

# WinDbg 内核调试配置

内核调试主要用来调试驱动代码、分析内核结构等。WinDbg 通过两台电脑可以实现内核调试，其中一台电脑运行 WinDbg，被称为主机；另外一台电脑运行被调试的程序或系统，被称为目标机。一般情况下两台电脑都是真实机器，这样调试最符合实际情况，两台电脑通过串口线、1394 线或 USB 对连线连接起来实现双机内核调试。如果没有两台电脑，也可以用虚拟机来模拟目标机，主机上运行 WinDbg，虚拟机中安装 Windows 运行被调试的程序，虚拟机通过模拟的串口输出为主机上的一个命名管道，从而和主机上的 WinDbg 连接起来实现双机内核调试。

除双机内核调试外，WindowsXP 后还引入了一种本机内核调试方式，只需要一台电脑，直接运行 WinDbg 就能查看修改系统内核结构等，不过所有和中断目标机系统相关的命令都不能执行，如断点命令。

如果采用虚拟机模拟目标机，调试响应速度有时候是个问题，比 1394 线连接的真实双机调试速度要慢不少，特别是执行操作大量内存的命令时（如搜索内存命令），感觉非常明星。所以针对这种情况还会介绍一个特殊的辅助调试工具 vmkd，该工具可以大大加速内核调试的速度，为咱们带来不少方便。

下面分节详细介绍各种内核调试情景下的配置，尽量每个步骤都截图说明。

## 真实机双机内核调试

真实机之间的内核调试首先需要准备连接线，可以用串口线、1394 线或者 USB 对连线。

串口线速度太慢，而且电脑城一般买不到可以直接使用的串口线，需要把线和接头买回来自己焊，按照 WinDbg 帮助中的说明交叉焊接，就能得到一根可用来调试的串口线。用串口线把两台电脑连接上后，先用 Windows 自带的超级终端工具，选择好串口和波特率连接。如果在超级终端中按键能在另外一台电脑的超级终端上显示按键，则表示串口线连接成功。接下来就可以用 WinDbg 连接串口调试。某些笔记本上可能没有串口，可以买一个 USB 转串口的接头，然后设置 USB 转换后的串口号，就能把这台笔记本当作主机使用。

1394 线速度快，价格也便宜，如果电脑上没有 1394 口，可以再另外买一个 1394 卡，价格也很便宜。1394 分大口和小口，只需按照电脑上的接口大小购买合适的线就行。如果没有 1394 口，装一个 1394 卡又很麻烦，则也可以买一个 USB 转 1394 口的接头，不过一样只能当作主机使用。

USB 对连线是 Vista 系统以后支持的内核调试连接方式，没见到哪里能买到这样的线，估计速度会更快一点吧！

因为 1394 线连接调试最方便，速度也比较快，所以建议使用这种方式。

电脑、连接线等准备好后，先设置目标机系统，启用内嵌在系统中的内核调试引擎。Vista 之前的系统，用记事本打开系统盘根目录下的 boot.ini 文件，添加新的启动项，在新的启动项上添加调试选项。如下表，红色行表示新加的启动项，/debug 表示打开内核调试引擎，/debugport=1394 表示采用 1394 连接方式，/channel=10 表示设置通道号为 10。

```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP Professional"
/noexecute=optin /fastdetect
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="WINXP - Debug" /noexecute=optin /debug
/debugport=1394 /channel=10
```

Vista 之后的系统需要修改 BCD 数据库，利用 bcdedit 工具添加启动项，设置调试选项。如下表，以管理员方式

运行命令程序，先复制当前项生成新的启动项，然后在新的启动项上操作，打开内核调试、设置调试连接方式、设置 1394 通道号。

```
C:\>bcdedit /copy {default} /d "Vista - Remote Debug"
```

已将该项成功复制到 {13fbbcdc-756a-11dc-aed8-0016e68bceb3}。

```
C:\>bcdedit /set {13fbbcdc-756a-11dc-aed8-0016e68bceb3} debugtype 1394
```

操作成功完成。

```
C:\>bcdedit /set {13fbbcdc-756a-11dc-aed8-0016e68bceb3} channel 10
```

操作成功完成。

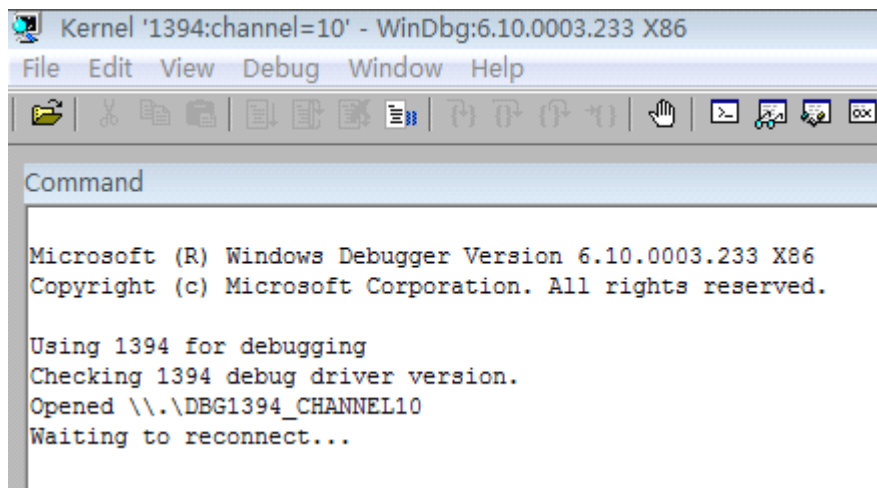
```
C:\>bcdedit /debug {13fbbcdc-756a-11dc-aed8-0016e68bceb3} on
```

操作成功完成。

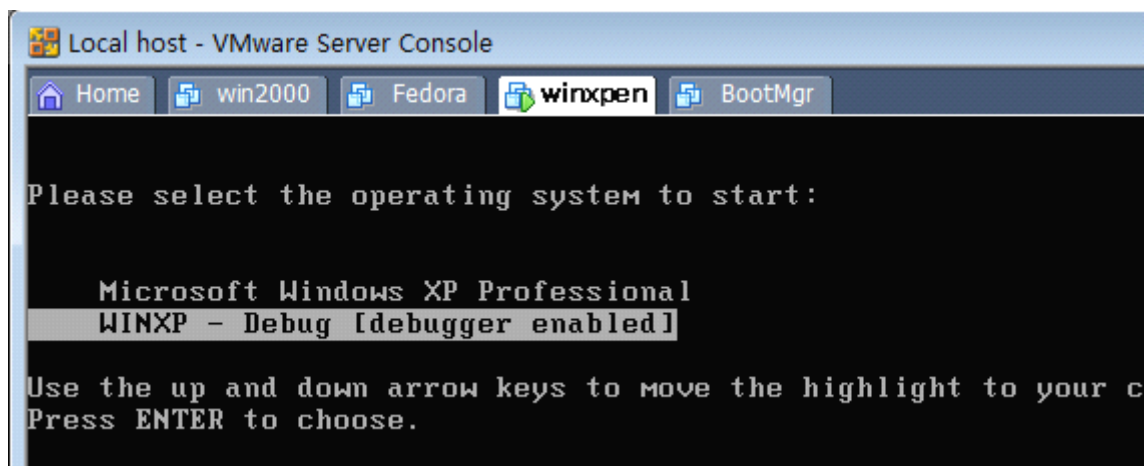
设置完目标系统后重启，在选择启动菜单时停下来，在主机上通过如下命令行运行 WinDbg 准备连接到目标机。

```
D:\WinDbg>windbg -d -k 1394:channel=10
```

