
# Übung I
# Windows Research Kernel

Michael Schöbel

Betriebssystemarchitektur I – WS 2007

01. November 2007

# Agenda

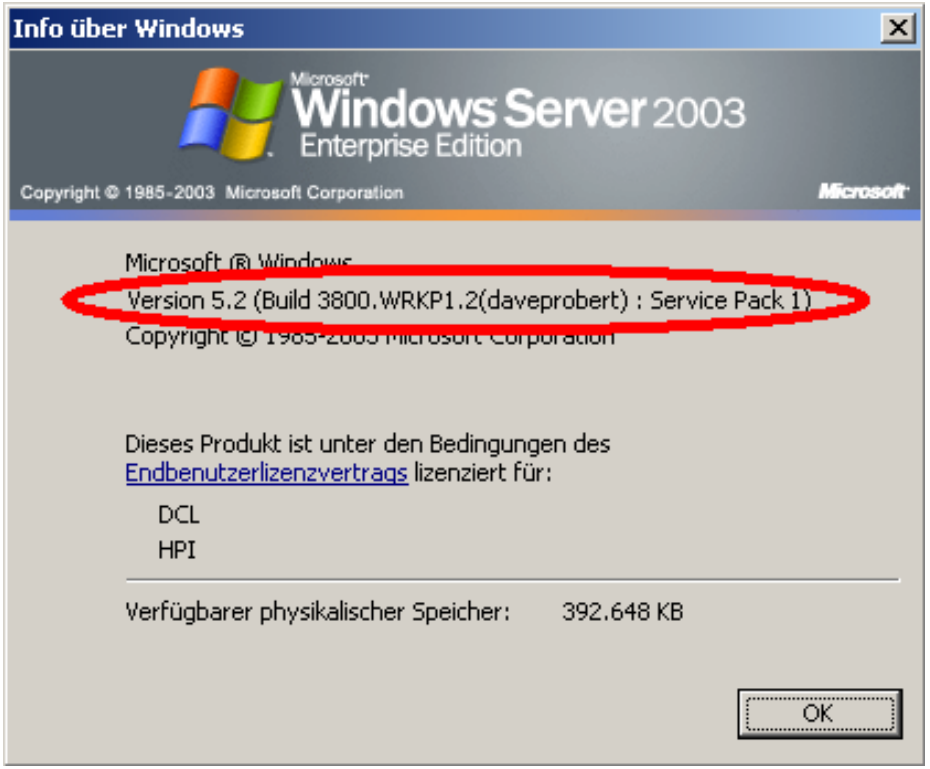


- **Windows Research Kernel**
- **(Online-) Dokumentation**
- **Entwicklungsumgebung**
- **Ausführung & Debugging**

# Windows Research Kernel Überblick



- **Erhältlich seit Sommer 2006**
- **Akademische Einrichtungen**
- **Kernelexperimente und –analysen**




- **„Compare and Contrast“ – Dave Probert**

# Windows Research Kernel Umfang



- **Etwa 27 MByte Quellcode**

| Datei Typ | # | Größe in KByte |
|---|---|---|
| *.c | 430 | 17,472 |
| *.h | 224 | 5,721 |
| *.asm | 61 | 1,143 |

