs
 - One (`_ptr`) version for flat scroll bars properties

Win32 and Win64 are consistently named the Windows API

API	Data Model	<code>int</code>	<code>long</code>	pointer
Win32	ILP32	32	32	32
Win64	LLP64	32	32	64
UNIXes	LP64	32	64	64

Services, Functions, and Routines

- Windows API functions:
 - Documented, callable subroutines
 - *CreateProcess*, *CreateFile*, *GetMessage*
- Windows system services:
 - Undocumented functions, callable from user space
 - *NtCreateProcess* is used by Windows *CreateProcess* and POSIX *fork()* as an internal service
- Windows internal routines:
 - Subroutines inside the Windows executive, kernel, or HAL
 - Callable from kernel mode only (device driver, NT OS components)
 - *ExAllocatePool* allocates memory on Windows system heap

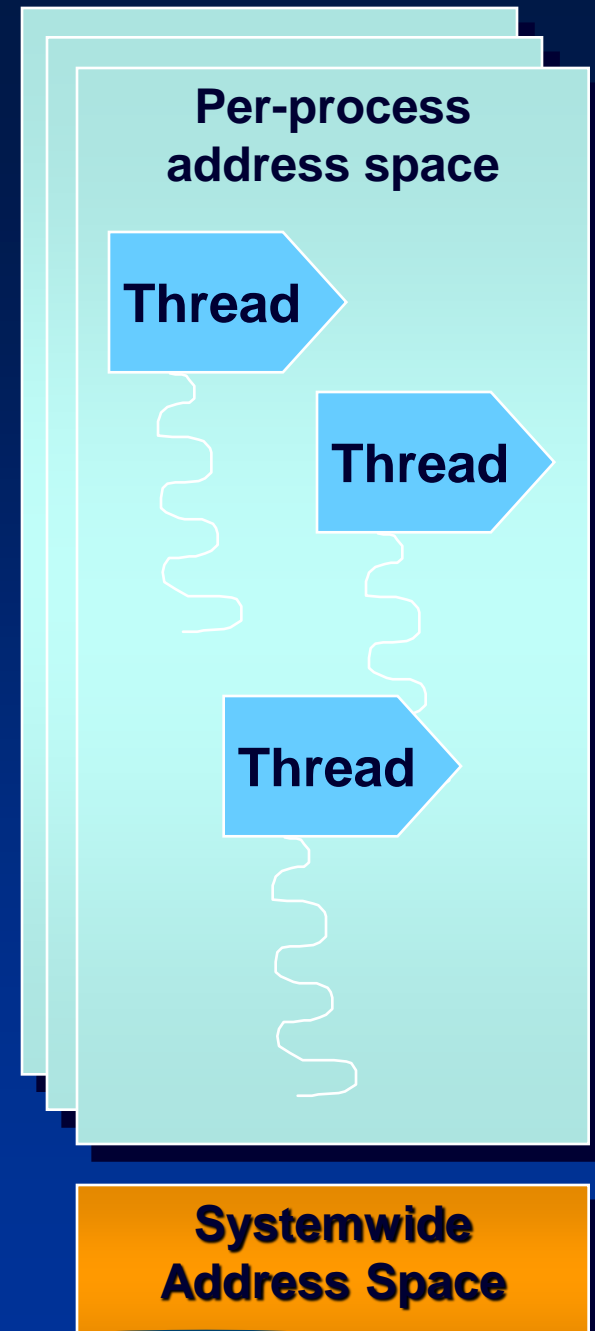
Services, Functions, and Routines (contd.)

- Windows services:
 - Processes which are started by the Service Control Manager
 - Example: The *Schedule* service supports the AT command
- DLL (dynamic link library)
 - Subroutines in binary format contained in dynamically loadable files
 - Examples: MSVCRT.DLL – MS Visual C++ run-time library
KERNEL32.DLL – one of the Windows API libraries

Processes & Threads

• What is a process?

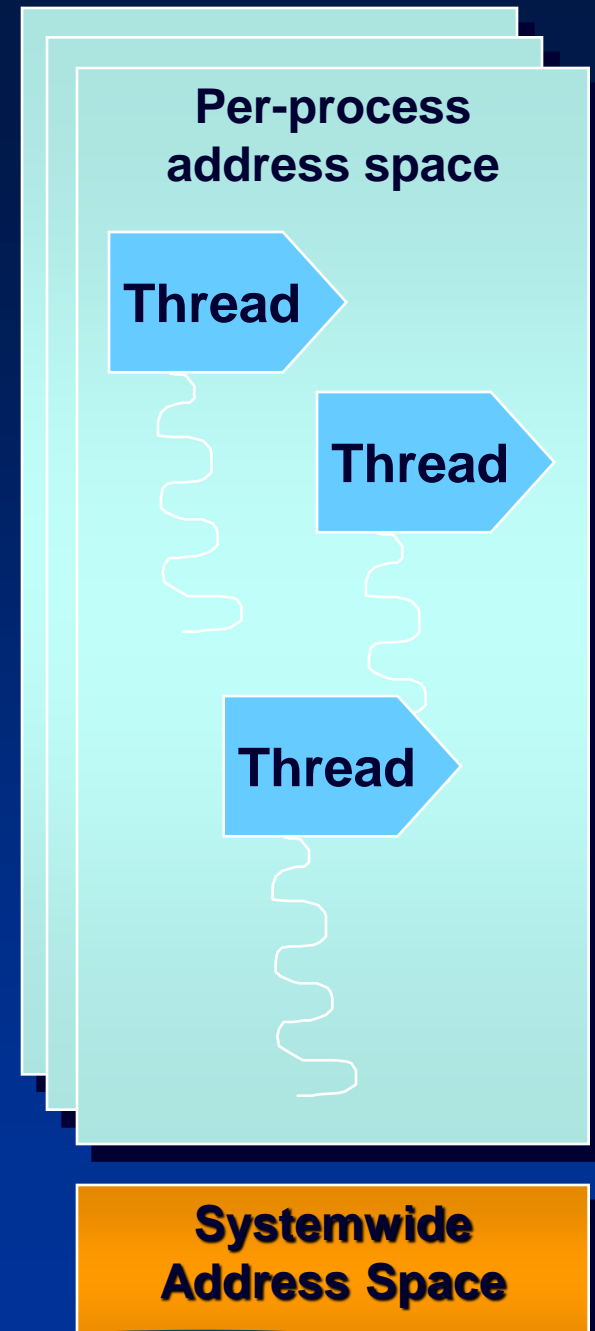
- Represents an instance of a running program
 - you create a process to run a program
 - starting an application creates a process
- Contains the program code and its current activity
- Made up of multiple threads of execution that execute instructions concurrently



