

Windows 7的新特性

——背景和MinWin

张银奎 (Raymond Zhang)

Win7简介

- 基于Vista精雕细琢
 - 磨去棱角
 - 改善稳定性、安全、用户体验（兼容性）
- 提前完成
 - 2009年7月22日RTM
 - 10月22日正式零售
- 一个时代的结束
- 一个时代的开始



主要的功能团队

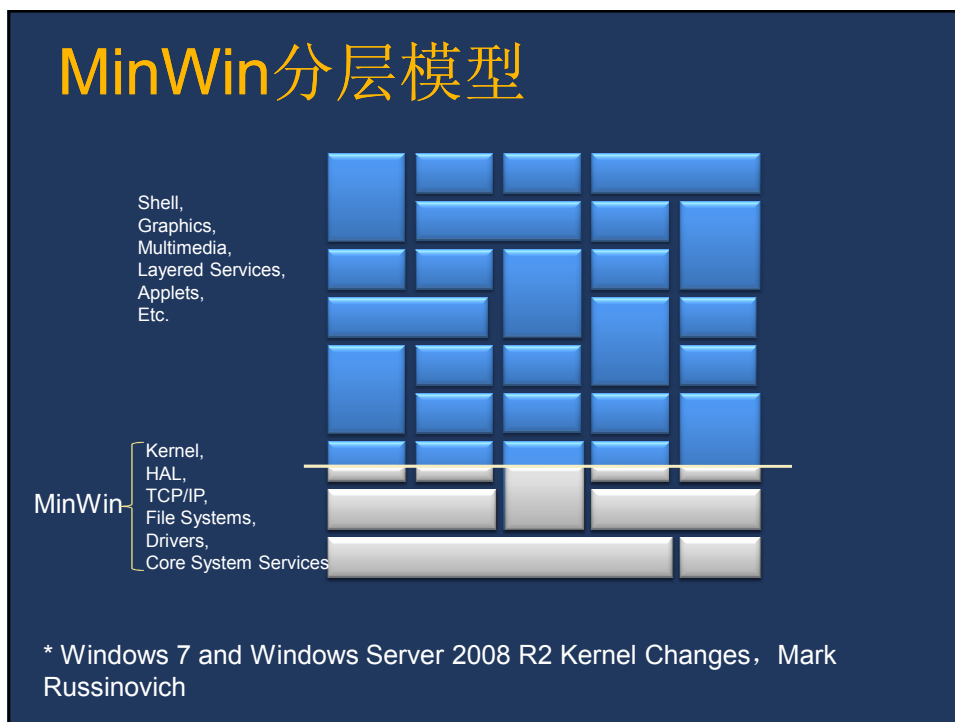
- | | |
|--|---|
| 1. Applets and Gadgets | 12. Find and Organize |
| 2. Assistance and Support Technologies | 13. Fundamentals |
| 3. Core User Experience | 14. Internet Explorer (including IE 8 down-level) |
| 4. Customer Engineering and Telemetry | 15. International |
| 5. Deployment and Component Platform | 16. Kernel & VM |
| 6. Desktop Graphics | 17. Media Center |
| 7. Devices and Media | 18. Networking - Core |
| 8. Devices and Storage | 19. Networking - Enterprise |
| 9. Documents and Printing | 20. Networking - Wireless |
| 10. Engineering System and Tools | 21. Security |
| 11. File System | 22. User Interface Platform |
| | 23. Windows App Platform |

3

MinWin

- 问题：模块的依赖和高耦合严重制约Windows的发展
- 2003年开始MinWin项目，Vista的失败给这个项目绝好的机会
- 从架构上动手，分隔区块，解耦合
- 把核心部分独立出来
 - 能够单独启动
 - 让内核和底层可以分别演进，独立发展
- 组成
 - 内核、HAL、文件系统、TCP/IP栈、部分驱动程序
 - 不包含WMI、图形、声音和外壳(Shell)
- 150个模块，在磁盘上25MB，内存中40MB





DLL重构

- 重构Windows子系统DLL，解开应用层与MinWin的耦合
- 应用层仍使用老式DLL
 - Kernel32.dll, user32.dll, advapi32.dll, ...
- MinWin公开新的DLL作为对外接口
 - Kernelbase.dll
- 老式DLL转发调用到新的DLL
 - Kernel32.dll -> Kernelbase.dll

DLL转发

- Kernel32.dll和ADVAPI32.dll会把某些函数转发给kernelbase.dll
- 静态转发——编译期
- 动态转发——运行期，老的DLL中有Stub函数
- 可能给某些软件(如杀毒软件)带来问题
 - 遍历PE输出表
 - 直接依赖函数的函数体汇编指令

demo

观察Win7引入的子系统 DLL重构

9

虚拟DLL

- API接口与实现分离
- 每个API集合称为一个虚拟DLL
- 目前一共有34个API集合
 - API-MS-WIN-CORE-LIBRARYLOADER-L1-1-0.dll
 - API-MS-WIN-CORE-LOCALREGISTRY-L1-1-0.dll
 - API-MS-WIN-CORE-LOCALIZATION-L1-1-0.dll

Requirements	
Minimum supported client	Windows 7
Minimum supported server	Windows Server 2008 R2
Header	Winreg.h (include Windows.h)
DLL	Api-ms-win-core-localregistry-l1-1-0.dll
Unicode and ANSI names	RegDeleteKeyExW (Unicode) and RegDeleteKeyExA (ANSI)

RegDeleteKeyEx (Api-ms-win) Function

[This function is exported from Api-ms-win-core-localregistry-l1-1-0.dll in Windows 7 and Windows Server 2008 R2. This may change in subsequent versions. Instead of calling this function directly in Api-ms-win-core-localregistry-l1-1-0.dll, call the function through Advapi32.dll. See [RegDeleteKeyEx](#).]

Deletes a subkey and its values from the specified platform-specific view of the registry.

伪装的虚拟DLL

- 位于c:\windows\system32目录下
- 默认隐藏
- 只包含导出表信息
- 目的是为了更方便检查程序

c:\Windows\System32*		
Name	Ext	Size
api-ms-win-core-datetime-l1-1-0	dll	3,072
api-ms-win-core-debug-l1-1-0	dll	3,072
api-ms-win-core-delayload-l1-1-0	dll	3,072
api-ms-win-core-errorhandling-l1-1-0	dll	3,072
api-ms-win-core-file-l1-1-0	dll	3,072
api-ms-win-core-file-l1-1-0	dll	5,120
api-ms-win-core-handle-l1-1-0	dll	3,072
api-ms-win-core-heap-l1-1-0	dll	3,584
api-ms-win-core-interlocked-l1-1-0	dll	3,584
api-ms-win-core-io-l1-1-0	dll	3,072
api-ms-win-core-libraryloader-l1-1-0	dll	3,584
api-ms-win-core-localization-l1-1-0	dll	4,096
api-ms-win-core-localregistry-l1-1-0	dll	4,096
api-ms-win-core-memory-l1-1-0	dll	3,584
api-ms-win-core-misc-l1-1-0	dll	4,096
api-ms-win-core-namedpipe-l1-1-0	dll	3,584
api-ms-win-core-processenvironment-l1-1-0	dll	3,584
api-ms-win-core-processthreads-l1-1-0	dll	4,608
api-ms-win-core-profile-l1-1-0	dll	3,072
api-ms-win-core-rtlsupport-l1-1-0	dll	3,072
api-ms-win-core-string-l1-1-0	dll	3,072
api-ms-win-core-synch-l1-1-0	dll	4,096
api-ms-win-core-sysinfo-l1-1-0	dll	4,096
api-ms-win-core-threadpool-l1-1-0	dll	4,608
api-ms-win-core-util-l1-1-0	dll	3,072

11

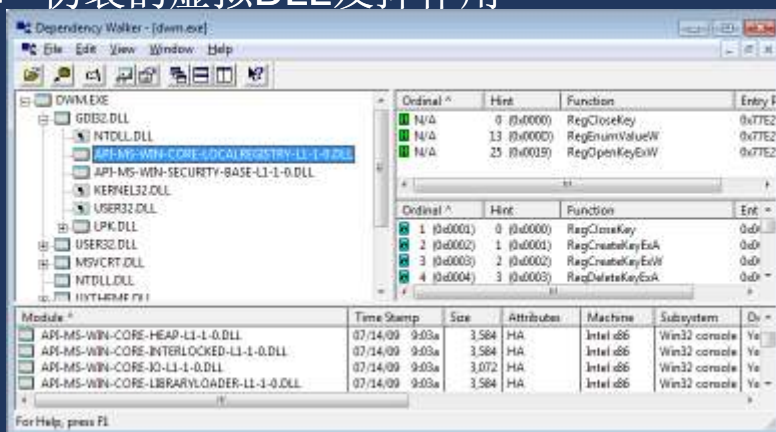
逻辑DLL

- 真正的API实现
- 目前: kernel32.dll, ntdll.dll, sechost.dll, 以及KernelBase.dll
- SECHOST.DLL
 - 系统服务有关的API, CreateService, StartService
 - Local Security Authority: LsaXXX

12

观察依赖性

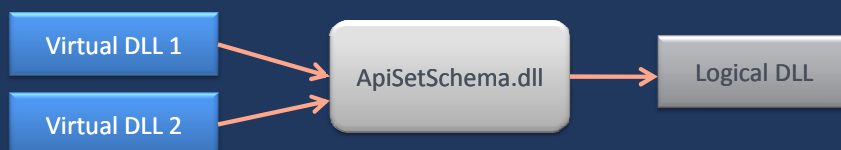
- 使用Depends观察依赖性
- 伪装的虚拟DLL发挥作用



13

虚拟DLL绑定

- 映射关系存储在Apisetschema.dll中
- 内核在启动阶段加载这个DLL，并将其映射到每个进程空间中
- 进程加载器在加载无路径的DLL时会在Schema表中搜索
- LdrpInitializeProcess函数



14

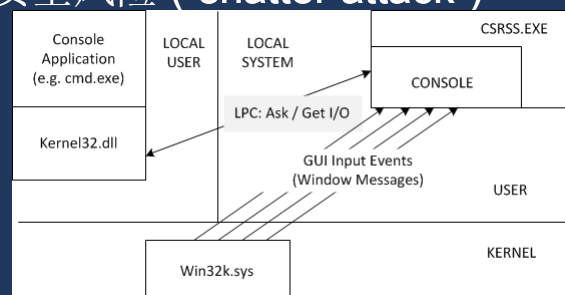
demo

观察虚拟DLL和逻辑DLL的绑定

15

控制台窗口的变化(1/2)