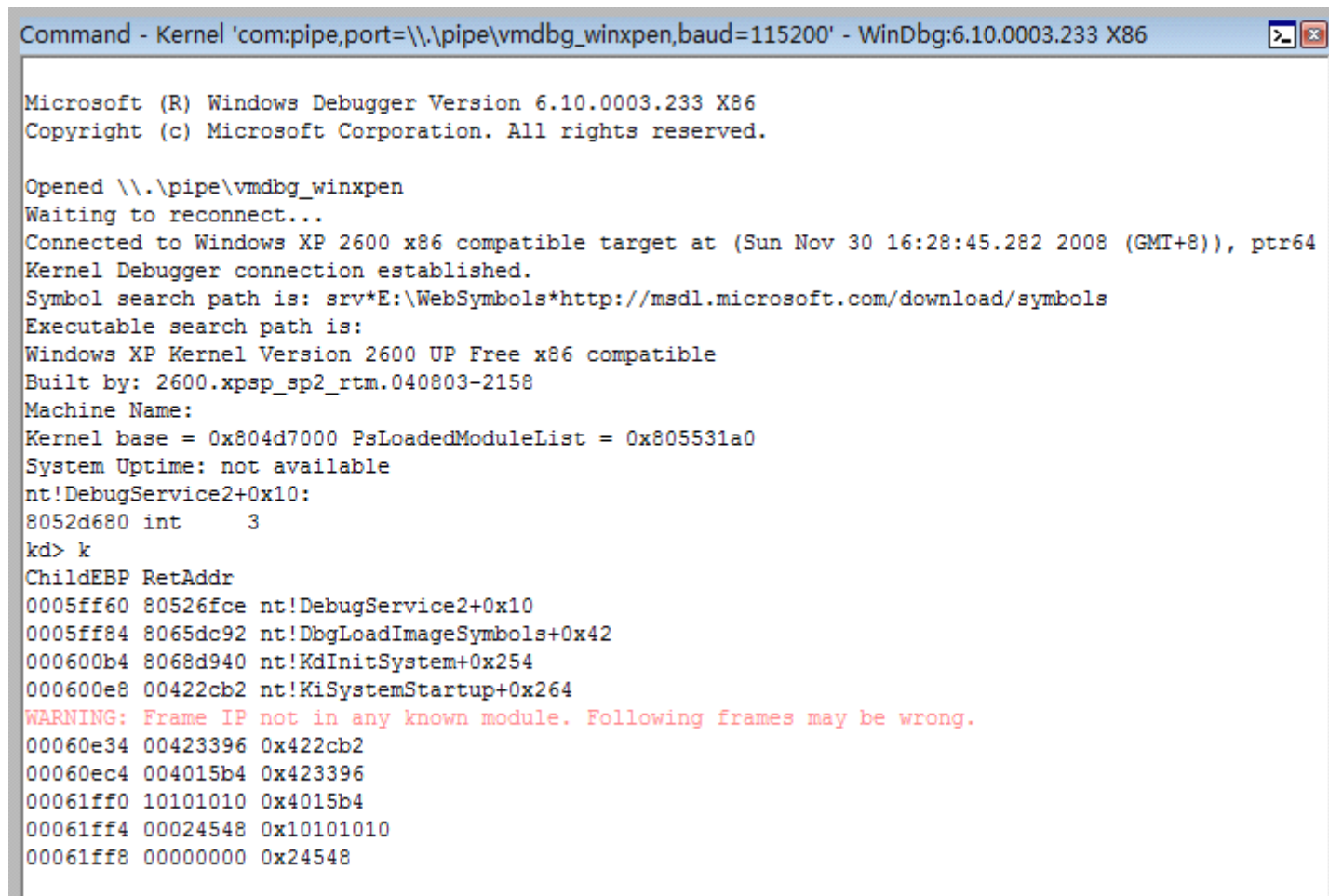
-d 表示在连接上目标系统后马上中断下来，这个也是最早的断点，该选项主要是为了在目标系统初始化早期就断下来查看系统状态，当前情况下可以不要，1394:channel=10 表示连接方式和通道号。第一次采用 1394 内核调试时，WinDbg 可能会显示启动驱动失败，再次运行 WinDbg 应该就好了。运行 WinDbg 后，显示如下界面：



表示 WinDbg 已经准备好，正在等待 1394 连接。然后返回目标机，选择前面添加的启动项（也就是打开了内核调试选项的启动项），如下图，真实机器无法截图，这里用虚拟机截图代替。




选择该启动项后，系统就会加载内核，初始化内核调试引擎，不一会（几秒钟）就能在主机的 WinDbg 上看到输出信息，接着中断在 WinDbg 中，如下图（写这篇文章时还是用的虚拟机截图，呵呵）：