Processes & Threads

A process consists of:

- An **image** of the executable machine code associated with a program.
- Memory (typically some region of virtual memory):
 - executable code,
 - process-specific data (input and output)
 - a call stack
 - a heap to hold intermediate computation data
- Operating system descriptors of resources:
 - file descriptors (Unix) or handles (Windows)
 - data sources and sinks (stream buffers)
- Security attributes:
 - process
 - process' set of permissions (allowable operations)
- Processor state (context):
 - the content of registers
 - physical memory addressing, etc.



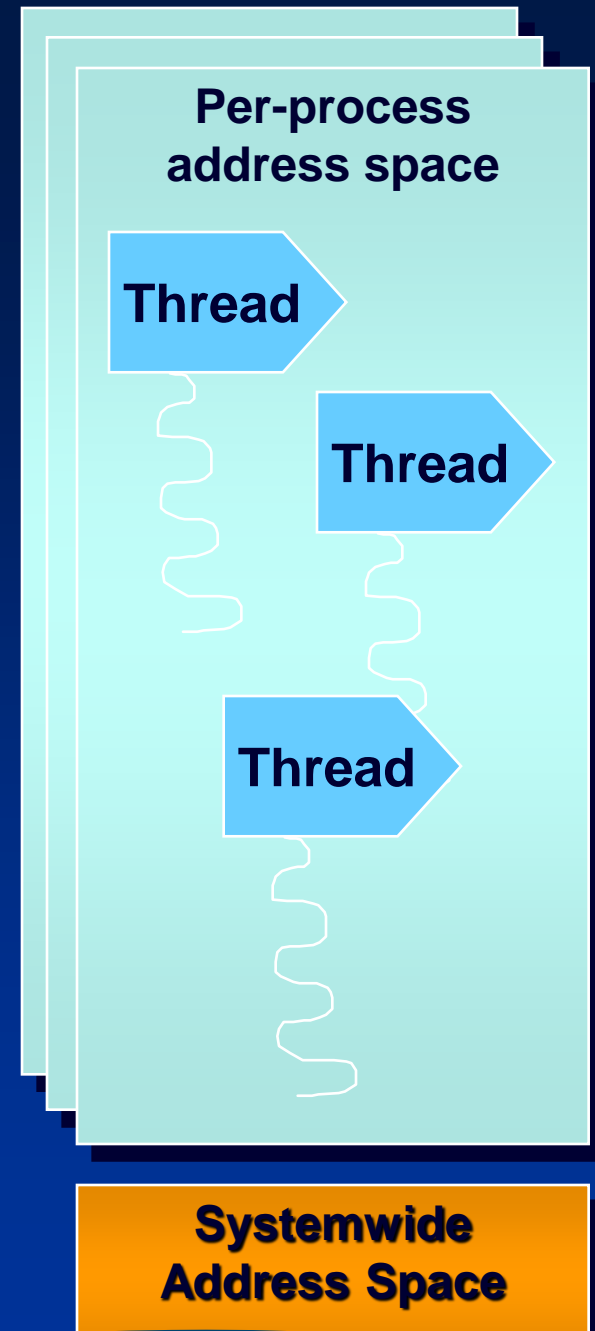
Processes & Threads

What is a thread?

- An **execution context** within a process
- Unit of scheduling (threads run, processes don't run)
- All threads in a process share the same per-process address space
 - Services provided so that threads can synchronize access to shared resources (critical sections, mutexes, events, semaphores)
- All threads in the system are scheduled as peers to all others, without regard to their "parent" process

System calls

- Primary argument to `CreateProcess` is image file name (or command line)
- Primary argument to `CreateThread` is a function entry point address