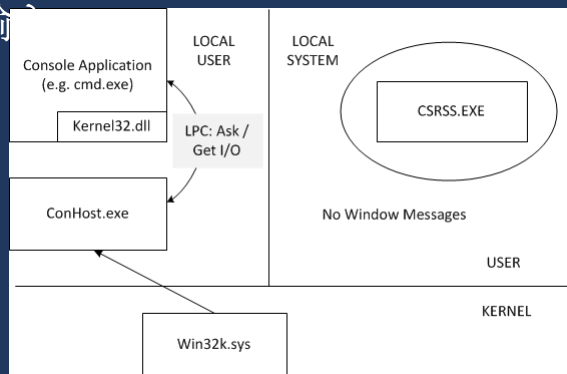
- 此前使用CSRSS进程来管理控制台窗口的消息循环
- 安全风险 (“shatter attack”)



* <http://blogs.technet.com/askperf/archive/2009/10/05/windows-7-windows-server-2008-r2-console-host.aspx>

控制台窗口的变化(2/2)

- 引入新的进程Conhost.exe
- 转发键盘输入
- 一对一



csrss.exe	588	Client Server Runtime Process
conhost.exe	3160	Console Window Host

问与答



Windows 7的新特性

——后台进程管理

张银奎 (Raymond Zhang)

问题

- 系统服务越来越多
 - Win7安装后有140多个服务
- Win7以前只有两种启动类型
 - On demand
 - Automatic
- 自动启动的服务会在系统启动阶段启动
 - 影响用户登录的速度
 - 始终占有系统资源
 - 增加关机和睡眠的延迟

20

UBPM

- Unified Background Process Manager
- 整合旧的进程管理器
 - Service Control Manager
 - Task Scheduler
 - Windows Management Instrumentation
 - DCOM Server Process Launcher
- 支持事件触发
- 主要逻辑实现在UBPM.DLL中

21

重温ETW

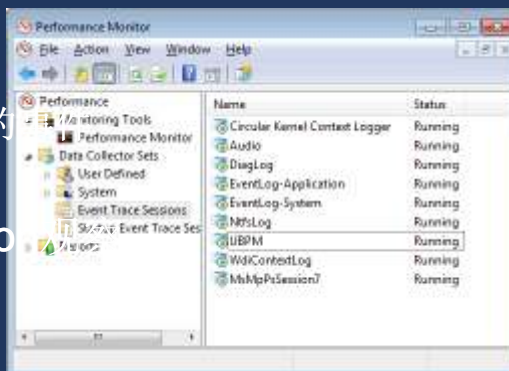
- Event Tracing for Windows
- Windows 2000引入的事件追踪机制
- 格式信息与变量域分离
- 高效，开销低，适合频繁输出



22

依赖ETW事件触发动作

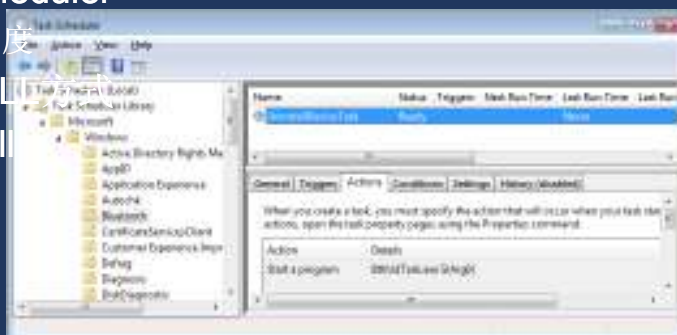
- 使用名为UBPM的ETW会话
- 默认已经注册很多事件提供器
 - 触发来源
- 实时消耗器
 - 监听预先注册的事件
 - 通知触发听众
- 可以通过PerfMon



23

客户

- Service Control Manager
 - Services.exe
 - 服务管理器
- Task scheduler
 - 任务调度
- 进程内DLL
- UBPM.dll



触发方式启动服务

- 多种触发方式
 - 硬件设备到达和移除(arrival/removal)
 - Bthserv: 蓝牙设备到达时启动
 - IP地址到达和移除
 - Lmhosts: IP地址可用时启动, IP地址不可用时停止
 - 防火墙短端口事件(Firewall port event)
 - Browser: open of NS and DGM ports
 - 加入和离开域(Domain join and unjoin)
 - W32Time: start on join, stop on unjoin
 - 定制的ETW事件
 - EFS: 第一次访问EFS加密的文件时启动
- 使用API定制, 保存在注册表中

通过API配置触发启动

```
SERVICE_TRIGGER trigger = {0};
trigger.dwTriggerType =
    SERVICE_TRIGGER_TYPE_IP_ADDRESS_AVAILABILITY;
trigger.dwAction =
    SERVICE_TRIGGER_ACTION_SERVICE_START;
trigger.pTriggerSubtype =
    (GUID*)&NETWORK_MANAGER_FIRST_IP_ADDRESS_ARRIVAL_GUID;
SERVICE_TRIGGER_INFO info = {0};
info.cTriggers = 1;
info.pTriggers = &trigger;
ChangeServiceConfig2 (hService,
    SERVICE_CONFIG_TRIGGER_INFO, &info);
```

查询触发启动信息

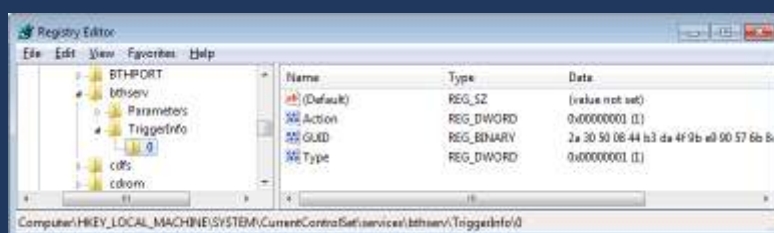
- 保存在注册表中
- “sc qtriggerinfo 服务名”

```

Administrator: C:\Windows\System32\cmd.exe
C:\Windows\System32>sc qtriggerinfo bthserv
[SC] QueryServiceConfig2 SUCCESS

SERVICE_NAME: bthserv

        START SERVICE
        SERVICE_INTERFACE_ATTRIBUTES: 0050302a-b344-4fda-9be9-98576b8d46f0 (I
        INTERFACE_CLASS_GUID:
  
```



27

更多触发启动的服务

Service Name	Description	Trigger Type
AELookupSvc	Processes application compatibility cache requests for applications as they are launched	Custom ETW
BDESVC	Provides BitLocker client services for user interface and auto-unlocking of data volumes	Custom ETW
BTHSERV	The Bluetooth service supports discovery and association of remote Bluetooth devices.	Device
SensorsMTPMonitor	Monitors MTP (Media Transfer Protocol) sensors (such as a cell phone with a GPS receiver) to communicate sensor data to programs	Device
TabletInputService	Enables Tablet PC pen and ink functionality	Device
WinDefend	Protection against spyware and potentially unwanted software	Group Policy

28

demo

Win7的任务调度器

29

问与答



Windows 7的新特性

——IT管理

张银奎 (Raymond Zhang)

Win7的可管理性

- Manageability
- 降低总拥有成本（TOC）
 - 系统恢复环境——WinRE
 - 故障诊断
 - 可靠性监视器
 - 资源监视器
 - AppLocker
 - 网络方面的增强

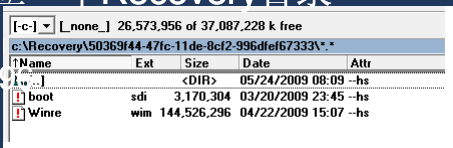


WinRE

- Windows Recovery Environment
- 基于WinPE 3.0 (Windows Preinstallation Environment)
- Vista引入此功能
 - 当受此前的各种系统恢复工具启发和影响 AdminPak
 - 放在安装光盘上
- Win7安装时，默认会将WinRE安装在硬盘上

安装

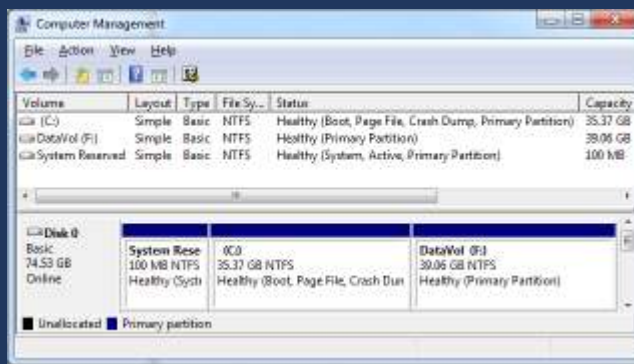
- 默认会为WinRE建立一个分区
 - 100MB
 - 分区类型为0x27（系统）
 - 在正常的Windows系统中不可见
- 在Win7所在盘，会建立一个Recovery目录
 - boot.sdi, 3,170,304
 - Winre.wim, 144,526,296
 - 资源管理器不可访问
 - 有管理员权限的工具可以查看
- 对于Vista系统，可以手工将WinRE安装到硬盘上
 - <http://blogs.msdn.com/winre/archive/2007/01/12/how-to-install-winre-on-the-hard-disk.aspx>



Name	Ext	Size	Date	Attr
<DIR>			05/24/2009 08:09	-hs
boot	sdi	3,170,304	03/20/2009 23:45	-hs
Winre	wim	144,526,296	04/22/2009 15:07	-hs

盘符

- 在Win7中观察，Win7所在盘的盘符为C:，D:不可用，第一个数据盘的盘符为E:
- 在WinRE中，C:为WinRE，D:为Win7

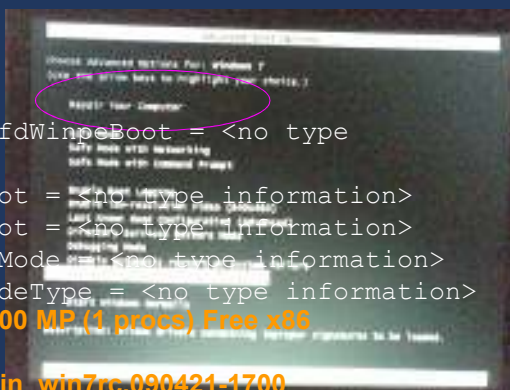


启动

- 正常启动失败后，下次会自动启动WinRE
- 启动早期按F8，选Repair Your Computer

```
kd> x nt!*WinPE*
8a9983a2 nt!SiCheckForUfdWinPEBoot = <no type
information>
8a9863fc nt!BiIsWinPEBoot = <no type information>
8a997b52 nt!SiIsWinPEBoot = <no type information>
8a93eac4 nt!InitIsWinPEMode = <no type information>
8a93eac8 nt!InitWinPEModeType = <no type information>
```

kd> version
Windows 7 Kernel Version 7100 MP (1 procs) Free x86
compatible
Built by: 7100.0.x86fre.winmain_win7rc.090421-1700