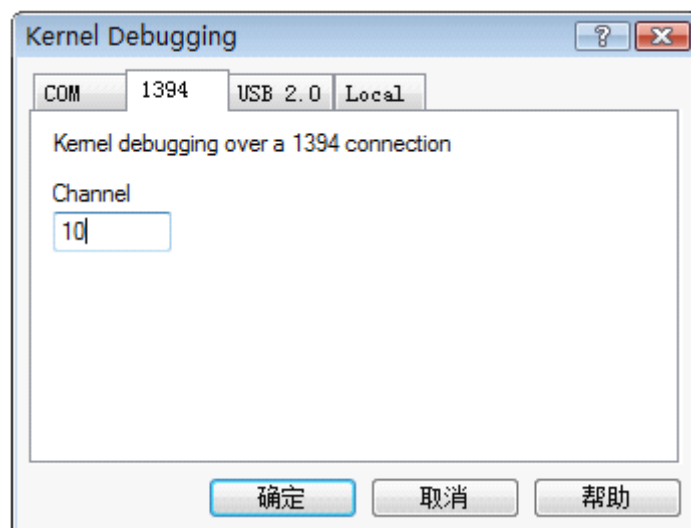
```
Command - Kernel 'com:pipe,port=\\.\pipe\vmdbg_winxpen,baud=115200' - WinDbg:6.10.0003.233 X86

Microsoft (R) Windows Debugger Version 6.10.0003.233 X86
Copyright (c) Microsoft Corporation. All rights reserved.

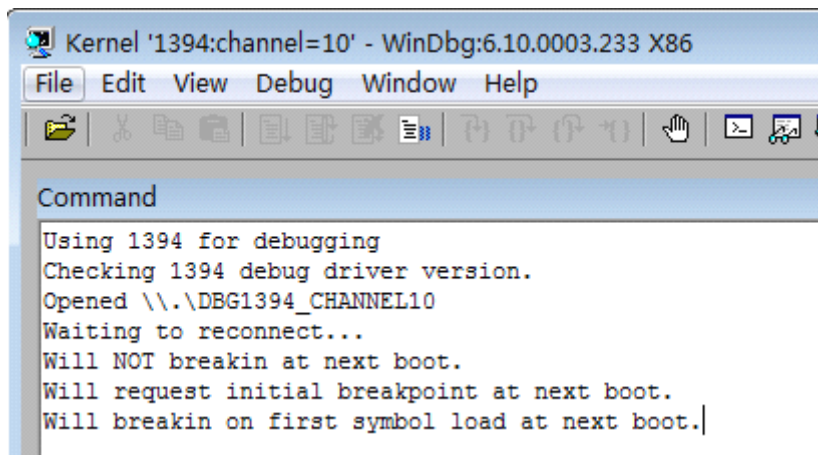
Opened \\.\pipe\vmdbg_winxpen
Waiting to reconnect...
Connected to Windows XP 2600 x86 compatible target at (Sun Nov 30 16:28:45.282 2008 (GMT+8)), ptr64
Kernel Debugger connection established.
Symbol search path is: srv*E:\WebSymbols*http://msdl.microsoft.com/download/symbols
Executable search path is:
Windows XP Kernel Version 2600 UP Free x86 compatible
Built by: 2600.xpsp_sp2_rtm.040803-2158
Machine Name:
Kernel base = 0x804d7000 PsLoadedModuleList = 0x805531a0
System Uptime: not available
nt!DebugService2+0x10:
8052d680 int      3
kd> k
ChildEBP RetAddr
0005ff60 80526fce nt!DebugService2+0x10
0005ff84 8065dc92 nt!DbgLoadImageSymbols+0x42
000600b4 8068d940 nt!KdInitSystem+0x254
000600e8 00422cb2 nt!KiSystemStartup+0x264
WARNING: Frame IP not in any known module. Following frames may be wrong.
00060e34 00423396 0x422cb2
00060ec4 004015b4 0x423396
00061ff0 10101010 0x4015b4
00061ff4 00024548 0x10101010
00061ff8 00000000 0x24548
```

可以看到 WinDbg 显示已经连接到 WindowsXP 系统上，并显示了符号路径、内核基址等信息，然后中断下来得到控制权，输入 k 命令可以看到当前调用栈，输入 g 命令让目标系统继续启动。开始调试后，在 WinDbg 中可以随时按 Ctrl+Break 组合键或通过工具栏  按钮中断目标系统。

如果前面运行 WinDbg 时不习惯命令行方式，也可以先打开 WinDbg，然后选择菜单 File/Kernel Debug 打开内核调试连接对话框，切换到第二个“1394”选项卡，在 channel 下面的编辑框中输入通道号 10，点击“确定”按钮开始连接目标机。



这样和前面命令行运行 WinDbg 的效果一样，就是没了 -d 选项，可以在等待连接的界面上按两次 **Ctrl+Alt+k** 组合键达到相同的效果。



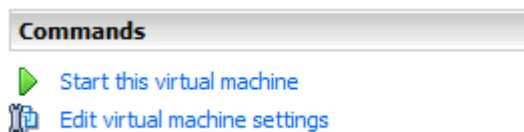
真实机采用 1394 调试时，如果目标机系统是 Windows 2003（未安装 Service Pack）或者 Windows XP SP1，则连接调试前，需要先在目标机上的“设备管理器”中禁用 1394 控制器。如果目标机上是 Vista、Windows 2003 SP1 或者 Windows XP SP2 等系统之后的系统，则不要禁用 1394 控制器。另外，如果主机是 Vista 之前的系统，用 1394 调试时连接有问题，则可以尝试在主机的“设备管理器”中禁用 1394 网络适配器。