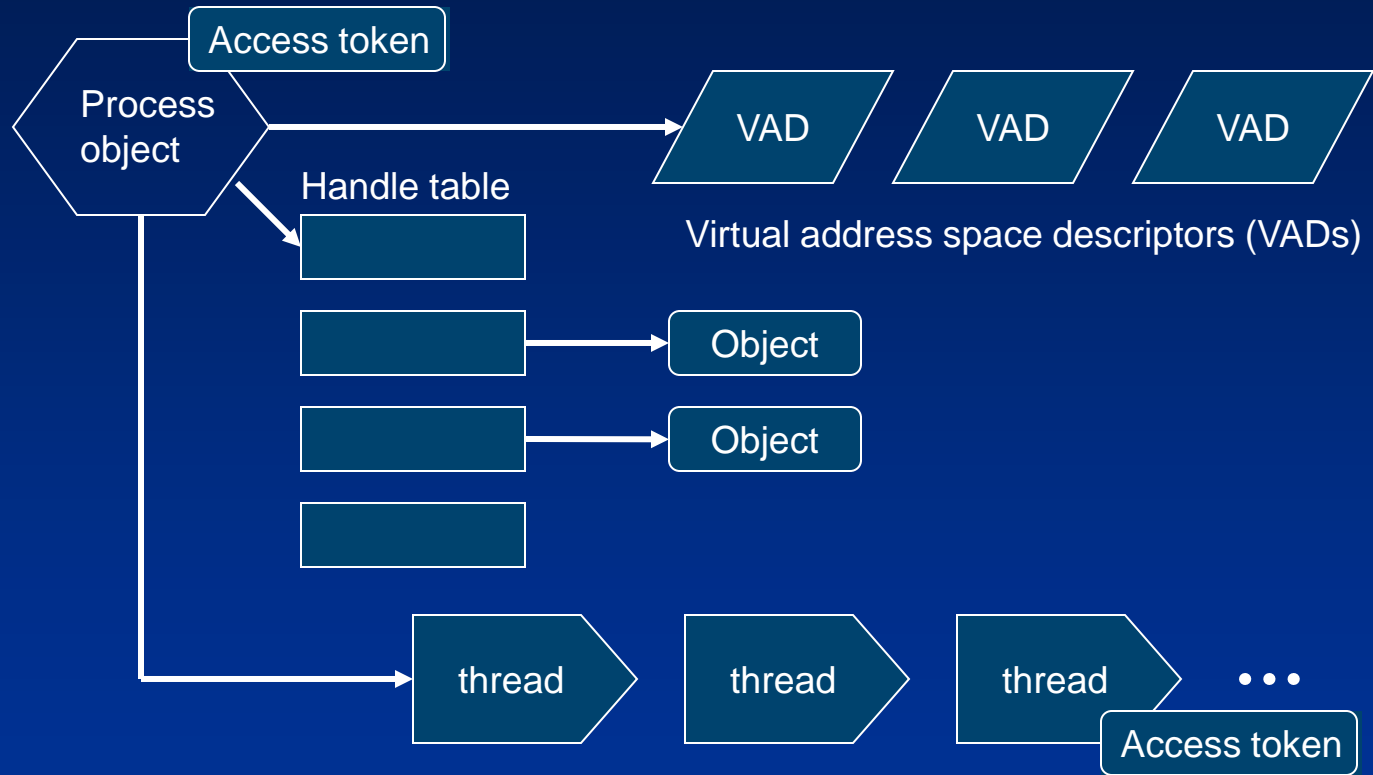
Processes & Threads

- Every process starts with one thread
 - First thread executes the program's "main" function
 - Can create other threads in the same process
 - Can create additional processes
- Why divide an application into multiple threads?
 - Perceived user responsiveness, parallel/background execution
 - Examples: MS Word background print – can continue to edit during print
 - Take advantage of multiple processors
 - On an MP system with n CPUs, n threads can literally run at the same time
 - Question: given a single threaded application, will adding a 2nd processor make it run faster?
 - Does add complexity
 - Synchronization
 - Scalability is a different question...
 - # of multiple run-able threads vs # CPUs
 - Having too many run-able threads causes excessive *context switching*

Memory Protection Model

- No user process can touch another user process address space (without first opening a **handle** to the process, which means passing through **Windows security**)
 - Separate **process page tables** prevent this
 - “Current” page table changed on context switch from a thread in one process to a thread in another process
- No user process can touch **kernel memory**
 - Page protection in **process page tables** prevent this
 - OS pages only accessible from “kernel mode”
 - x86: Ring 0, Itanium: Privilege Level 0
 - Threads change from user to kernel mode and back (via a secure interface) to execute kernel code
 - Does not affect scheduling (*not a context switch*)

A Process and its Resources



Acquire an access token to impersonate other process

Virtual Memory

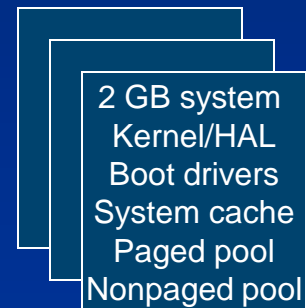
- 32-bit address space (4 GB)
 - 2 GB user space (per process)
 - 2 GB operating system
- 64-bit address space
 - 7192 GB user space (Itanium)
 - 8192 GB user space (x64)
 - ~6000 GB operating system
- Memory manager maps virtual onto physical memory