使用WinRE

- Recenv.exe, WinRE shell
- Startup Repair
 - 修正引导问题
- System Restore
 - 恢复到还原点
- System Image Recovery
 - 恢复系统映像
- Memory Diagnostic
 - 诊断内存
- Command Prompt
 - 控制台窗口



高级操作

- 操作注册表
 - 启动Command窗口
 - 运行regedit, 默认打开的是WinRE的注册表
 - 加载注册表文件 (File > Load Hive...)
 - D:\windows\system32\config
- 任务管理器
 - 执行taskmgr
- 启动其它程序
 - 找到EXE文件执行, 可能失败

定制WinRE

- 下载安装Windows Automated Installation Kit (AIK)
- 准备WinPE
 - 可以从Win7安装光盘复制
e:\sources\boot.wim
- 定制
 - CustomBackground
%systemroot%\system32\setup.bmp
- 配置脚本
- 生成映像文件
- <http://blogs.msdn.com/winre/archive/2006/12/12/creating-winre-using-waik.aspx>

调试WinRE

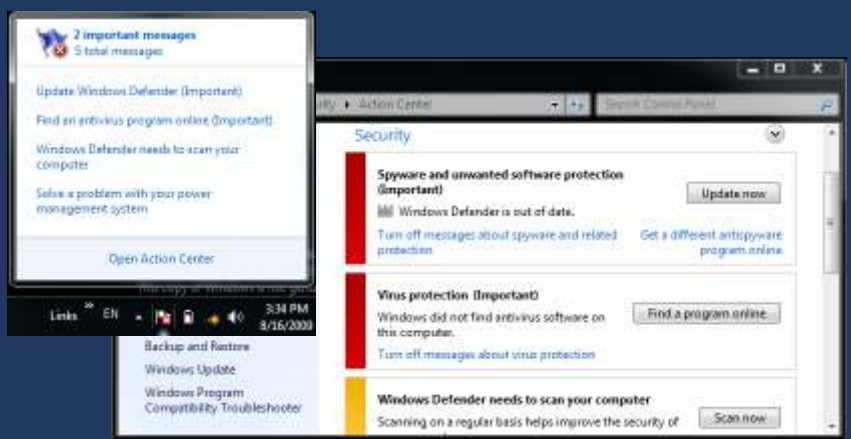
- 与普通的Windows内核调试类似
 - 两台机器，WinDBG，COM/1394/USB2.0
 - 在WinRE的命令行窗口启用
 - nt!InitlsWinPEMode
 - BOOL类型，WinRE中为TRUE
- ```
kd> db nt!InitlsWinPEMode l1
8a93aac4 01
```

## Troubleshooting Platform

- 减少IT支持费用
  - 少打电话，用户在向导的帮助下解决问题
- 一套可扩展的故障诊断框架
- 两个部分
  - 诊断包(Windows Troubleshooting Packs)
  - 构建工具(Windows Troubleshooting Pack Builder)，包含在SDK 7.0中
- 内建了20个诊断包，可以解决100多种常见问题

## 启动

- 多种途径启动：托盘区，帮助和支持中心，Action Center等



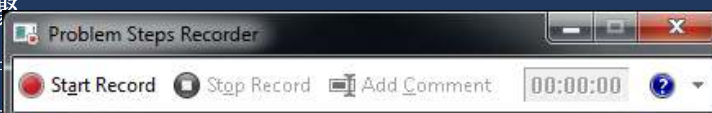
## 创建诊断包

- 包含在SDK 7.0中
- 向导方式
- 产生.diag文件



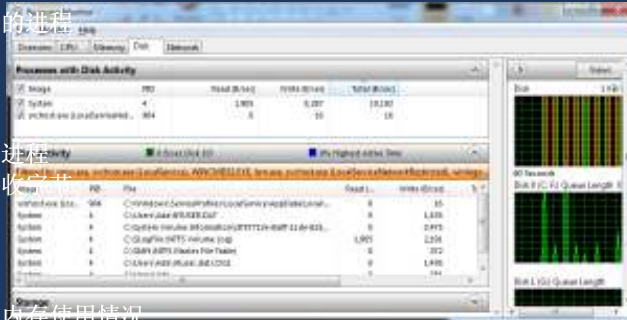
## Problem Steps Recorder

- Psr.exe
- 记录鼠标和键盘动作
- 抓快照，不是一直录像
- 产生压缩包
  - 步骤
  - 图片
  - 系统日志
  - 状态



## Resource Monitor

- CPU
  - 平均CPU占用率
  - 进程中的句柄，类似Process Monitor
  - 进程中的服务
- 磁盘
  - 包含磁盘行为的进程
  - 访问的文件
  - 每秒读写次数
- 网络
  - 有网络行为的进程
  - 每秒发送和接收字节
  - TCP连接
  - 监听端口
- 内存
  - 图形化的物理内存使用情况



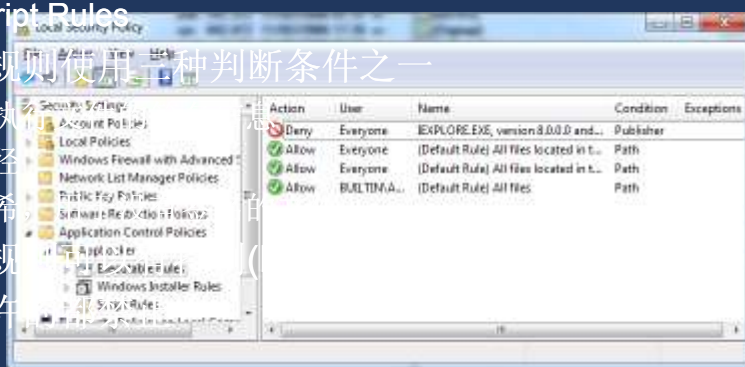
## AppLocker

- Win7和Svr2K8 R2引入，取代Software Restriction Policies
- 两种模式
  - 强制
  - Audit Only
    - 可以运行，事件记录到AppLocker日志
- Application Identify服务必须启动



## 基于规则

- 三种规则(Rules)
  - 可执行文件
  - Windows Installer
  - Script Rules
- 每条规则使用一种判断条件之一
  - 可执行文件
  - 路径
  - 哈希
- 每个规则
- 不允许



## 原理

- 内核态做的检查
  - NtCreateUserProcess
- 直接调用CreateProcess API来启动禁止的程序也会失败
- 基于数字签名和HASH值匹配, 改可执行文件的名称无法绕过

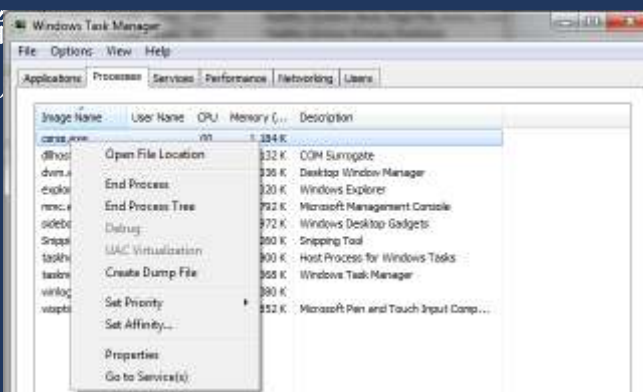
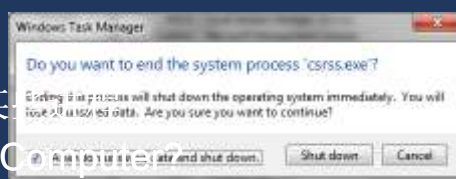


## 网络增强

- DirectAccess
  - 远程登录企业网，更好的用户体验
- BranchCache
  - 公司分支机构缓存企业网上的内容
  - 两种模式
    - 分布式(distributed cache)，点对点架构，缓存在第一个访问该内容的终端
    - 集中式(hosted cache)，C/S架构，缓存在服务器
- 基于URL的QoS
  - 根据URL定义流量控制

## 任务管理器的改进

- 可以终止任何进程
  - XP开始不能终止某些系统进程
  - Why Not? Whose Computer?
- 产生转储
  - “拍照”



## 其它增强

- PowerShell 2.0
  - 集成开发环境——ISE(Integrated Scripting Environment)
  - 500+ Cmdlets
- VHD管理和发布
- 查看可用网络 (View Available Network)、
- 动态驱动部署(Dynamic Driver Provisioning)  
驱动可以集中存储在服务器上，与映像分离
- Windows 7 Manageability Overview

## 问与答



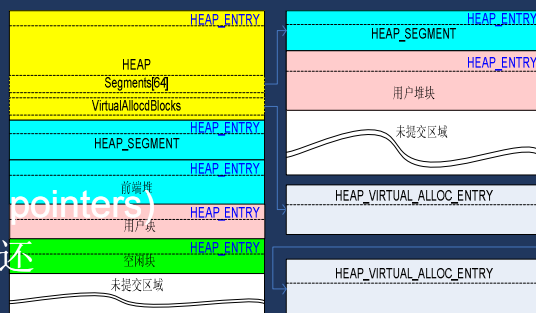
# Windows 7的新特性

——FTH

张银奎 (Raymond Zhang)

## 堆腐败

- Heap Corruption
- 错误使用堆
  - 堆块溢出
  - 多次释放
  - 野蛮指针(wild pointers)
  - 张家借，李家还
- 普遍问题
  - 15% of all user-mode crashes
  - 30% of user-mode crashes during shutdown
- 难以在第一时间发现，难以调试