## 虚拟机双机内核调试

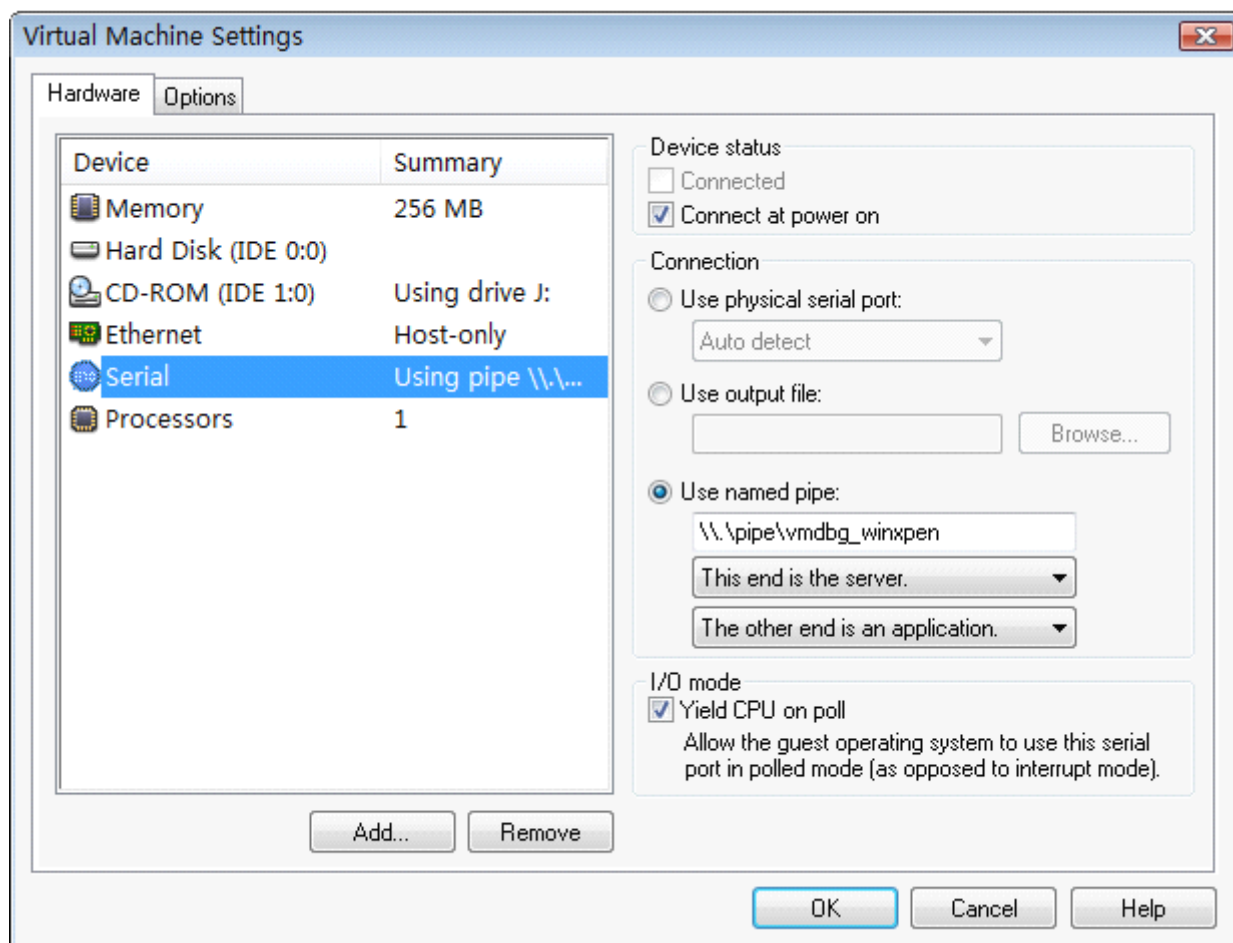
真实机毕竟硬件成本高，需要两台电脑，还得买连接线。如果调试要求不高，很多情况下可以用虚拟机模拟目标机，实现双机内核调试。当前常用的虚拟机有两种：VMWare 和 VirtualPC，都提供了免费版本，运行速度相当于主机的一半左右。估计现在大家都是双核的电脑，运行虚拟机还是没问题的。VMWare 似乎速度要快一点点，而且提供的快照功能非常方便，可以随时把系统还原到以前保存的状态。VirtualPC 似乎兼容性要好一点，有时候在 VMWare 上安装系统蓝屏，用 VirtualPC 就没问题。

首先也是选择连接方式，用虚拟机只有一种串口方式，通过虚拟机模拟的串口输出到主机上的命名管道，然后 WinDbg 连接这个命名管道，从而实现主机和虚拟机的连接。

在虚拟机中安装好系统后，关闭虚拟机系统，打开虚拟机系统的设置框，VMWare 中如下图：

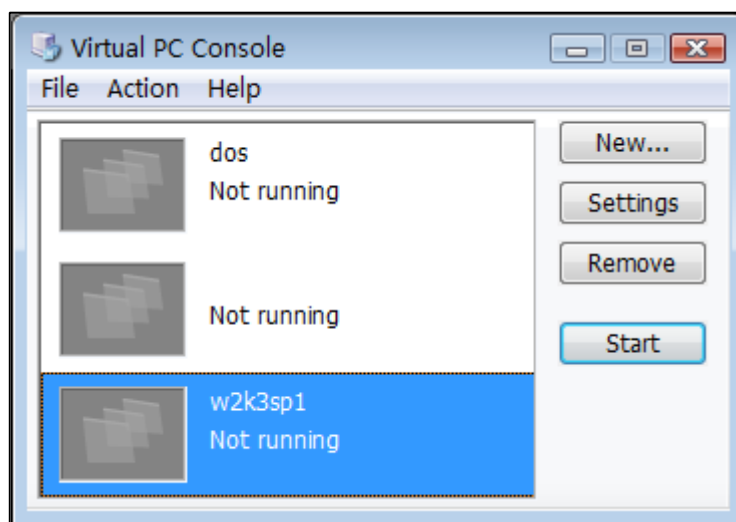


选择 Edit virtual machine settings，打开设置对话框：



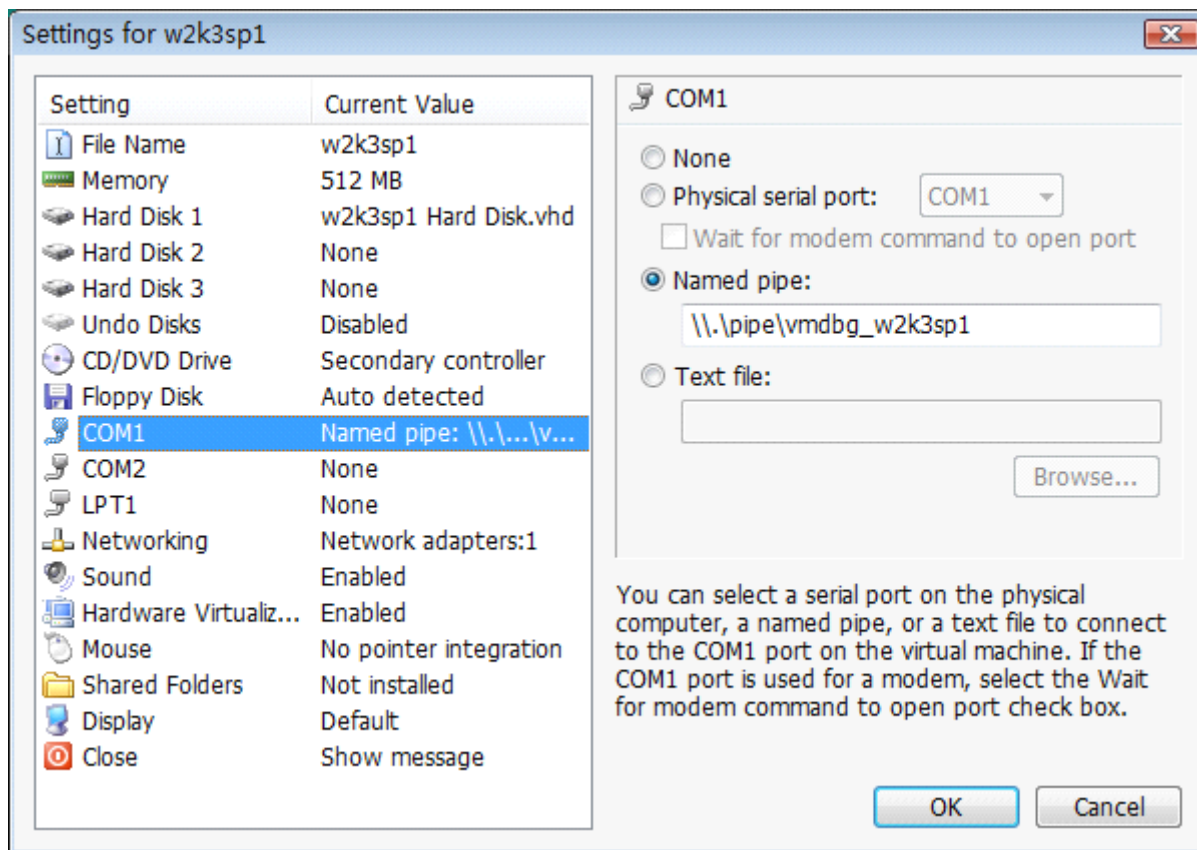
首先选择 **Add...** 按钮添加 **Serial** 设备，然后按照上图设置 **Serial** 属性。命名管道名称为 **WinDbg** 连接时需要用到的管道名，`\\.\pipe\` 前缀不可少，后面接一个容易理解的名称即可。