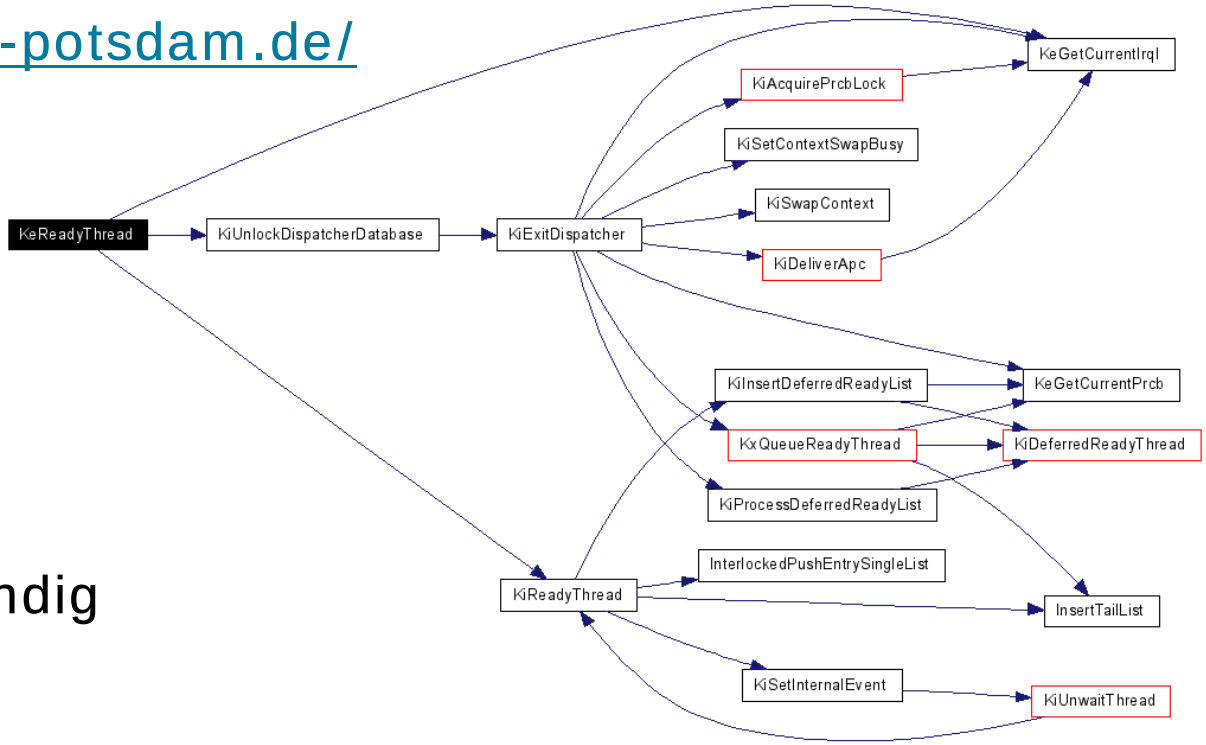
# Windows Research Kernel Verwendung



- **Windows Server 2003 Enterprise Edition SP1**
  - Installation erforderlich
  - Kompilieren und ersetzen: `ntoskrnl.exe`
  - Multiprozessor HAL installieren
  - Anpassen: `boot.ini`
  - Booten!
  - → Messungen, neue Kernelkonzepte, …

# (Online-) Dokumentation

- **WRK Doxygen-Dokumentation**
  - Ergänzung zum Quelltext
  - https://pao.dcl.hpi.uni-potsdam.de/
  - Callgraphen
  - Include-Beziehungen
  - Verlinkter Code
  - Stellenweise unvollständig
  - Hiwi gesucht! ☺



# Entwicklungsumgebung (I)

- **WRK v1.2** https://maniac.hpi.uni-potsdam.de/
  - Enthält alle notwendigen Compiler und Tools
- **Virtual Machine** http://www.vmware.com/de/products/server/
  - Windows Server 2003 Image für VMWare wird zur Verfügung gestellt
- **Debugging Tools** http://www.microsoft.com/whdc/devtools/debugging/default.mspx
  - Ermöglicht Kerneldebugging
  - Alternativ: http://www.microsoft.com/technet/sysinternals/Miscellaneous/DebugView.mspx
    - → Anzeigen der Debug-Ausgaben, kein Live-Debugging

# Entwicklungsumgebung (II)

- **VisualStudio Solution- / Project-Dateien**
  - → http://www.dcl.hpi.uni-potsdam.de/research/WRK/?p=12

- **Dokumentation**
  - NT Designdocuments
  - Doxygen-Dokumentation

  - Windows Systemaufrufe: http://www.metasploit.com/users/opcode/syscalls.html
  - Undocumented Windows 2000 Secrets: http://www.rawol.com/?topic=77