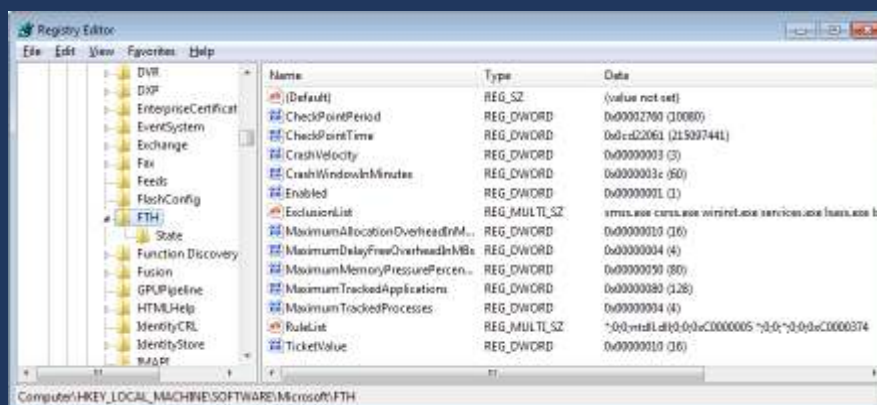
54

**FTH**

- Fault Tolerant Heap
- Win7引入
- 旨在降低堆误用的影响
- 程序崩溃时检查是否属于堆误用
- 动态应用缓解方案
- 跟踪缓解方案的效果，无效自动解除
- 通过WDI机制向ISV转发信息

## 注册表 (1/3)

- HKLM\Software\Microsoft\FTH



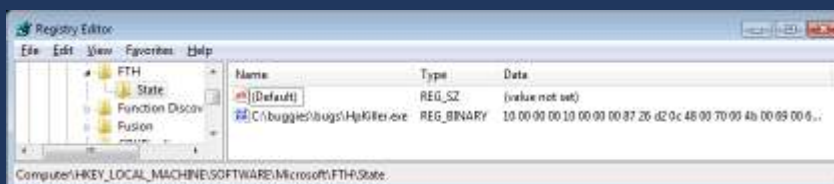
## 注册表 (2/3)

| 键名                                | 类型       | 默认值                        | 含义                |
|-----------------------------------|----------|----------------------------|-------------------|
| <b>CrashVelocity</b>              | DWORD    | 3                          | 崩溃次数阈值            |
| <b>CrashWindowInMinutes</b>       | DWORD    | 60                         | 时间跨度              |
| <b>Enabled</b>                    | DWORD    | 1                          | 是否启用              |
| <b>MaximumTrackedApplications</b> | DWORD    | 4                          | 同时追踪的最多程序数        |
| <b>MaximumTrackedProcesses</b>    | DWORD    | 128                        | 对于同一程序，同时追踪的最多实例数 |
| <b>CheckPointPeriod</b>           | DWORD    | 10,080 (7 days)            | 盘点周期              |
| <b>ExclusionList</b>              | MULTI_SZ | smss.exe<br>csrss.exe, ... | 排除列表              |

57

## 注册表 (3/3)

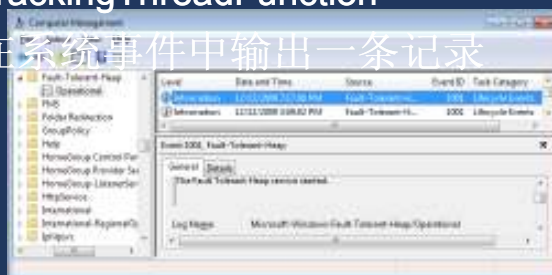
- State子键
- 用以记录正在追踪的程序
- 通过这个表键了解FTH在补偿(Shim)的程序



58

## FTH服务

- 实现在FTHSVC.DLL中
- 运行在SVCHOST进程中，作为WDI服务 (Diagnostic Policy Service)的一部分
  - FthServerMainThreadFunction
  - FthServerTrackingThreadFunction
- 启动后，会在系统事件中输出一条记录



59

## FTH补偿模块

- 基于Windows的模块补偿机制(DLL Shim)
  - 应用程序兼用
- C:\Windows\AppPatch\AcXtrnal.DLL
- 可以看做是FTH的客户端
- 内部类: NS\_FaultTolerantHeap

```

0:000> x AcXtrnal!*FTH*
6c860e10 AcXtrnal!NS_FaultTolerantHeap::FthFreeTr
6c860ac8 AcXtrnal!NS_FaultTolerantHeap::FthAssert
6c860e14 AcXtrnal!NS_FaultTolerantHeap::FthRandom
6c860e30 AcXtrnal!NS_FaultTolerantHeap::FthFreeTr
6c860e10 AcXtrnal!NS_FaultTolerantHeap::FthPerfor
6c860e08 AcXtrnal!NS_FaultTolerantHeap::FthWatson
6c860ac0 AcXtrnal!NS_FaultTolerantHeap::FthAssert
6c860e1c AcXtrnal!NS_FaultTolerantHeap::FthHeapLo
6c860f03c AcXtrnal!NS_FaultTolerantHeap::FthSpecia
6c860e2c AcXtrnal!NS_FaultTolerantHeap::FthHeapLo
6c860dc8 AcXtrnal!NS_FaultTolerantHeap::FthDelayF
6c8608c0 AcXtrnal!NS_FaultTolerantHeap::FthErrorH
6c860c0c AcXtrnal!NS_FaultTolerantHeap::FthMessag
6c860db4 AcXtrnal!NS_FaultTolerantHeap::FthClient
6c860e18 AcXtrnal!NS_FaultTolerantHeap::FthCookie

0:000> !m vm AcXtrnal
start end module name
6c800000 6ca89000 AcXtrnal
Image path: C:\Windows\AppPatch\AcXtrnal.DLL
Image name: AcXtrnal.DLL
Timestamp: Tue Jul 14 09:04:10 2009 (4A5ED98A)
CheckSum: 0003FA66
ImageSize: 00259000
File version: 6.1.7600.16385
Product version: 6.1.7600.16385
File flags: 0 (Mask: 3F)
File OS: 40004 NT Win32
File type: 2 0 DLL
File date: 00000000.00000000
Translations: 0409 04b0
CompanyName: Microsoft Corporation
Product Name: Microsoft® Windows® Operating System
Internal Name: Microsoft® Windows® Operating System
OriginalFilename: Microsoft® Windows® Operating System
ProductVersion: 6.1.7600.16385
FileVersion: 6.1.7600.16385 (win7_rtm.090713-1255)
FileDescription: Windows Compatibility DLL
LegalCopyright: © Microsoft Corporation. All rights reserved.

```

60

## 加载补偿模块

- LOADER在初始化进程期间调用 LdrpLoadShimEngine

```
0:000> kNL
ChildEBP RetAddr
00 0012f57c 77bf507c ntdll!KiFastSystemCallRet
01 0012f580 77c10fad ntdll!ZwMapViewOfSection+0xc
02 0012f5d4 77c11023 ntdll!LdrpMapViewOfSection+0xc7
03 0012f66c 77c0f4a6 ntdll!LdrpFindOrMapDll+0x303
04 0012f7ec 77c0f5f9 ntdll!LdrpLoadDll+0x2b2
05 0012f820 75c0bb94 ntdll!LdrLoadDll+0x92
06 0012f938 75c0b7f6 apphelp!SeiInit+0x4a7
07 0012fb2c 77bcd4c3 apphelp!SE_InstallBeforeInit+0x67
08 0012fb48 77bcd3c7 ntdll!LdrpLoadShimEngine+0xdb
09 0012fcb0 77c18fc0 ntdll!LdrpInitializeProcess+0x130b
0a 0012fd00 77c0b2c5 ntdll!_LdrpInitialize+0x78
0b 0012fd10 00000000 ntdll!LdrInitializeThunk+0x10
```

61

## 初始化

- 使用Hook方法截取应用程序对堆函数的调用
  - dds poi(AcXtrnal!NS\_FaultTolerantHeap::g\_pAPIHooks)
- 准备内部数据结构
- FTH: (584): \*\*\* Fault tolerant heap shim applied to current process. This is usually due to previous crashes. \*\*\*

```
0:000> kNL
ChildEBP RetAddr
00 0012f76c 6c837c8a kernel32!OutputDebugStringA
01 0012f78c 6c839565 AcXtrnal!NS_FaultTolerantHeap::FthClientInitialize+0x1cb
02 0012f7a0 6c83ab27 AcXtrnal!NS_FaultTolerantHeap::NotifyFn+0x4d
03 0012f7b8 6c837683 AcXtrnal!NS_FaultTolerantHeap::InitializeHooksMulti+0x79
04 0012f7cc 6c8376ea AcXtrnal!MultiShimEntry+0x4e
05 0012f804 6c8459a2 AcXtrnal!ShimLib::InitializeHooksEx+0x3e
06 0012f824 75c15571 AcXtrnal!ShimLib::GetHookAPIs+0x18
07 0012f938 75c0b7f6 apphelp!SeiInit+0x598
08 0012fb2c 77bcd4c3 apphelp!SE_InstallBeforeInit+0x67
09 0012fb48 77bcd3c7 ntdll!LdrpLoadShimEngine+0xdb
0a 0012fcb0 77c18fc0 ntdll!LdrpInitializeProcess+0x130b
0b 0012fd00 77c0b2c5 ntdll!_LdrpInitialize+0x78
0c 0012fd10 00000000 ntdll!LdrInitializeThunk+0x10
```

62

# demo

## 观察FTH的组件

63

## 补偿调用

- 当堆函数被调用时
- HEAP[HpKiller.exe]: Invalid address specified to RtlFreeHeap( 00320000, 0012FF48 )