VirtualPC 虚拟机也差不多，先安装好系统，选择系统，点击 **Settings** 按钮：



打开虚拟机系统设置对话框，选择左边的 **COM1** 项，在右边设置输出的命名管道名称。





设置好硬件连接方式后（这里是虚拟硬件），启动虚拟机中的系统，添加调试启动项。因为虚拟机只是虚拟了串口，所以设置系统启动项时只能使用串口方式。**Vista** 之前的系统通过修改 `boot.ini` 文件实现，如下表。红色项表示新添加的启动项，`/debug` 表示打开内核调试引擎，`/debugport=com1` 表示采用串口 1 通信，`/baudrate=115200` 设置串口 1 的波特率为 115200。

```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP Professional"
/noexecute=optin /fastdetect
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="WINXP - Debug" /noexecute=optin /debug
/debugport=com1 /baudrate=115200
```

**Vista** 之后的系统，以管理员权限打开命令行窗口，通过 `bcdedit` 工具修改 BCD 数据添加启动项。

```
C:\>bcdedit /copy {default} /d "Vista - Remote Debug - 1394"
已将该项成功复制到 {13fbbcdc-756a-11dc-aed8-0016e68bceb3}。

C:\>bcdedit /set {13fbbcdc-756a-11dc-aed8-0016e68bceb3} debugtype SERIAL DEBUGPORT:1
BAUDRATE:115200
操作成功完成。

C:\>bcdedit /debug {13fbbcdc-756a-11dc-aed8-0016e68bceb3} on
操作成功完成。
```

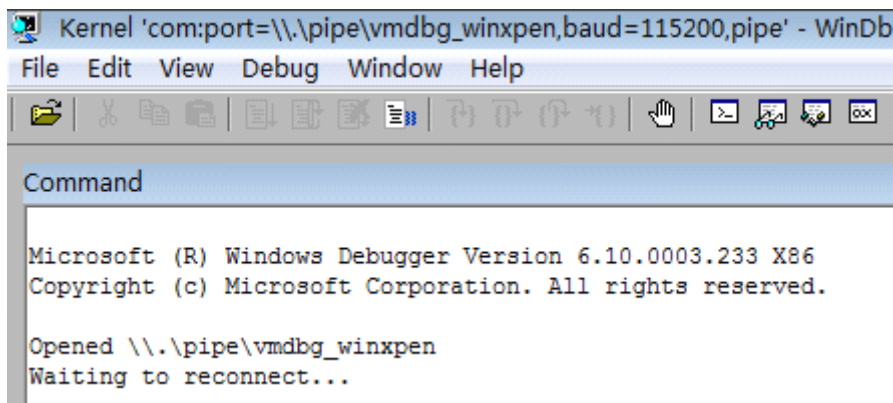
设置好启动项后，重启虚拟机中的系统，在选择启动菜单项时停下来，返回主机，通过命令行启动 WinDbg。

```
D:\WinDbg>windbg -d -k com:pipe,port=\\.\pipe\vmdbg_winxpen,baud=115200
```

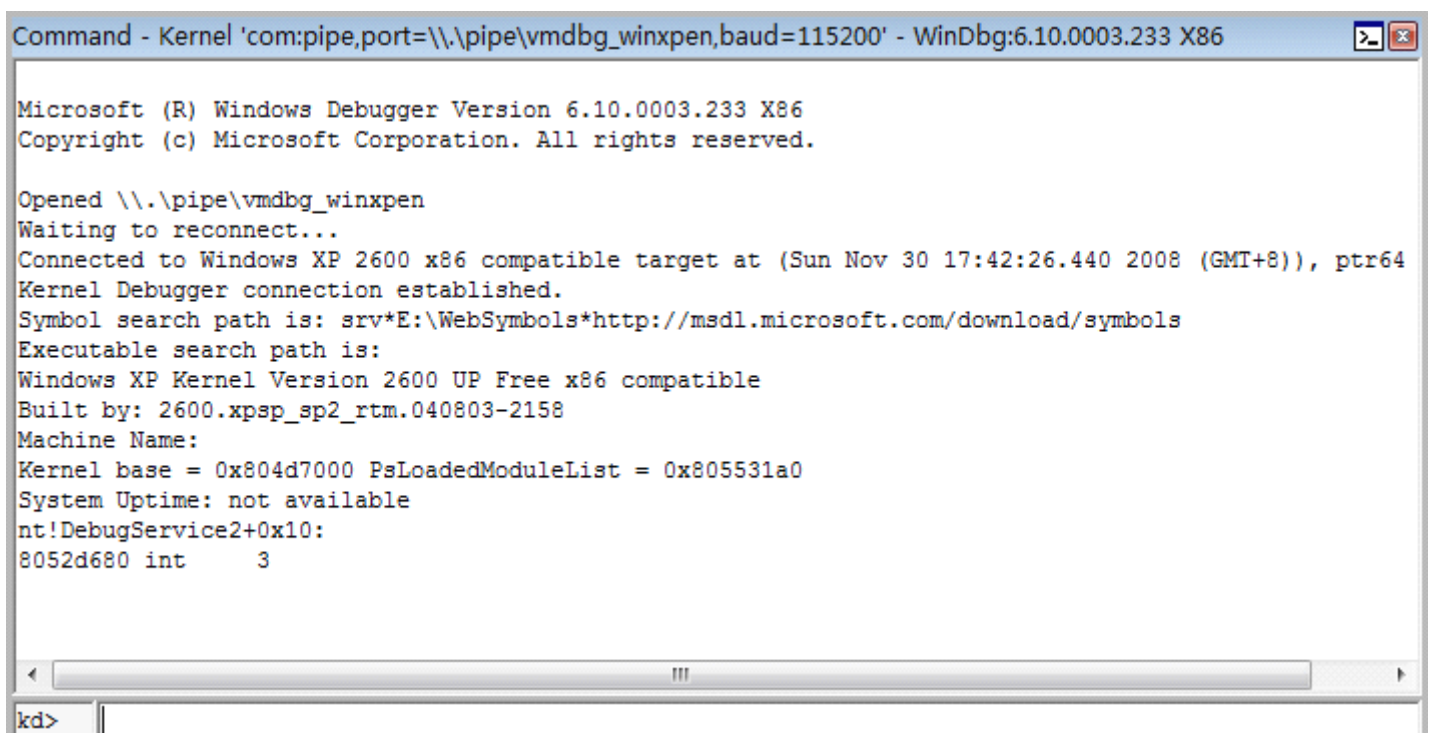
注意红色字表示的管道名称，需要和虚拟机中的设置一样。这里照样可以通过菜单打开内核调试连接对话框来操作。