Default 32-bit layout

Unique per process



Systemwide



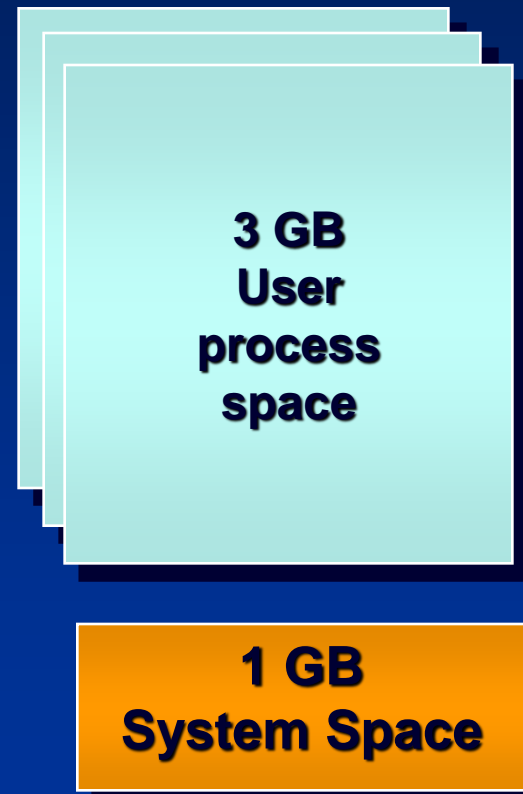
32-bit x86 Address Space

● 32-bits = 2^{32} = 4 GB

Default



3 GB user space



Boot time option: *increaseuserva*

Even Larger User Address Space?

- Address Windowing Extension (AWE)
 - 32-bit application to allocation up to 64GB of physical memory
 - Map views/windows onto 2GB virtual address space
 - Burden on the programmer

Kernel Mode vs. User Mode

- No protection against components running in kernel mode
- Transition from user mode to kernel mode through special instruction (processor changes privilege level)
 - OS traps this instruction and validates arguments to syscalls
 - Transition from user to kernel mode does not affect thread scheduling
- Performance Counters: System/Processor/Process/Thread – Privileged Time/User time
 - Windows kernel is thoroughly instrumented
 - Hundreds of performance counters throughout the system
- Performance Monitor – perfmon.msc - MMC snap-in

Fibers vs. Threads

- Schedule its own “threads” of execution
- Not relying on Windows build-in scheduler
- “Light-weight threads”
- To create an initial fiber:
 - Call *ConvertThreadToFiber*
- To create additional fiber from existing one:
 - Call *CreateFiber*
- To run a fiber:
 - Call *SwitchToFiber*

Objects and Handles

- **Object**: single, runtime instance of statically defined type
- *Process, thread, file, event* objects in Windows - are based on low-level executive objects
- Object **attributes**: defines object's state
- Object **methods**: means for manipulating objects – read/write attributes

Objects and Handles (cont'd)

- Objects enable:
 - Human-readable names for system resources
 - Resource sharing among processes
 - Resource protection against unauthorized access
 - Reference counting – let system know when an object is no longer in use and can be deallocated

Security

- Key capabilities:
 - Mandatory integrity protection of all **shareable system objects** (files, directories, processes, threads)
 - Security auditing
 - User authentication at logon
 - Prevention of access of other user's uninitialized resources (e.g. free memory)
- Three forms of access control:
 - Discretionary control: read/write/access permissions
 - Privileged access: administrator may take ownership of files
 - Mandatory integrity control: protection within the same account (e.g. protected mode internet explorer)

Common Criteria

- New standard, called Common Criteria (CC), is the new standard for computer security certification
 - Consortium of US, UK, Germany, France, Canada, and the Netherlands in 1996
 - Became ISO standard 15408 in 1999
 - For more information, see <http://www.commoncriteriaportal.org/> and <http://csrc.nist.gov/cc>
- CC is more flexible than TCSEC trust ratings, and includes concept of Protection Profile (PP) to collect security requirements into easily specified and compared sets, and the concept of Security Target (ST) that contains a set of security requirements that can be made by reference to a PP
- Windows XP and Server 2003 was certified as compliant with the CC Controlled Access Protection Profile (CAPP) in 2006

Networking

- Integral, application-transparent networking services
 - Basic file and print sharing and using services
- A platform for distributed applications
 - Application-level inter-process communication (IPC)
- Windows provides an expandable platform for other network components

Registry

- System database : boot & config info
- System wide software settings: operation of Windows
- Security database
- Per-user profile settings
- Window to In-memory volatile data (current hardware state)
 - What devices are loaded?
 - Resources used by devices
 - Performance counters are accessed through registry functions
- Regedit.exe is the tool to view/modify registry settings
 - HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control
 - HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services
 - HKEY_LOCAL_MACHINE\Software

Unicode

- Most internal text strings are stored/processed as 16-bit wide Unicode strings
- Windows API string functions have 2 versions
 - Unicode (wide) version
 - L“This string uses 16-bit characters“
 - ANSI(narrow) version
 - “This string uses 8-bit characters“
 - Generic character representation in Windows API
 - _T (“This string uses generic characters“)

(Windows 95/98/ME have Windows API but no Unicode characters, Windows CE has Windows API but Unicode characters only)

Tools used to dig in

- Many tools available to dig into Windows internals
 - Helps to see internals behavior “in action”
- We'll use these tools to explore the internals
 - Many of these tools are also used in the labs that you can do after each lecture
- Several sources of tools
 - Support Tools
 - Resource Kit Tools
 - Debugging Tools
 - Sysinternals.com
- Additional tool packages with internals information
 - Platform Software Development Kit (SDK)
 - Device Driver Development Kit (DDK)