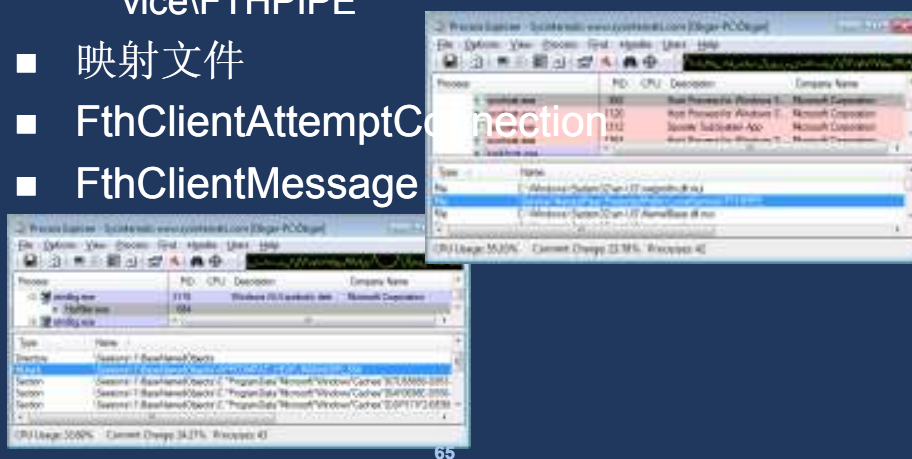
```
0:000> knt
ChildEBP RetAddr
00 0012fd64 77c3fe48 ntdll!RtlpBreakPointHeap+0x23
01 0012fd80 77c756df ntdll!RtlpValidateHeapEntry+0x16d
02 0012fdc8 77c37aca ntdll!RtlDebugFreeHeap+0x9a
03 0012feb0 77c02d68 ntdll!RtlpFreeHeap+0x5d
04 0012fedc 6c839ed9 ntdll!RtlFreeHeap+0x142
05 0012fef4 76f2f1ac AcXtrnal!NS_FaultTolerantHeap::APIHook_RtlFreeHeap+0x61
06 0012ff08 00401ca7 kernel32!HeapFree+0x14
07 0012ff24 0040112a HpKiller!free+0x66
08 0012ff2c 00401048 HpKiller!operator delete+0x9
09 0012ff48 00401240 HpKiller!main+0x48
0a 0012ff88 76f31174 HpKiller!mainCRTStartup+0xb4
0b 0012ff94 77c0b3f5 kernel32!BaseThreadInitThunk+0xe
0c 0012ffd4 77c0b3c8 ntdll!_RtlUserThreadStart+0x70
0d 0012ffec 00000000 ntdll!_RtlUserThreadStart+0x1b
```

64



## 客户端与服务端通信

- 命名管道
  - \Device\NamedPipe\ProtectedPrefix\LocalService\FTHPIPE
- 映射文件
- FthClientAttemptConnection
- FthClientMessage



## FTH Mitigations

| Heap error type            | Mitigated?               | FTH Aware of mitigation? |
|----------------------------|--------------------------|--------------------------|
| Buffer overrun (read)      | Yes (up to 8 bytes)      | No                       |
| Buffer overrun (write)     | Yes (up to 8 bytes)      | Yes                      |
| Double free                | Yes (while free delayed) | Yes                      |
| Reuse after free (read)    | Yes (while free delayed) | No                       |
| Reuse after free (write)   | Yes (while free delayed) | Yes                      |
| Free of stack buffer       | Yes                      | Yes                      |
| Wrong free during shutdown | Yes                      | No                       |
| Free in the wrong heap     | No (NT heap crashes)     | No                       |
| Free of global buffer      | No (NT heap crashes)     | No                       |

# demo

## 理解FTH的工作过程

### 问与答



# Windows 7的新特性

——WDI

张银奎 (Raymond Zhang)

## 可靠性反馈和遥感系统

- Windows Reliability Feedback & Telemetry Systems
- Vista引入
- WER for OS Crashes
- WER for App Crashes and Hang
- AutoBug
- CEIP (Customer Experience Improvement Program)
- Winqual伙伴门户

70

## 遥感

- Telemetry
- 远程测量和报告信息
- Vista引入，在WER的基础上
- 不仅收集崩溃和挂死
- 还收集用户反馈
- 以及很多其它性能、兼容性信息
- Win7进一步完善



71

## 用户反馈

- Beta版本
  - 桌面连接
  - 窗口的标题栏
- RC版本
  - **rundll32.exe**
  - **FeedbackTool.o**
- RTM版本



72

## CEIP

- 数以十万计Vista-SP1用户从世界各地向微软发送实时反馈数据
- 收集丰富信息
  - 启动、休眠
  - 唤醒、关机
  - 崩溃、挂死
  - 安装驱动
  - 资源耗尽
- 计算故障流行



73

## Jon谈收益

- I spend a lot of time on core reliability of the OS and in studying the telemetry we collect from real users (only if they opt-in to the Customer Experience Improvement Program). The telemetry guided us on what to address in SP1.
- Jon DeVaan在E7(Engineering Win7)博客上的文章Organizing the Windows 7 Project

74

## Steve谈收益

- So with all that telemetry, let's take a look at a few of the examples that we gathered through the course of pre-release leading up to general availability of Windows 7. So, let's take a look at some numbers. So, 1.7 million. That's how many times you clicked the "send feedback" button.
- <http://www.microsoft.com/presspass/exec/ssinofsky/2009/11-18PDC.msp>

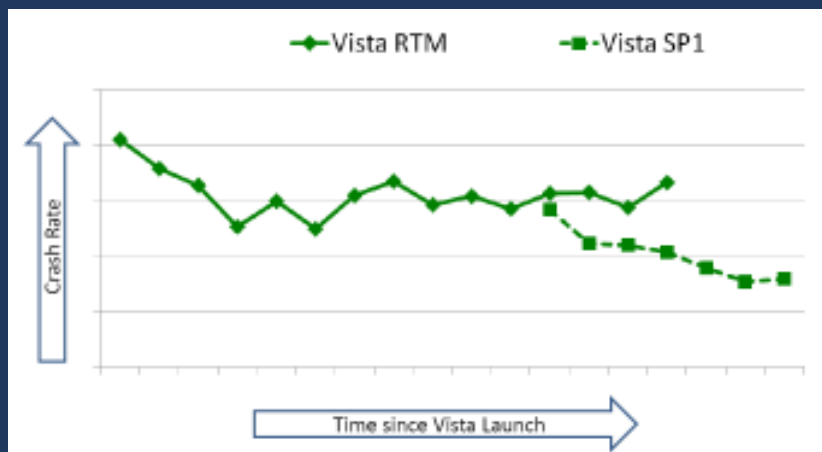
75

## 2008年9月Top 500崩溃原因

| Category              | % Crashes | Notes                                         |
|-----------------------|-----------|-----------------------------------------------|
| Networking            | 12.5%     | Most are power mgmt; fix distribution slow    |
| Display               | 9.4%      | Mostly video card; first time display < 10%   |
| OS Core               | 8.6%      | Kernel is 3.3%; USB is 3.3%                   |
| Application Drivers   | 6.5%      | Antivirus 3.6%; Malware 1.1%; Firewall 0.5%   |
| Hardware              | 6.3%      | 2.7% is general, 2.2% is memory, 1.4% is disk |
| Triage                | 5.7%      | These are not well classified today           |
| Corruption            | 5.1%      | These cannot be well classified               |
| Storage               | 5.0%      | Mostly RAID controllers, some IDE/Atapi       |
| Peripherals           | 2.6%      | Mostly personal media players, now fixed      |
| Imaging               | 1.7%      | Camera drivers, USBvideo                      |
| Streaming Media       | 0.8%      | Third party cameras and TV tuners             |
| Audio                 | 0.6%      | Audio cards and HD drivers                    |
| Input                 | 0.5%      | Third party mice                              |
| Issues Beyond Top 500 | 34.6%     | Haven't looked at many of these               |

76

## 崩溃率



- Vista SP1的稳定性大大改进

77

## 其它规律

- 390000个不同的设备连接到Vista
- 平均每天有25个新的驱动，另加100个更新
- 恶件(Malware)在增长
- CD/DVD烧录、USB和杀毒软件在改进
- 因OEM品牌有很大不同，最好品牌崩溃间隔是最差品牌的30多倍
- 地域差异很大，最好地区的崩溃间隔是最差地区的5倍

78

## 内部实现

- **Windows Diagnostics Infrastructure**
- 内核中一系列Wdi开头的函数
- 建立在ETW机制之上
- **WDI.DLL**输出函数供用户态使用
- **SEM**(Scenario Event Mapper)从注册表中加载定义的情节：
  - 一个启动事件
  - 任意数量的终止事件
  - 任意数量的事件提供者

79

## WDI的架构



- 依赖ETW来传递事件
- 例如EtwEventWriteEndScenario

80



## WDI初始化

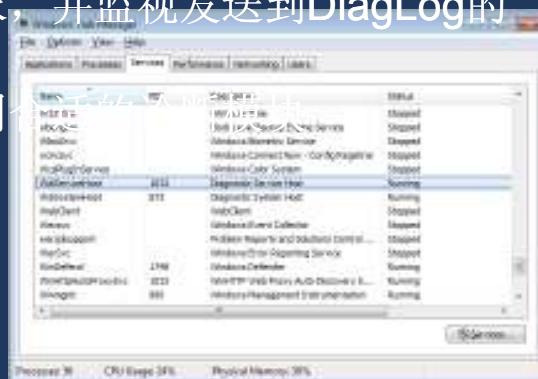
- 执行体阶段1初始化期间
- ETW初始化的一个例子
- WdiInitialize
  - WdipSemAllocate
  - WdipSemEnable
  - WdipSemInitialize
  - WdipSemLoadScenarioTable
  - WdipSemLoadConfigInfo
  - WdipSemUpdateProviderTableWithScenario

```
kd> kn
#(nil)BP RetAddr 4
00 80d8bba0 8318a743 nt!WdiInitialize
01 80d86be4 833b63d2 nt!EtwInitialize+0x254
02 80d86c6c 833ba96a nt!IoInitSystem+0x4e0
03 80d86d48 8319944b nt!Phase1InitializationDiscard+0xcde
04 80d86d50 831fdd16 nt!Phase1Initialization+0xd
05 80d86d50 8300f159 nt!PropSystemThreadStartup+0x9e
06 00000000 00000000 nt!KiThreadStartup+0x19
```

81

## WDI服务

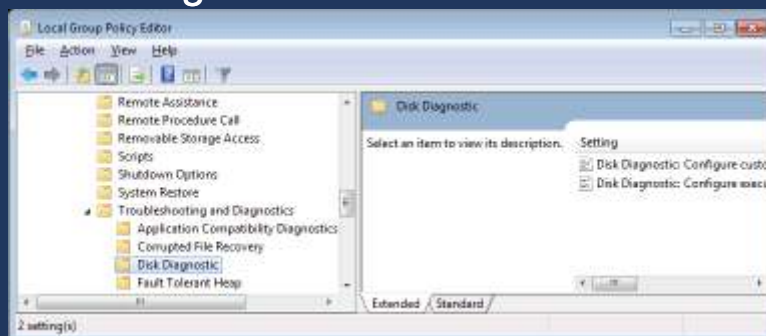
- Diagnostic Policy Service
- System32\DPS.dll
- 随时接收请求，并监视发送到DiagLog的ETW事件
- 根据请求调用