注意管道名称要一致，选中 **Pipe** 项，然后确定，WinDbg 则会开始等待连接。



此时返回虚拟机中选择调试启动项，一会就能看到 WinDbg 中显示连接上虚拟机的信息。



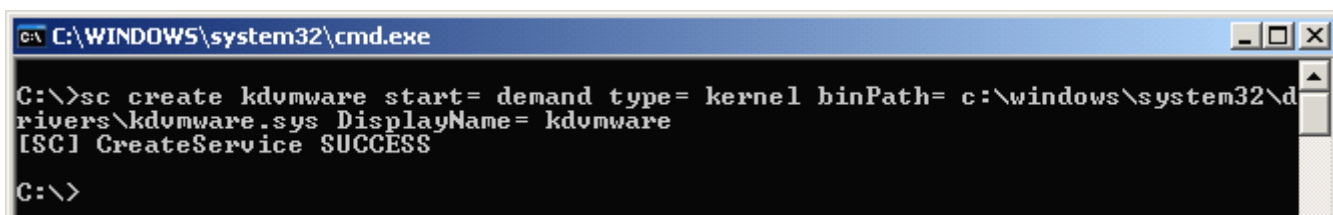
建立连接后，就和调试真实机一样，操作也一样。只有当调试某些和特殊硬件相关的驱动，或者调试和 3D 游戏

相关的程序时，还是得用到真实机。另外，运行虚拟机毕竟影响主机速度，特别是读取硬盘时感觉比较明显。而且当 WinDbg 中断下目标系统后，虚拟机基本上就把 CPU 给占满了，双核时还好，单核时搞的主机都没办法动。当然调试速度也是个问题，照理说都是一台电脑（虚拟机也还是运行在主机上），传输速度应该很快。但因为串口是完全模拟的，串口设计决定了速度不可能快，所以下面介绍一个用于虚拟机调试时的加速工具：vmkd。

Vmkd 是内核调试高手 Skywing 推出的一个工具，主要用于加快 VMWare 内核调试的速度，官方网站在：[http://www.nynaeve.net/?page\\_id=168](http://www.nynaeve.net/?page_id=168)。vmkd 相当于接管了内核调试引擎传输通道，以前是：虚拟机系统内核<->模拟串口<->命名管道<->WinDbg，使用 vmkd 后变成：虚拟机系统内核<->vmkd 内核模块 kdvmware<->vmkd 注入模块 vmxpatch<->新的命名管道<->WinDbg，因为不走虚拟机的模拟串口传输数据，而是虚拟机端的 kdvmware 和主机端的 vmxpatch 直接复制内存，所以传输速度大大加快。

详细使用步骤如下：

1. 在虚拟机中安装好系统，并按照前一节所述设置好串口调试。
2. 把 vmkd 带的 kdvmware.sys 复制到虚拟机中 C:\WINDOWS\system32\drivers 目录下，在虚拟机中使用命令行 `sc create kdvmware start= demand type= kernel binPath= c:\windows\system32\drivers\kdvmware.sys DisplayName= kdvmware` 安装驱动。



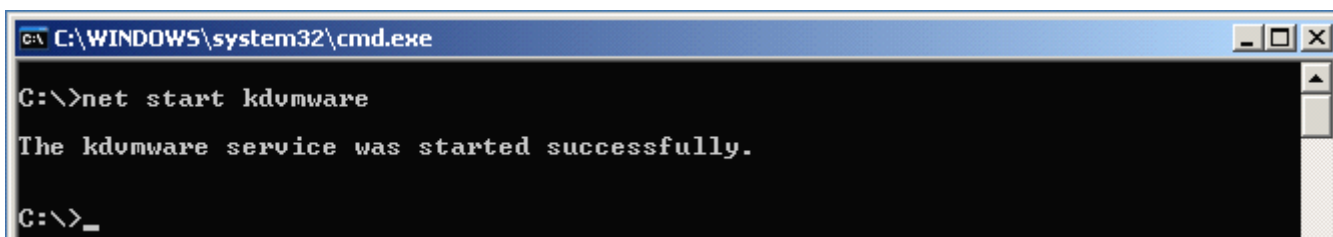
```
C:\WINDOWS\system32\cmd.exe
C:\>sc create kdvmware start= demand type= kernel binPath= c:\windows\system32\drivers\kdvmware.sys DisplayName= kdvmware
[SC] CreateService SUCCESS
C:\>
```

3. 重新启动虚拟机中的系统，选择调试启动项进系统。
4. 在主机上找到启动虚拟机的 vmware-vmx.exe 进程的 pid，利用 vmxinject.exe pid 命令把 vmxpatch.dll 注入到 vmware-vmx.exe 进程。

```
D:\DbgTools\vmkd>tlist -m vmware-vmx.exe
D:\Program Files\VMware\VMware Server\bin\vmware-vmx.exe - 6096 vmware-vmx.exe
OleMainThreadWndName

D:\DbgTools\vmkd>vmxinject 6096
OK
```