Tools for Viewing Windows Internals

Tool	Image Name	Origin
Startup Programs Viewer	AUTORUNS	www.sysinternals.com
Dependency Walker	DEPENDS	Support Tools, Platform SDK
DLL List	LISTDLLS	www.sysinternals.com
EFS Information Dumper	EFSDUMP	www.sysinternals.com*
File Monitor	FILEMON	www.sysinternals.com
Global Flags	GFLAGS	Support Tools
Handle Viewer	HANDLE	www.sysinternals.com
Junction tool	JUNCTION	www.sysinternals.com
Kernel debuggers	WINDBG, KD	Debugging tools, Platform SDK, Windows DDK
Live Kernel Debugging	LIVEKD	www.sysinternals.com
Logon Sessions	LOGINSESSIONS	www.sysinternals.com
Object Viewer	WINOBJ	www.sysinternals.com
Open Handles	OH	Resource kits
Page Fault Monitor	PFMON	Support Tools, Resource kits, Platform SDK
Pending File Moves	PENDMOVES	www.sysinternals.com

Tools for Viewing Windows Internals (contd.)

Tool	Image Name	Origin
Performance tool	PERFMON.MSC	Windows built-in tool
PipeList tool	PIPELIST	www.sysinternals.com
Pool Monitor	POOLMON	Support Tools, Windows DDK
Process Explorer	PROCEXP	www.sysinternals.com
Get SID tool	PSGETSID	www.sysinternals.com
Process Statistics	PSTAT	Support Tools, Windows 2000 Resource kits, Platform SDK, www.reskit.com
Process Viewer	PVIEWER (in the Support Tools) or PVIEW (in the Platform SDK)	Platform SDK
Quick Slice	QSLICE	Windows 2000 resource kits
Registry Monitor	REGMON	www.sysinternals.com
Service Control	SC	Windows XP, Platform SDK, Windows 2000 resource kits
Task (Process) List	TLIST	Debugging tools
Task Manager	TASKMGR	Windows built-in tool
TDImon	TDIMON	www.sysinternals.com

Support Tools

- A suite of management, administration and troubleshooting tools
 - Win2K: 40+ tools, WinXP: 70+ tools, Server 2003: 70 tools
- Located on Windows Installation CD in `\support\tools`
- Not shipped with installation since Windows Vista

Windows Resource Kit Tools

- Windows 2000 Server Resource Kit Tools (Supplement 1 is latest)
 - Not freely downloadable
 - Comes with MSDN & TechNet, so most sites have it
 - May be legally installed on as many PCs as you want at one site
 - Installs fine on 2000/XP Professional (superset of 2000 Professional Resource Kit)
- Windows XP/Vista/7 Resource Kit: no tools, just documentation
- Windows Server 2003 Resource Kit Tools
 - Free download – visit <http://www.microsoft.com/windows/reskits/default.asp>
 - Tool updates are at <http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=17657>
 - NOTE: Windows 2000 Server Resource Kit has more tools than 2003 Resource Kit (225 vs 115 .EXEs)
 - Many tools dropped due to lack of support
 - Tools are still officially unsupported
 - But, can send bug reports to ntreskit@microsoft.com

Windows Debugging Tools

- Separate package of advanced debugging tools
 - Installs XP, 2003, Vista and Win 7
- Download latest version from:
 - <http://www.microsoft.com/whdc/ddk/debugging>
- Tools
 - User-mode and kernel-mode debuggers
 - Kd – command line interface
 - WinDbg – GUI interface (kernel debugging still mostly “command line”)
 - Allow exploring internal system state & data structures
 - Ntsd, Cdb – command line user-mode debugger (newer versions than what ships with OS)
 - Misc other tools (some are also in Support Tools):
 - kill, remote, tlist, logger/logview (API logging tool), Autodump

Live Kernel Debugging

- Useful for investigating internal system state not available from other tools
 - Previously, required 2 computers (host and target)
 - Target would be halted while host debugger in use
- XP & Server 2003 support live local kernel debugging
 - Technically requires system to be booted /DEBUG to work correctly
 - You can edit kernel memory on the live system (!)
 - But, not all commands work
- LiveKd (<http://live.sysinternals.com/livekd.exe>)
 - Tricks standard Microsoft kernel debuggers into thinking they are looking at a crash dump
 - Works on Windows XP, Server 2003, Vista and Windows 7
 - Was originally shipped on Inside Windows 2000 book CD-ROM—now is free on Sysinternals
 - Commands that fail in local kernel debugging work in LiveKD:
 - Kernel stacks (!process, !thread)
 - Lm (list modules)
 - Can snapshot a live system (.dump)
 - Does not guarantee consistent view of system memory
 - Thus can loop or fail with access violation
 - Just quit and restart

Sysinternals Tools

- **Freeware Windows internals tools from www.sysinternals.com**
 - Written by Mark Russinovich & Bryce Cogswell (cofounders of Winternals)
- **Useful for developers, system administrators, and power users**
 - Most popular: Process Explorer, Diskmon, TCPView
- **Require no installation – run them directly after downloading and unzipping**
- **Many tools require administrative privileges**
 - Some load a device driver
- **Tools regularly updated, so make sure to check for updated versions**
 - RSS feed available
 - Free Sysinternals newsletter
 - See Mark's blog: <http://blogs.technet.com/b/markrussinovich/>

Platform SDK

(Software Development Kit)