# Kompilieren des Kernels



- **Umgebungsvariablen definieren**

```
set WRKPATH=x:\WindowsResearchKernel
set ARCH=x86
set PATH=%WRKPATH%\tools\x86;%PATH%
```

- **Kompilieren**

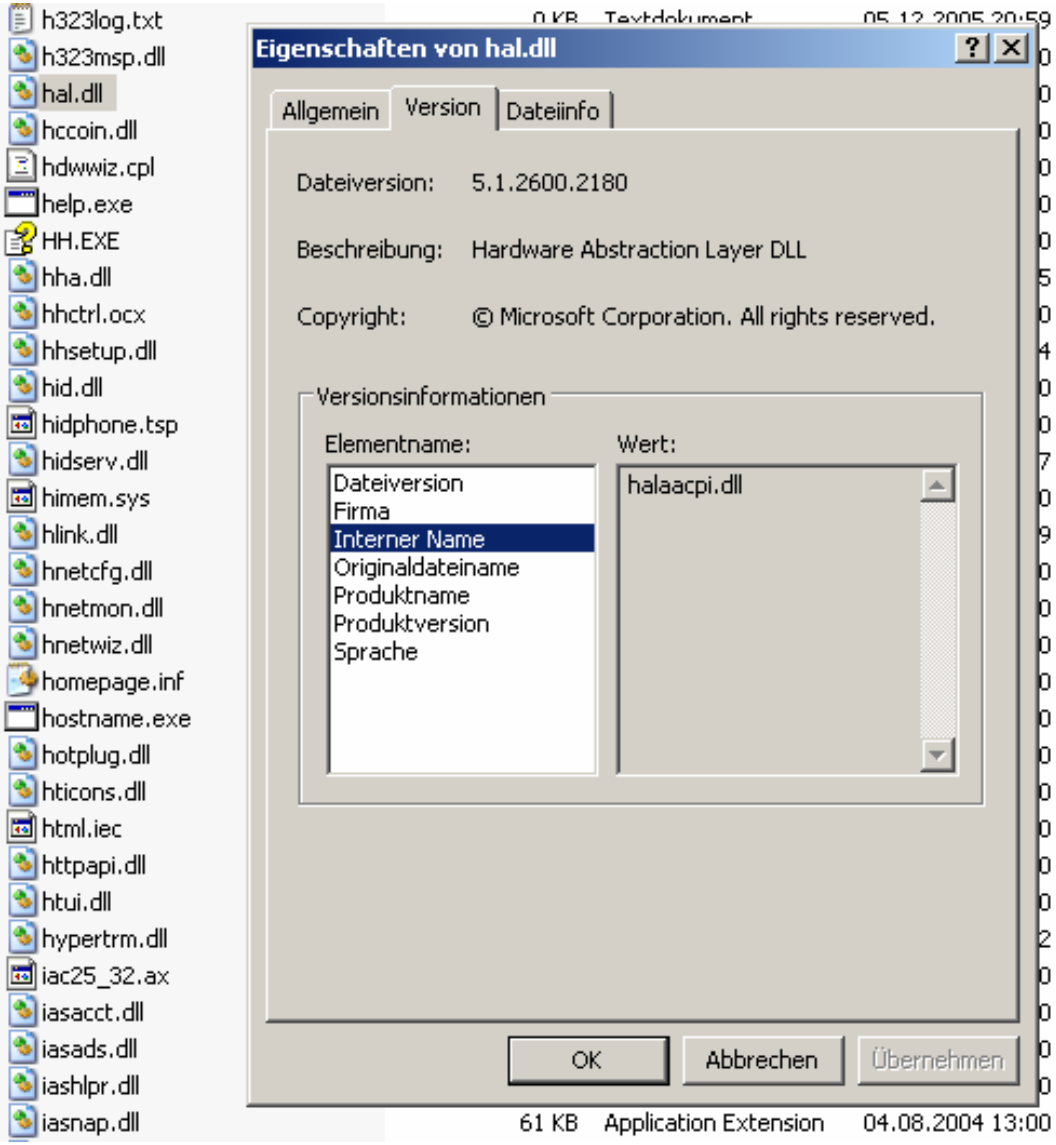
```
cd %WRKPATH%\base\ntos\
nmake -nologo %ARCH%=
```