5. 在虚拟机中通过 `net start kdvmware` 命令启动驱动。



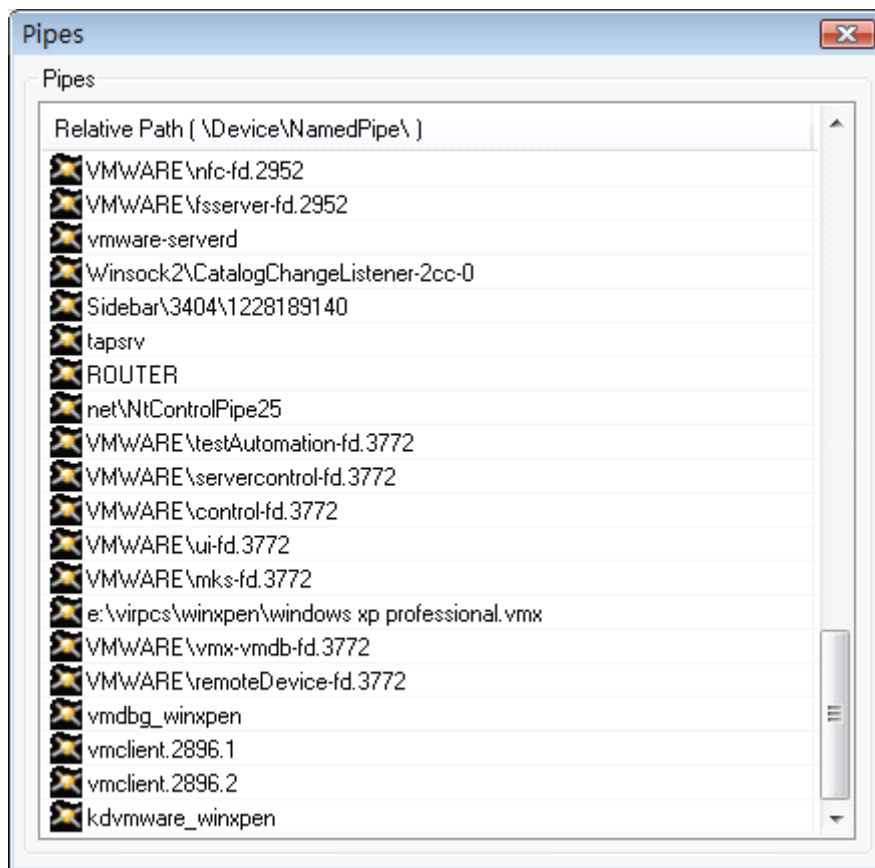
```
C:\WINDOWS\system32\cmd.exe
C:\>net start kdvmware
The kdvmware service was started successfully.
C:\>_
```

6. 在主机上启动 WinDbg 连接虚拟机中的系统开始调试，命名管道名称中的 winxpen 是虚拟机系统的安装文件夹（比如：d:\VMS\winxpen）。

```
windbg -k com:pipe,port=\\.\pipe\kdvmware_winxpen,baud=115200
```

命名管道名称如果不确定，可用 WinObjEx 工具查看系统中的所有命名管道名，如下图，最下面的就是 vmkd 使用的管道名。





连接后如下图:

```
Command - Kernel 'com:pipe,port=\\.\pipe\kdvmware_winxpen,baud=115200' - WinDbg:6.10.0003.233 X86
Microsoft (R) Windows Debugger Version 6.10.0003.233 X86
Copyright (c) Microsoft Corporation. All rights reserved.

Opened \\.\pipe\kdvmware_winxpen
Waiting to reconnect...
Connected to Windows XP 2600 x86 compatible target at (Tue Dec 2 12:46:45.549 2008
Kernel Debugger connection established.
Symbol search path is: srv*E:\WebSymbols*http://msdl.microsoft.com/download/symbols
Executable search path is:
Windows XP Kernel Version 2600 (Service Pack 2) UP Free x86 compatible
Product: WinNt, suite: TerminalServer SingleUserTS
Built by: 2600.xpsp_sp2_rtm.040803-2158
Machine Name:
Kernel base = 0x804d7000 PsLoadedModuleList = 0x805531a0
Debug session time: Wed Dec 3 03:46:41.015 2008 (GMT+8)
System Uptime: 0 days 0:47:54.609
Break instruction exception - code 80000003 (first chance)
```

此时再调试会发现单步速度非常快，运行一下搜索命令 **s** 就能感觉出来，甚至 **.dump** 命令也很快能执行完。

## 本机内核调试

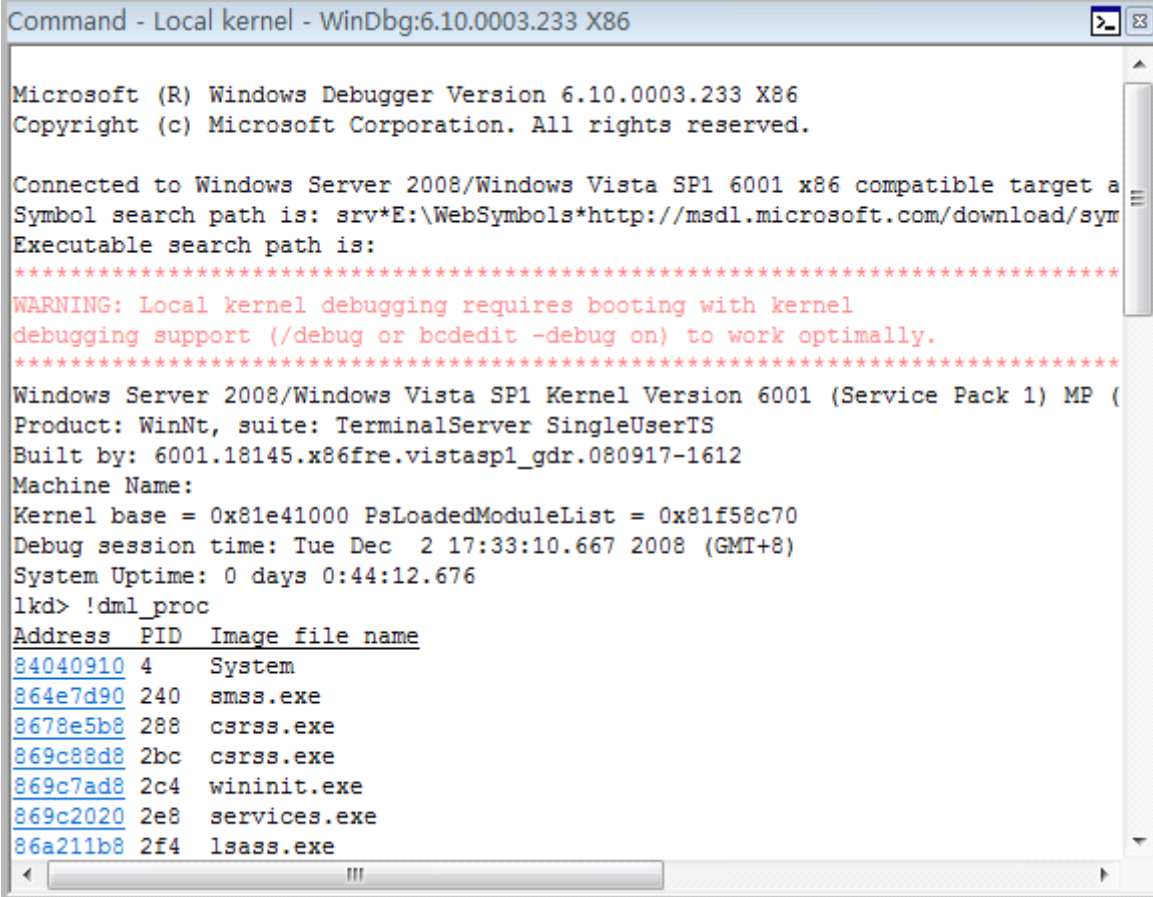
本机内核调试是 Windows XP 之后的系统引入的一种内核调试方式，说调试其实有点不准确，因为没有单步、断点等调试必备的功能，只能读写内存。不过用来查看系统内核信息时还是非常方便的，不需要两台电脑（连虚拟机也不需要，完全在本机运行），能够利用大部分 WinDbg 自带的扩展命令，在 XP 以后的系统上都能使用，包括 32/64 系统。