- a set of tools, code samples, documentation, compilers, headers, and libraries developers can use to create applications that run on Microsoft Windows operating systems using native (Win32) or managed (.NET Framework) programming models.
 - “Core SDK” contains core services, COM, messaging, active directory, management, etc.
- **Latest version for Windows 7:**
<http://msdn.microsoft.com/en-us/windows/bb980924>
 - Part of MSDN Professional (or higher) subscription
- **Always matches operating system revision**
 - Check the “archive”
- **Not absolutely required for Win32 development (because VC++ comes with the Win32 API header files), but...**
 - VC++ headers, libs, doc are not updated
 - Also provides a few tools (e.g. WinObj, Working Set Tuner) not available elsewhere

Lab: sysinternal website

The screenshot shows a web browser window with the address bar containing `http://technet.microsoft.com/en-us/sysinternals`. The page title is "Windows Sysinternals: Documentation, downloads and additional resources". The browser's address bar also shows a search bar with "Google". The website header includes "Windows Sysinternals" and a search bar with "Search TechNet with Bing". The navigation menu includes "Home", "Learn", "Downloads", and "Community". The main content area features the Windows logo and the text "Windows Sysinternals". Below this, a paragraph states: "The Sysinternals web site was created in 1996 by Mark Russinovich and Bryce Cogswell to host their advanced system utilities and technical information. Whether you're an IT Pro or a developer, you'll find Sysinternals utilities to help you manage, troubleshoot and diagnose your Windows systems and applications." A section titled "Get up to speed fast!" contains a list of links: "Read the official guide to the Sysinternals tools, The Windows Sysinternals Administrator's Reference", "Watch Mark's top-rated Case-of-the-Unexplained troubleshooting presentations", "Read Mark's Blog which highlight use of the tools to solve real problems", "Check out the Sysinternals Learning Resources page", and "Post your questions in the Sysinternals Forum". Below this, there are two columns: "Sysinternals Utilities" with links for "Sysinternals Suite" and "Utilities Index", and "Sysinternals Live" with a paragraph explaining the service and a link to the directory. A "What's New" section is partially visible at the bottom.

Windows Sysinternals: Documentation, downloads and additional resources

http://technet.microsoft.com/en-us/sysinternals

Search TechNet with Bing

United States (English) Sign In

Home Learn Downloads Community

Microsoft | TechNet

Windows Sysinternals

The Sysinternals web site was created in 1996 by [Mark Russinovich](#) and Bryce Cogswell to host their advanced system utilities and technical information. Whether you're an IT Pro or a developer, you'll find Sysinternals utilities to help you manage, troubleshoot and diagnose your Windows systems and applications.

Get up to speed fast!

- Read the official guide to the Sysinternals tools, [The Windows Sysinternals Administrator's Reference](#)
- Watch Mark's top-rated [Case-of-the-Unexplained](#) troubleshooting presentations
- Read [Mark's Blog](#) which highlight use of the tools to solve real problems
- Check out the Sysinternals [Learning Resources](#) page
- Post your questions in the [Sysinternals Forum](#)

Sysinternals Utilities

- [Sysinternals Suite](#)
- [Utilities Index](#)

Sysinternals Live

Sysinternals Live is a service that enables you to execute Sysinternals tools directly from the Web without hunting for and manually downloading them. Simply enter a tool's Sysinternals Live path into Windows Explorer or a command prompt as `http://live.sysinternals.com/<toolname>` or `\\live.sysinternals.com\tools\<toolname>`.

You can view the entire Sysinternals Live tools directory in a browser at <http://live.sysinternals.com>.

What's New

What's New (September 1, 2011)

Lab: Viewing the Process Tree

```
C:\Program Files\Debugging Tools for Windows (x86)>tlist.exe/t
```

```
System Process (0)
```

```
System (4)
```

```
  smss.exe (300)
```

```
  csrss.exe (392)
```

```
  wininit.exe (452)
```

```
    services.exe (500)
```

```
      svchost.exe (652)
```

```
        BTStackServer.exe (4352)
```

```
        WmiPrvSE.exe (4592)
```

```
        WmiPrvSE.exe (5080)
```

```
        wlcomm.exe (11144)
```

```
        FlashUtil10i_ActiveX.exe (10852) OleMainThreadWndName
```

```
        OfficeLiveSignIn.exe (9576) OleMainThreadWndName
```

```
        WmiPrvSE.exe (8024)
```

```
        WmiPrvSE.exe (9132)
```

```
      nvvsvc.exe (708)
```