82

## 配置诊断策略

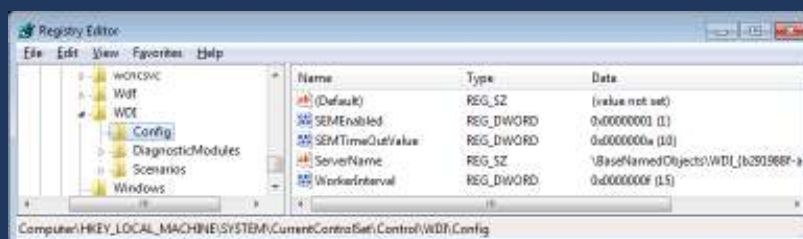
- GPEDIT.MSC
- Computer Configuration > Administrator Templates > System > Troubleshooting and Diagnostic



83

## 注册表

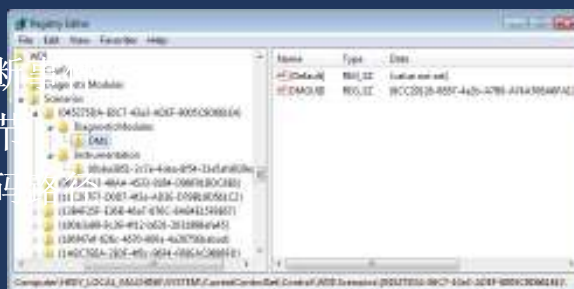
- HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Control\WDI



84

## 情节

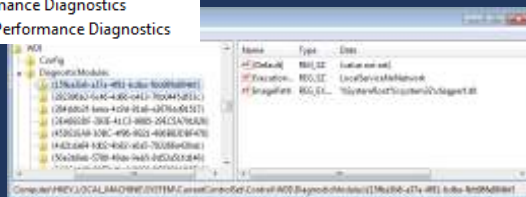
- WDI支持两类触发方式
- 按需 (On-demand)
- 情节(Scenario), 基于ETW事件
- 简单情节
  - 单点触发诊断
- Start-Stop情节
  - 诊断一个代码



85

## 内建的诊断模块

- Troubleshooting and Diagnostics
  - Application Compatibility Diagnostics
  - Corrupted File Recovery
  - Disk Diagnostic
  - Fault Tolerant Heap
  - Microsoft Support Diagnostic Tool
  - MSI Corrupted File Recovery
  - Scheduled Maintenance
  - Scripted Diagnostics
  - Windows Boot Performance Diagnostics
  - Windows Memory Leak Diagnosis
  - Windows Performance PerfTrack
  - Windows Resource Exhaustion Detection and Resolution
  - Windows Shutdown Performance Diagnostics
  - Windows Standby/Resume Performance Diagnostics
  - Windows System Responsiveness Performance Diagnostics



86

demo

```
kd> km 100
ChildEBP RetAddr
00 8cf87cc4 8327aa7f ntl!WdiDispatchControl
01 8cf87d14 8304b79a ntl!NtTraceControl+0x24a
02 8cf87d14 779f64f4 ntl!KiFastCallEntry+0x12a
03 0086f620 779f5d5c ntdll!KiFastSystemCallRet
04 0086f624 779b53f1 ntdll!ZwTraceControl+0xc
05 0086f68c 0050d9c4 ntdll!EvtEventWriteEndScenario+0x9e
06 0086f6d4 0050da22 winlogon!WLEventWriteStartStopScenario+0x89
07 0086f730 00502109 winlogon!WLGeneric_Welcome_Execute+0xb0
08 0086f748 779c6cf5 winlogon!StateMachineWorkerCallback+0x67
09 0086f76c 779de8d1 ntdll!TppWorkerExecuteCallback+0x10f
0a 0086f8cc 76741174 ntdll!TppWorkerThread+0x572
0b 0086f8d8 77a0b3f5 kernel32!BaseThreadInitThunk+0xe
0c 0086f918 77a0b3c8 ntdll!_RtlUserThreadStart+0x70
0d 0086f930 00000000 ntdll!_RtlUserThreadStart+0x1b
```

理解WDI的工作原理

87



# Windows 7的新特性

——安全方面的改进

张银奎 (Raymond Zhang)

"If you look at our investment in the next version of Windows, security would jump out as the thing we've spent the most time on."

--Bill Gates, keynote on 2006 RSA Conference

## NT 3.1的安全

- “At the outset of the NT project, Cutler treated security as an after-thought, another item on a long list of features.” – Showstopper, p144
- Gary Kimura最初负责安全，但是主要精力在FS，无暇顾及
- 1990年1月，Cutler请Jim Kelly加入NT团队负责安全
- “Kelly immediately found himself in a hole. ‘Security was Johnny-come-lately to NT,’ he realized.” –Showstopper, p144
- 正确的态度看待历史

## 安全努力

- Windows Server 2003 全面重视安全问题
- Windows XP SP2, Windows历史上最大的Patch工程，不仅是改正现用功能的瑕疵，而且加入了包括防火墙和DEP (Data Execution Prevention) 在内的很多新功能，用于提高系统的安全性
- Visual Studio .Net 2003引入了名为GS Switch的技术，可以自动向可能发生缓冲区溢出的函数加入保护性代码，该技术被用于编译Windows Server 2003。VS 2005进一步加强了这一技术，被用于编译Windows XP SP2

## Vista的安全增强

- Windows Vista, 从里到外, 脱胎换骨, .....把安全灌输和渗透到系统的每个部件, 包括用户态模块和内核模块
- Session 0隔离
- User Account Control
- Windows Defender (AntiSpyware)
- Address Space Layout Randomization (ASLR)
- Digital Rights Management
- PatchGuard
- Code Integrity .....

## ASLR

- 攻击代码调用或通过缓冲区溢出返回到预先分析出的地址, 或者破坏某个地址的数据。
- ASLR (Address Space Layout Randomization) 技术将模块的加载地址随机化, 使函数入口不再位于固定的地址。
- 《Vista部署ASLR安全技术 微软与黑客捉迷藏》——ZDNet China
- 可以有效地阻止包括return-to-libc在内的各种缓冲区溢出攻击。
- ASLR技术也被用于OpenBSD, Exec Shield for Linux等软件。

## PatchGuard

- 内核是系统的管理者，具有特权，但是内核态的驱动程序是内核的信赖代码，也具有同样的特权级别
- 堡垒最容易从内部被攻破，内核态的Malware比用户态的更具威胁
- Rootkit!!!
- IDT表，内核服务表（SSDT），MSR寄存器，或内核函数入口都是Rootkit攻击的目标
- PatchGuard对重要内核数据进行加密保护（PgEncryptContext），防止被篡改。
- 只用于x64系统

## 回顾UAC

- 核心问题：进程以不必要的高权限运行
  - 执行难以预知的特权行为
- 解决之道：让进程以低权限运行
  - 需要高权限时，再提升 (Elevate)
- 名称演变
  - LUA (Limited User Account), UAP (User Account Protection), 最终User Account Control (UAC)
  - Luafv.sys依然保持着最初名称



## UAC用户类型

- Standard User
  - “Users” 组成员
- Consent Admin
  - “Local administrators” 组成员
  - 以“filtered”标准用户登录
  - 可能提升到管理员权限
- 管理员用户登录时，创建两个访问令牌(Token)
  - Full Token
  - Filtered Token



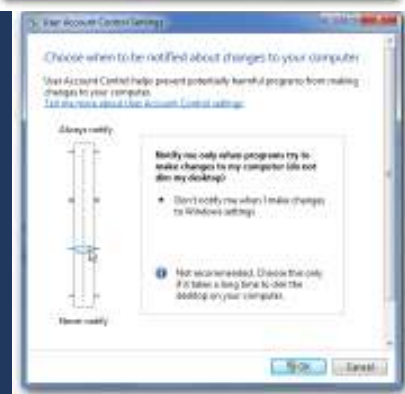
```

BOOL WINAPI CreateRestrictedToken(
 _in HANDLE ExistingTokenHandle,
 _in DWORD Flags,
 _in DWORD DisableSidCount,
 _in_opt PSID_AND_ATTRIBUTES SidsToDisable,
 _in DWORD DeletePrivilegeCount,
 _in_opt PLUID_AND_ATTRIBUTES PrivilegesToDelete,
 _in DWORD RestrictedSidCount,
 _in_opt PSID_AND_ATTRIBUTES SidsToRestrict,
 _out PHANDLE NewTokenHandle
)

```

## 调节UAC

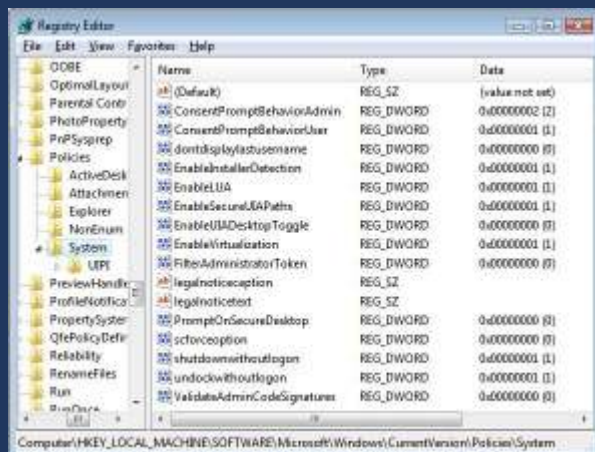
- Vista
  - Control Panel > uac
  - 关闭/打开
- Win7
  - Start Menu > uac
  - 调节
- 都可以通过LSP配置
- Secpol.msc



98

## 注册表

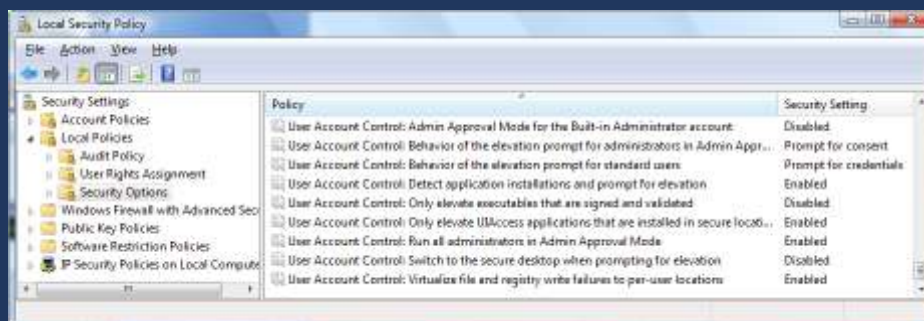
### ■ 注册表中的UAC和系统选项



99

## 配置UAC

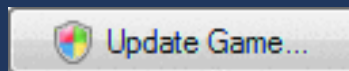
### ■ Secpol.msc启动本地安全策略



100

## 提升特权

- 启动时不需要提升
- 做个别操作前提升
- Button\_SetElevationRequiredState(hwndButton, true);
- 创建新的进程
- 清单文件
  - 单独文件
  - 放入资源

