Windows 8 Internals Course

Introduction

This 5-day course gives developers and support engineers the knowledge to design and troubleshoot with Windows 8, from an architectural operating system perspective. This course presents the operating system internal operation of Windows 8. Windows 8 is used during the course by students and instructor.

Course Objectives

At course completion, students should have the following knowledge and skills:

- A thorough understanding of the design of the Win8 kernel
- The role of objects to model and manage OS resources
- The internal design of processes and threads of the kernel
- An understanding of the I/O subsystem and the way that device drivers interact
- An understanding of the memory management techniques used in the kernel
- Knowledge of the NTFS file system
- Knowledge of the registry and the event log
- User Access Control (UAC) goals and "standard user rights"
- Application compatibility with previous Windows versions
- Virtualization options
- The ability to analyze crash dumps (kernel and application)

Prerequisites

Before taking this course, students should have the following skills:

- Operating system concepts such as
  - Memory management
  - Resource management
  - Reentrancy
  - File system management
- Debugging concepts and some techniques
- Preferable: Experience with Windows programming (C, C++, etc.)

Course Structure

This course is a combination of lecture and "hands-on" labs. Lectures include numerous demonstrations. Principles presented in each lecture are reinforced with the student labs. For example, a system blue screen is forced causing a crash dump. The crash is then analyzed using WinDbg.

Course Outline

Windows 8 Overview

- History of Windows
- Why “Touch First”?
- Design Goals
- Features of the OS
- Threads
- Processes
- Virtual Memory in Windows

The Windows 8 Kernel

- Kernel Objects
- Internal Data Structures
- Thread Scheduling and Preemption
- Interrupt and Exception Handling
- Wait Internals
- Multiprocessor Optimizations

Multi-Threading Programming

- Creating threads
- Controlling threads
- Setting thread priority
- Need for synchronization
- Mutexes
- Semaphores
- Events
- Critical sections

The Windows API

- Win32 vs. Win64
- Unicode vs. MBCS
- Working with files
- Miscellaneous functions

The I/O Subsystem

- The Windows I/O Model
- I/O Processing
- Driver Operation
- KMDF
- UMDF
- Plug-and-Play Manager
- Power Manager

NTFS

- NTFS Features
- The Master File Table (MFT)
- Clusters
- Attributes
- Directories
- NTFS Log File
- File Compression
- Windows Search

Memory Management

- Virtual Address Translation
- Page Faults
- Working Set Management
- Physical Memory Management

User Access Control

- Goals of UAC (Just “turn it off”, right?)
- The Protected Admin Account
- Virtualized Registry & File System
- The UAC API
- Writing “good” applications with UAC

Registry and WMI Services

- The System Registry
- Windows Management and Instrumentation
- The WMI Classes
- The WBEM Object Browser
- Event logging

Virtualization

- Why virtualization?
- Microsoft virtualization SW options
- Competitors for virtualization
- Creating a new virtual PC
- Windows 8 & virtualization

Application Compatibility

- Why old apps fail on new OS's
- Device Driver incompatibilities
- UAC issues
- UI issues
- Troubleshooting application incompatibility
- Compatibility options

Crashes & Dump Files

- Why Windows crashes
- Memory Dump Options
- Analyzing a Crash Dump with WinDbg
- User mode dump files
- Capturing and analyzing a user mode crash

Windows 8.1

- Windows "Blue" Overview
- The "new" Start button
- UMDF 2.0
- Other enhancements