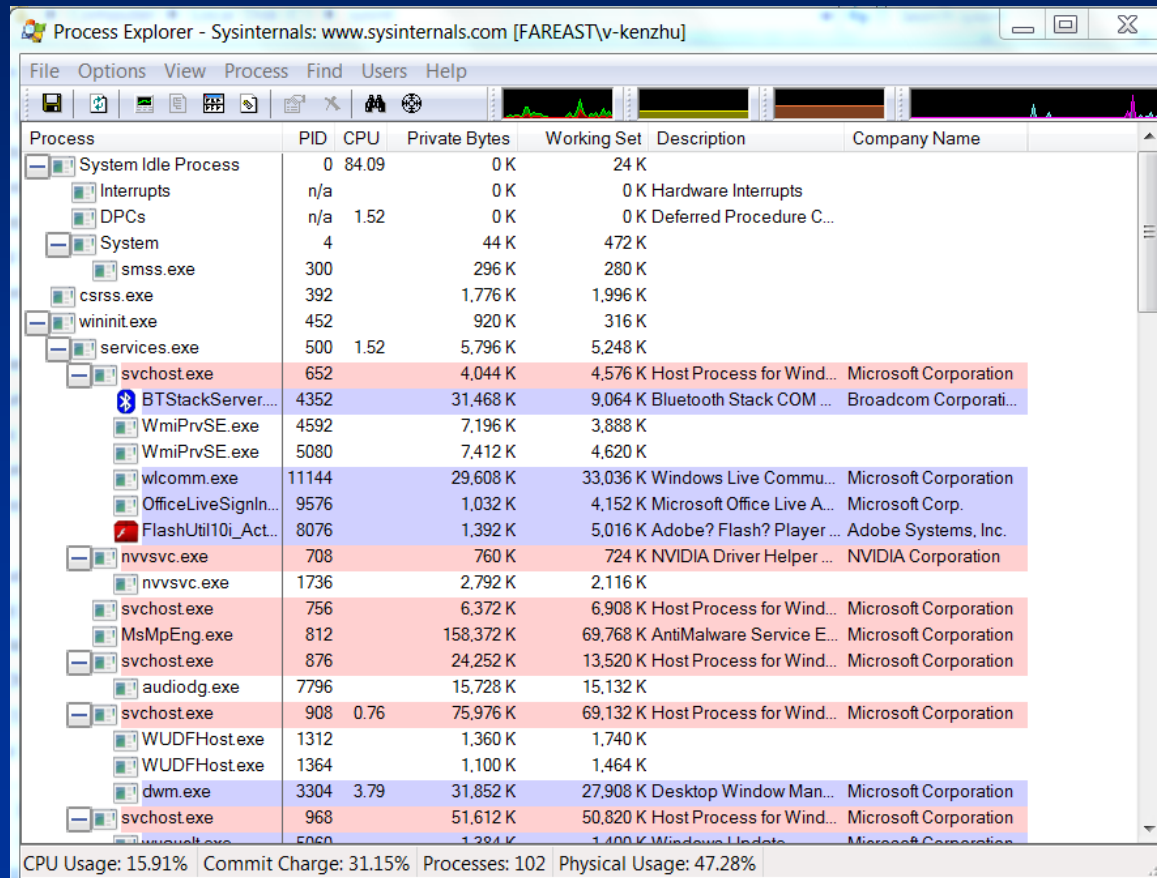
Lab: No more than Parent PID!

- Open a cmd prompt window
- Start another cmd prompt by typing “cmd” from the first window
- Bring up task manager
- Type “mspaint” from the second window
- Goto the second cmd window and type “exit” (notice mspaint still remains)
- Switch to task manager, click on “Application” tab
- Right click command prompt task select “Go to process”
- Click on cmd.exe highlighted in blue
- Right click on this process and select “End process tree”
- Click “yes” in the Task Manager Warning message box
- The first cmd window will disappear and mspaint remains since it's the grandchild

Lab: Using Process Explorer

“Super Task Manager”

- Shows full image path, command line, environment variables, parent process, security access token, open handles, loaded DLLs & mapped files



Process Explorer - Sysinternals: www.sysinternals.com [FAREAST\v-kenzhu]

File Options View Process Find Users Help

Process	PID	CPU	Private Bytes	Working Set	Description	Company Name
System Idle Process	0	84.09	0 K	24 K		
Interrupts	n/a		0 K	0 K	Hardware Interrupts	
DPCs	n/a	1.52	0 K	0 K	Deferred Procedure C...	
System	4		44 K	472 K		
smss.exe	300		296 K	280 K		
csrss.exe	392		1,776 K	1,996 K		
wininit.exe	452		920 K	316 K		
services.exe	500	1.52	5,796 K	5,248 K		
svchost.exe	652		4,044 K	4,576 K	Host Process for Wind...	Microsoft Corporation
BTStackServer...	4352		31,468 K	9,064 K	Bluetooth Stack COM ...	Broadcom Corporati...
WmiPrvSE.exe	4592		7,196 K	3,888 K		
WmiPrvSE.exe	5080		7,412 K	4,620 K		
wlcomm.exe	11144		29,608 K	33,036 K	Windows Live Commu...	Microsoft Corporation
OfficeLiveSignl...	9576		1,032 K	4,152 K	Microsoft Office Live A...	Microsoft Corp.
FlashUtil10i_Act...	8076		1,392 K	5,016 K	Adobe? Flash? Player ...	Adobe Systems, Inc.
nvsvcs.exe	708		760 K	724 K	NVIDIA Driver Helper ...	NVIDIA Corporation
nvsvcs.exe	1736		2,792 K	2,116 K		
svchost.exe	756		6,372 K	6,908 K	Host Process for Wind...	Microsoft Corporation
MsMpEng.exe	812		158,372 K	69,768 K	AntiMalware Service E...	Microsoft Corporation
svchost.exe	876		24,252 K	13,520 K	Host Process for Wind...	Microsoft Corporation
audiodg.exe	7796		15,728 K	15,132 K		
svchost.exe	908	0.76	75,976 K	69,132 K	Host Process for Wind...	Microsoft Corporation
WUDFHost.exe	1312		1,360 K	1,740 K		
WUDFHost.exe	1364		1,100 K	1,464 K		
dwm.exe	3304	3.79	31,852 K	27,908 K	Desktop Window Man...	Microsoft Corporation
svchost.exe	968		51,612 K	50,820 K	Host Process for Wind...	Microsoft Corporation
wmacthlp.exe	5060		1,284 K	1,400 K	Windows Update	Microsoft Corporation