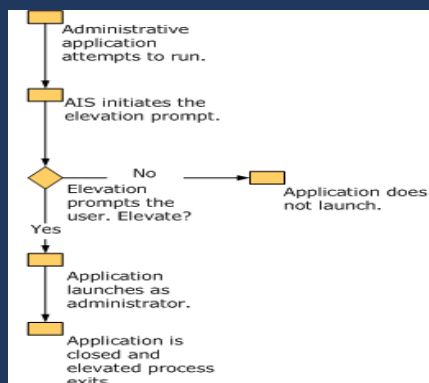


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1"
manifestVersion="1.0">
 <assemblyIdentity version="1.0.0.0"
processorArchitecture="X86" name="AdminApp" type="win32"/>
 <description>Description of your application</description>
 <!-- Identify the application security requirements -->
 <ms_asmv2:trustInfo xmlns:ms_asmv2="urn:schemas-microsoft-com:asm.v2">
 <ms_asmv2:security>
 <ms_asmv2:requestedPrivileges>
 <ms_asmv2:requestedExecutionLevel
level="requireAdministrator"
uiAccess="false"/>
 </ms_asmv2:requestedPrivileges>
 </ms_asmv2:security>
 </ms_asmv2:trustInfo>
</assembly>
```

101

## AIS服务

- Application Information Service
- 如果停止，需要提升权限的程序无法运行



102

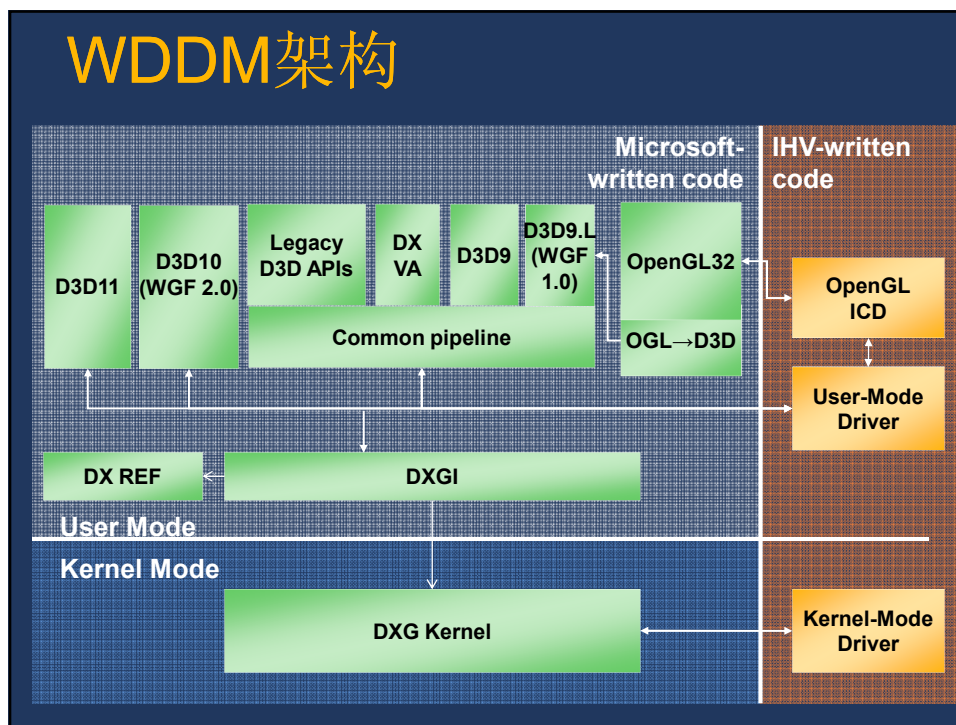


# Windows 7的新特性

——图形概览

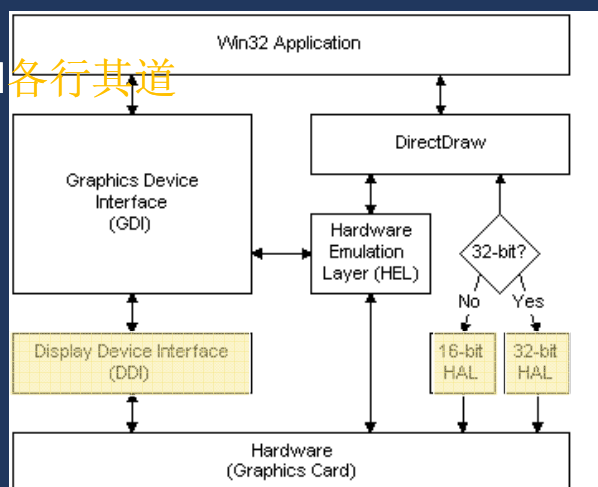
张银奎 (Raymond Zhang)

## WDDM架构



## XPDM

- Windows XP和Windows 2000所使用的显示模型
- DX与GDI各行其道



106

## GDI内核支持

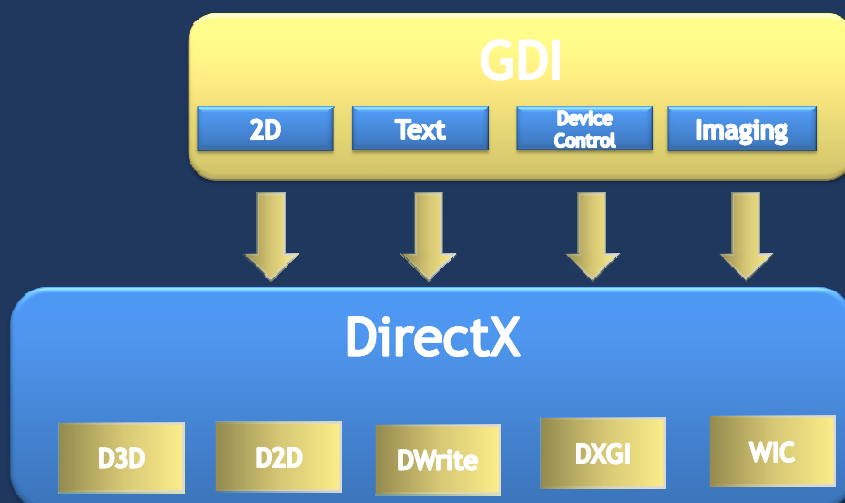
- 曾经是显卡驱动的主要功能，XPDM
  - DrvXXX, *DrvBitBlt()*, ...
  - WDK中仍有详细文档
- WDDM中由CDD映射到D3D
- DirectDraw淡出

**DirectDraw** is no longer recommended for use. With the release of Direct3D 9.0, all two-dimensional functionality is contained within Direct3D and its associated helper functions in D3DX. However, the DirectDraw documentation is still available, and it can be viewed by searching in the [MSDN Library Archive](#).

```
cdd!DrvBitBlt = <no type information>
cdd!DrvCreateDeviceBitmapEx = <no type information>
cdd!DrvStretchBlt = <no type information>
cdd!DrvAssertMode = <no type information>
cdd!DrvDisableDriver = <no type information>
cdd!DrvUnlockDisplayArea = <no type information>
cdd!DrvEnablePDEV = <no type information>
cdd!DrvTextOut = <no type information>
cdd!DrvAssertModeInternal = <no type information>
cdd!DrvEnableDriver = <no type information>
cdd!DrvStartDxInterop = <no type information>
cdd!DrvIcmSetDeviceGammaRamp = <no type information>
cdd!DrvDisableSurface = <no type information>
cdd!DrvDeleteDeviceBitmap = <no type information>
cdd!DrvTransparentBlt = <no type information>
cdd!DrvSynchronizeSurface = <no type information>
cdd!DrvAccumulateD3DDirtyRect = <no type information>
cdd!DrvSetPalette = <no type information>
cdd!DrvCreateDeviceBitmap = <no type information>
cdd!DrvLineTo = <no type information>
cdd!DrvMovePointer = <no type information>
cdd!DrvAssociateSharedSurface = <no type information>
cdd!DrvStretchBltROP = <no type information>
cdd!DrvSynchronizeRedirectionBitmaps = <no type information>
cdd!DrvPointerShape = <no type information>
CloseLocalGraphicsDevices
SetDisplayConfigValidateParameters
win32k!DrvGetDisplayDriverDpiSetting
win32k!DrvSetMonitorPowerState = <no type information>
win32k!DrvOpenLocalGraphicsDevices = <no type information>
win32k!DrvAddAdapterLuid = <no type information>
win32k!DrvGetRegistryHandleFromDevice = <no type information>
win32k!DrvProcessMonitorEventCallback = <no type information>
win32k!DrvCheckProcessSettings = <no type information>
```

107

## GDI蓝图



\*来自DirectX, Core Graphics For Windows 7, WinHec 2008

108

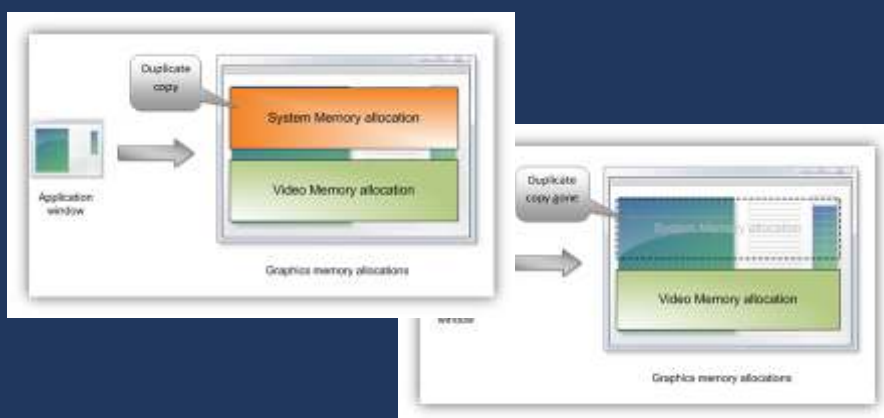
## GDI执行示例

- 通过GDI32.dll保持应用程序兼容
- DX SDK的SampleBrowser程序
- “运行在DX11系统中的所有窗口程序都是D3D程序”