在 Windows 2003 SP1 之前的系统上,系统通过 `ntdll!ZwSystemDebugControl` 函数提供本机内核调试支持。之后的系统通过 WinDbg 自带的 `kldbgdrv.sys` 驱动调用 `nt!KdSystemDebugControl` 函数提供本机内核调试支持。官方文档建议在启动项中添加 `/Debug` 选项启动系统,然后使用本机内核调试功能。但一般情况下不需要这么做,Vista 之前的系统上,可以直接运行 WinDbg 打开本机内核调试;Vista 及之后的 32 位系统,可以利用笔者写的一个小工具 VistaLKD 开启本机内核调试功能,再运行 WinDbg 打开本机内核调试;Vista 及之后的 64 位系统暂时只能修改启动项添加 `/Debug` 选项来开启本机内核调试功能。

启动本机内核调试非常简单,加上 `-kl` 命令行启动 WinDbg 即可。

```
D:\WinDbg>windbg -kl
```

显示界面如下:

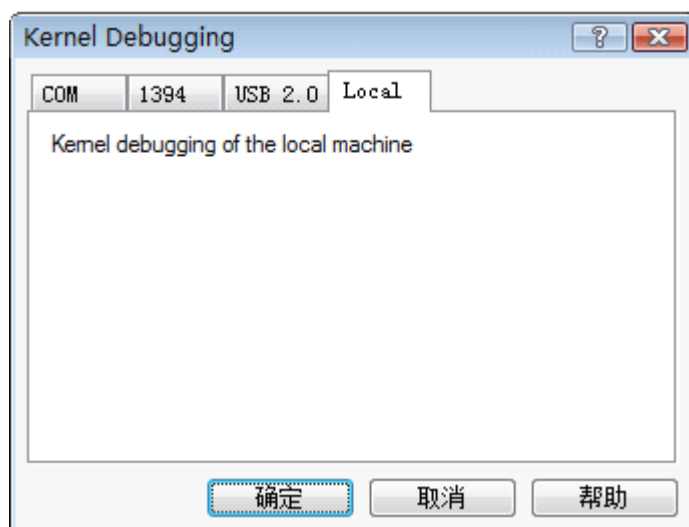


```
Command - Local kernel - WinDbg:6.10.0003.233 X86

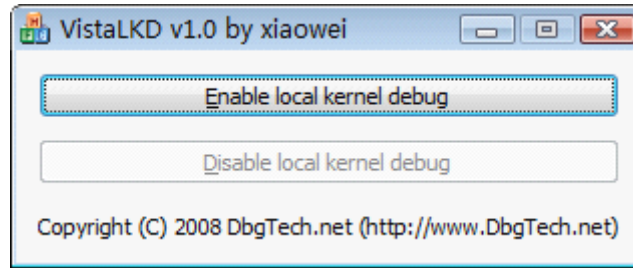
Microsoft (R) Windows Debugger Version 6.10.0003.233 X86
Copyright (c) Microsoft Corporation. All rights reserved.

Connected to Windows Server 2008/Windows Vista SP1 6001 x86 compatible target a
Symbol search path is: srv*E:\WebSymbols*http://msdl.microsoft.com/download/sym
Executable search path is:
*****
WARNING: Local kernel debugging requires booting with kernel
debugging support (/debug or bcdedit -debug on) to work optimally.
*****
Windows Server 2008/Windows Vista SP1 Kernel Version 6001 (Service Pack 1) MP (
Product: WinNt, suite: TerminalServer SingleUserTS
Built by: 6001.18145.x86fre.vistasp1_gdr.080917-1612
Machine Name:
Kernel base = 0x81e41000 PsLoadedModuleList = 0x81f58c70
Debug session time: Tue Dec 2 17:33:10.667 2008 (GMT+8)
System Uptime: 0 days 0:44:12.676
lkd> !dml_proc
Address PID Image file name
84040910 4 System
864e7d90 240 smss.exe
8678e5b8 288 csrss.exe
869c88d8 2bc csrss.exe
869c7ad8 2c4 wininit.exe
869c2020 2e8 services.exe
86a211b8 2f4 lsass.exe
```

也可以运行 WinDbg 后打开内核调试连接对话框,选择 Local 选项页,点击“确定”按钮打开本机内核调试功能。



Vista 及以后的系统，记得先以管理员权限运行 VistaLKD 工具，点击 Enable local kernel debug 按钮开启本机内核调试功能，然后再以管理员权限运行 WinDbg 打开本机内核调试。



本机内核调试下可以做很多事情，比如查看内核结构定义、反汇编内核函数、显示内核 Hook 等。

```
Command - Local kernel - WinDbg:6.10.0003.233 X86
lkd> dt nt!_KDPC
+0x000 Type           : UChar
+0x001 Importance     : UChar
+0x002 Number         : UInt2B
+0x004 DpcListEntry   : _LIST_ENTRY
+0x00c DeferredRoutine : Ptr32 void
+0x010 DeferredContext : Ptr32 Void
+0x014 SystemArgument1 : Ptr32 Void
+0x018 SystemArgument2 : Ptr32 Void
+0x01c DpcData        : Ptr32 Void
lkd> dt nt!_KIDENTRY
+0x000 Offset         : UInt2B
+0x002 Selector       : UInt2B
+0x004 Access         : UInt2B
+0x006 ExtendedOffset : UInt2B
lkd> u nt!NtOpenProcess
nt!NtOpenProcess:
82052b18 mov     edi,edi
82052b1a push    ebp
82052b1b mov     ebp,esp
82052b1d push    ecx
82052b1e push    ecx
82052b1f mov     eax,dword ptr fs:[00000124h]
82052b25 mov     al,byte ptr [eax+0E7h]
82052b2b mov     ecx,dword ptr [ebp+14h]
```