CPU Usage: 15.91% Commit Charge: 31.15% Processes: 102 Physical Usage: 47.28%

Lab: Viewing Proc. Info in TaskMgr

Windows Task Manager

File Options View Windows Help

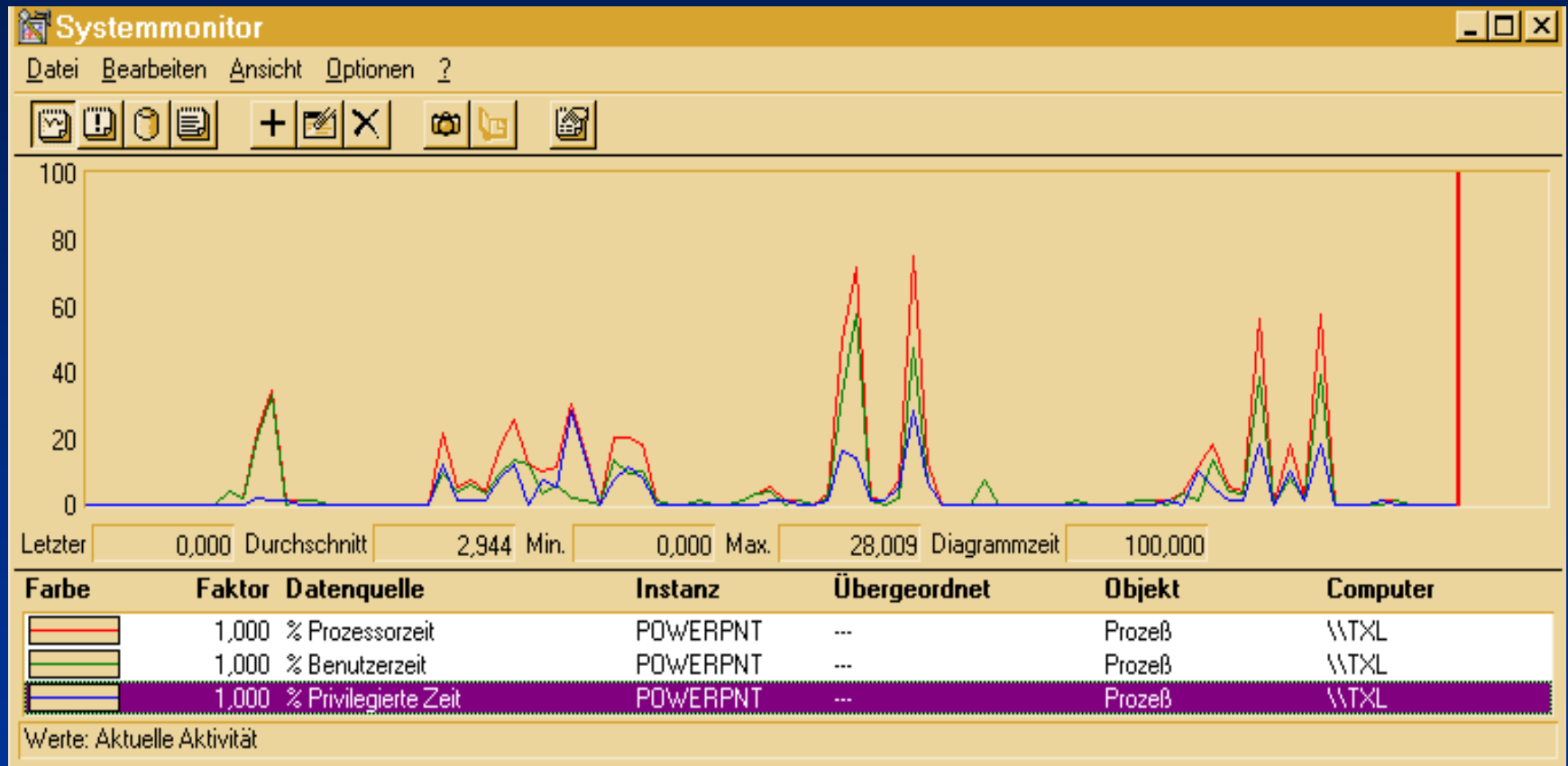
Applications Processes Services Performance Networking Users

Task	Status
C:\Windows\system32\cmd.exe	Running
Computer	Running
Inbox in kzhu@cs.sjtu.edu.cn - Microsoft Outlook	Running
Kenny Zhu (Available)	Running
login	Running
Microsoft PowerPoint - [2_Win-concepts-tools [Com...	Running
Performance Monitor	Running
Skype™ - kenzhu2000	Running
Untitled - Notepad	Running
xterm	Running
xterm	Running

End Task Switch To New Task...

Processes: 105 CPU Usage: 18% Physical Memory: 41%

Lab: Performance Monitor



Lab: Explorer to view security attributes

● Lab:

- Use Explorer to view Windows FS Access rights/ownerships, ACLs
- Passwd change: CTRL-ALT-DEL (secure login sequence)

Lab: Kernel Debugging

- Run livekd from elevated command prompt
- To display the kernel structures
 - `dt nt!_*` -- for all
 - `dt nt!_*interrupt*` -- for interrupt object
 - `dt nt!_kinterrupt` -- show details of a specific structure
 - `dt nt!_kinterrupt -r` -- show substructures

Further Reading

- Mark E. Russinovich, *et al.*
Windows Internals,
 - 5th Edition, Microsoft Press, 2009.
 - Ch 1. Concepts and Tools (pp. 1 – pp. 32)