```
CH10EBP RetAddr
00 aef112f4 95db1d09 cdd!copySurfBitsFast+0xa2
01 aef11350 95dc33bb cdd!copySurfBits+0x81
02 aef116dc 95dc3d74 cdd!BitBltBitMap+0x3eb
03 aef11890 95dc482f cdd!DrvBitBlt+0xc2
04 aef118c4 95bf8e27 cdd!DrvCopyBits+0x27
05 aef1190c 95bdeac4 win32k!OffCopyBits+0x7d
06 aef11bb0 95bfc401 win32k!Sp8BitBlt+0x252
07 aef11be4 95bfc533 win32k!SpCopyBits+0x27
08 aef11ccc 95be4e36 win32k!NtGdiBitBltInternal+0x6ab
09 aef11d00 83c8b44a win32k!NtGdiBitBlt+0x2f
0a aef11d00 77c664f4 nt!KiFastCallEntry+0x12a
0b 0006b674 761f7209 ntdll!KiFastSystemCallRet
0c 0006b678 761f71f1 GDI32!NtGdiBitBlt+0xc
0d 0006b6bc 6886b44f GDI32!BitBlt+0x1fa
0e 0006b6f4 6886b85a mshtml!CDispSurface::Draw+0x43
...
4d 0006f620 77823578 USER32!DispatchMessageWorker+0x35e
4e 0006f630 696053c4 USER32!DispatchMessageA+0xf
4f 0006f640 6961eb1a MFC42!CWinThread::PumpMessage+0x42
50 0006f660 69631640 MFC42!CWnd::RunModalLoop+0x6
51 0006f6a4 0046399d MFC42!CDialog::DoModal+0x11e
52 0006f9c0 69603493 SampleBrowser!CSampleBrowserApp::InitInstance+0x184
53 0006f9d4 00469cb8 MFC42!AFXWinMain+0x4f
...
```

109

## Win7对GDI的改进



- 参考Engineering Windows 7 Graphics Performance

110

# demo

## 使用调试器浏览图形世界

111

### 问与答





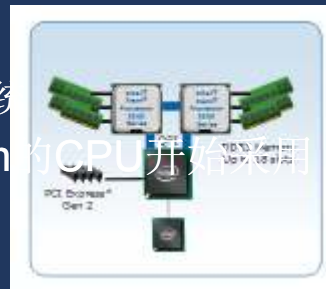
# Windows 7的新特性

## ——其它改进

张银奎 (Raymond Zhang)

## NUMA

- Non-Uniform Memory Access
- 每个CPU有自己的内存（local memory）
- 一个CPU也可以访问其它CPU的内存，但是访问速度要比访问自己的内存慢很多
- 相对的UMA架构
  - 目前正在使用的大多x86系统
- IA架构从代号为Nehalem的CPU开始使用NUMA



## Win7的NUMA设施

- 新增内核函数
  - IoGetDeviceNumaNode
  - KiNonNumaDistance
  - KiNonNumaQueryNodeCapacity
  - KiNonNumaQueryNodeDistance
  - KiNumaQueryProximityNodeEx
  - MiInitializeNumaMemoryNodeEx
  - MiNonNumaPageFaultEx
- API
  - GetNumaAvailableMemoryNodeEx
  - GetNumaNodeNumberFromHandle
  - GetNumaNodeProcessorMaskEx
  - GetNumaProcessorNodeEx
  - GetNumaProximityNodeEx
  - GetActiveProcessorCount
  - GetActiveProcessorGroupCount
  - GetCurrentProcessorNumberEx

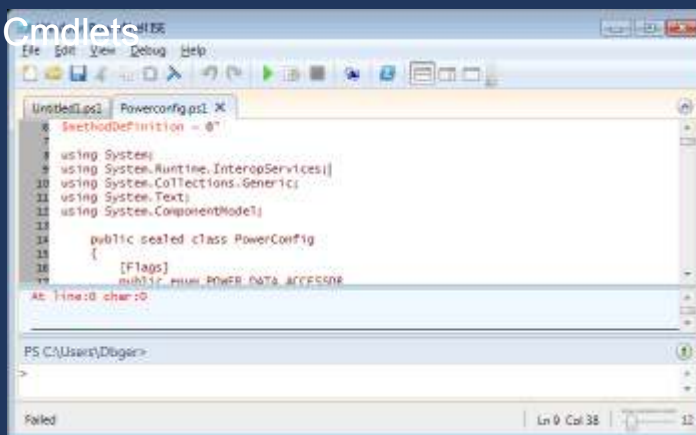
115

## 进程反射

- *Process Reflection*
- 克隆进程
- 用于诊断进程，为其产生转储
  - 泄露检查
  - 跨进程的挂死检测
- 减少对原来进程的影响
- NtDll!RtlCreateProcessReflection
  - ZwCreateSection
  - ZwDuplicateObject
- 模仿fork()

## PowerShell 2.0

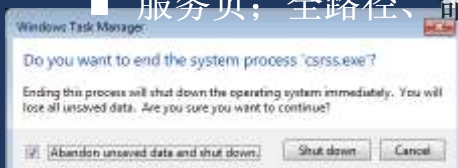
- 集成开发环境——ISE(Integrated Scripting Environment)
- 500+ Cmdlets



117

## 任务管理器

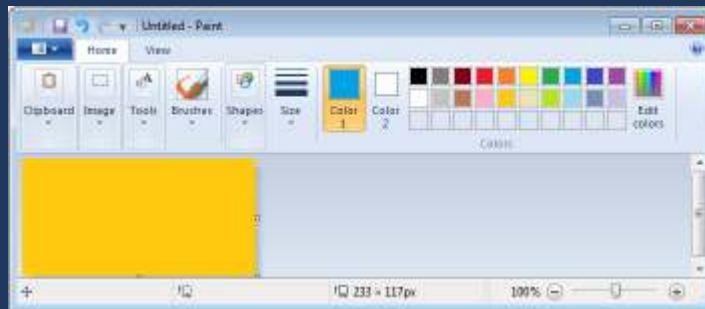
- 随时为进程“拍照”
- 杀掉关键进程
  - 以管理员身份运行
  - 有警告
  - 我的电脑我做主!
- 更多信息
  - 服务页; 全路径、命令行、UAC、描述



118

## Scenic Ribbon

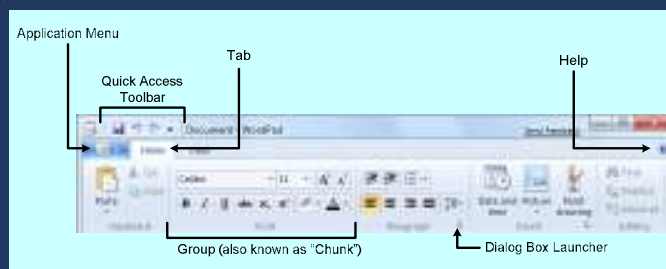
- Office 2007风格的界面
- Win7的自带小程序普遍采用
- 公开SDK供ISV使用



119

# demo

## 编写使用Ribbon UI的程序



120

## Superbar

- 改进的任务条
- 图形化的Thumbnail预览
- 加大的图标
- 横条右端Show Desktop
- 图标层叠显示程序实例个数
- 减少显示出的Tray Icon



121

## 虚拟化支持

- 内核中引入一系列Hv开头的符号
- 内核初始化早期会调用HvInitSystem
  - 将调试需要的数据注册转储
    - IoAddTriageDumpDataBlock
  - 调用CPUID指令检测CPU支持
- 检测结构保存在nt!HvlpFlags 变量中
- 调用HvlpTryConnectHypervisor尝试与在VMM

|                    |
|--------------------|
| hvlp.obj(0)        |
| hvlb.obj(0)        |
| hvlpower.obj(0)    |
| hvlmain.obj(0)     |
| hvldata.obj(0)     |
| hvlmisc.obj(0)     |
| hvlconnect.obj(0)  |
| hvlbrighten.obj(0) |
| hvlproc.obj(0)     |
| hvlhal.obj(0)      |
| hvlcall.obj(0)     |

```
kd> kn
ChildEBP RetAddr
00 83122ca0 833123de nt!HvInitSystem
01 83122d0c 83119801 nt!KiInitializeKernel+0xa3
02 00000000 f000eef3 nt!KiSystemStartup+0x329
WARNING: Frame IP not in any known module. Follow
03 00000000 00000000 0xf000eef3
```

122

# demo

## 使用调试器浏览Win7的新特征

123

## 接收手势消息

- WM\_GESTURE
- GetGestureInfo()

```
case WM_GESTURE:
 // Get all the vertical scroll bar information
 si.cbSize = sizeof (si);
 si.fMask = SIF_ALL;
 GetScrollInfo (hWnd, SB_VERT, &si);
 yPos = si.nPos;

 ZeroMemory(&gi, sizeof(GESTUREINFO));
 gi.cbSize = sizeof(GESTUREINFO);
 bResult = GetGestureInfo((HGESTUREINFO)lParam, &gi);

 if (bResult){
 // now interpret the gesture
 switch (gi.dwID){
 case GID_BEGIN:
 lastY = gi.ptsLocation.y;
 CloseGestureInfoHandle((HGESTUREINFO)lParam);
 break;
 // A CUSTOM PAN HANDLER
 // COMMENT THIS CASE OUT TO ENABLE DEFAULT HANDLER BEHAV
 case GID_PAN:
```

124

## 接收触控消息

- WM\_TOUCH
- RegisterTouchWindow(hWnd, 0);
- GetTouchInputInfo()

```
case WM_TOUCH:
 cInputs = LOWORD(wParam);
 pInputs = new TOUCHINPUT[cInputs];
 if (pInputs)
 {
 if (GetTouchInputInfo((HTOUCHINPUT)lParam, cInputs, pInputs, si))
 {
 for (int i=0; i < static_cast<INT>(cInputs); i++){
 TOUCHINPUT ti = pInputs[i];
 index = GetContactIndex(ti.dwID);
 if (ti.dwID != 0 && index < MAXPOINTS){
 // Do something with your touch input handle
 ptInput.x = TOUCH_COORD_TO_PIXEL(ti.x);
 ptInput.y = TOUCH_COORD_TO_PIXEL(ti.y);
 ScreenToClient(hWnd, &ptInput);

 if (ti.dwFlags & TOUCHEVENTF_UP){
 points[index][0] = -1;
 points[index][1] = -1;
 }else{
 points[index][0] = ptInput.x;
 points[index][1] = ptInput.y;
 }
 }
 }
 }
 }
}
```

125

## 问与答