- **Ergebnis**

```
%WRKPATH%\base\ntos\BUILD\EXE\
        → wrkx86.exe
        → wrkx86.pdb
```

# Ausführung & Debugging
# Hardware Abstraction Layer DLL

- **Multiprozessor HAL**
  - **`Windows\System32\hal.dll`**
  - → „Interner Name“
  - kopieren aus **`WS03SP1HALS\x86`**
  - **`halacpi`** → **`halacpim`**
  - **`halaacpi`** → **`halmacpi`**
  - **`halapic`** → **`halmps`**



# Ausführung & Debugging
# Kernel ersetzen ...



- **Bootmenü erweitern:** `boot.ini`

```
[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="Windows 2003"
       /noexecute=optout /fastdetect

multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="WRK"
       /kernel=wrkx86.exe /hal=halmacpi.dll
       /debug /debugport=com1
```

# Ausführung & Debugging
# ... und booten



Microsoft ® Windows
Version 5.2 (Build 3790.srv03_sp1_rtm.050324-1447 : Service Pack 1)
Copyright © 1985-2005 Microsoft Corporation

```
Wählen Sie das zu startende Betriebssystem:

    Windows Server 2003 Enterprise
    Windows Research Kernel [Debugger aktiviert]

Verwenden Sie Pfeil nach oben bzw. unten, um einen Eintrag zu markieren.
Drücken Sie anschließend die EINGABETASTE.
```

Microsoft ® Windows
Version 5.2 (Build 3800.WRKP1.2(daveprobert) : Service Pack 1)
Copyright © 1985-2005 Microsoft Corporation

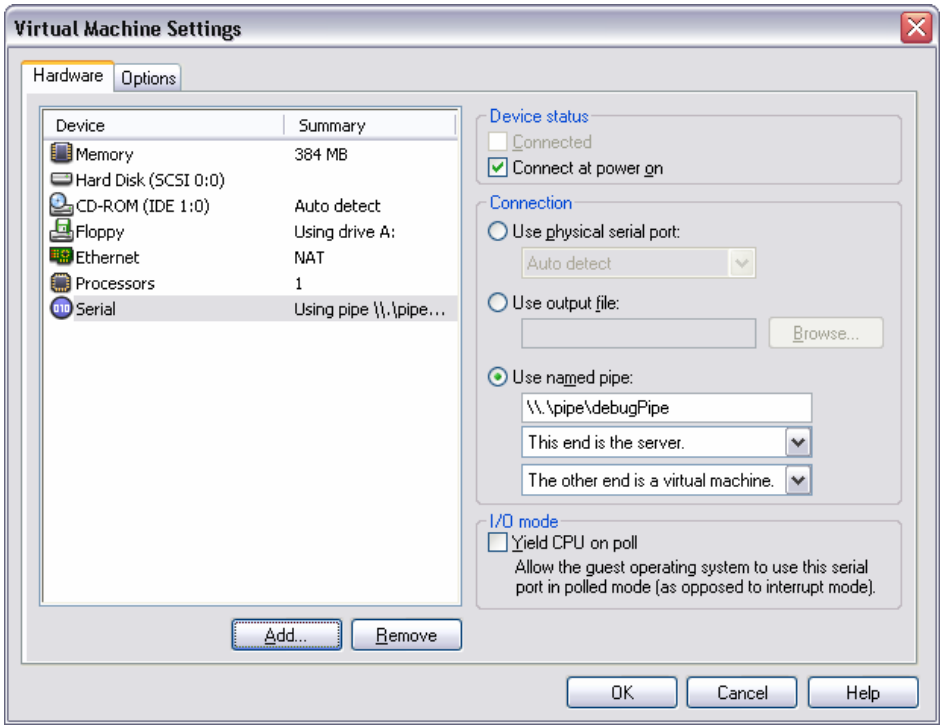
- **Informationen über Windows:** `winver`

# Ausführung & Debugging
# Virtual Machine Konfiguration

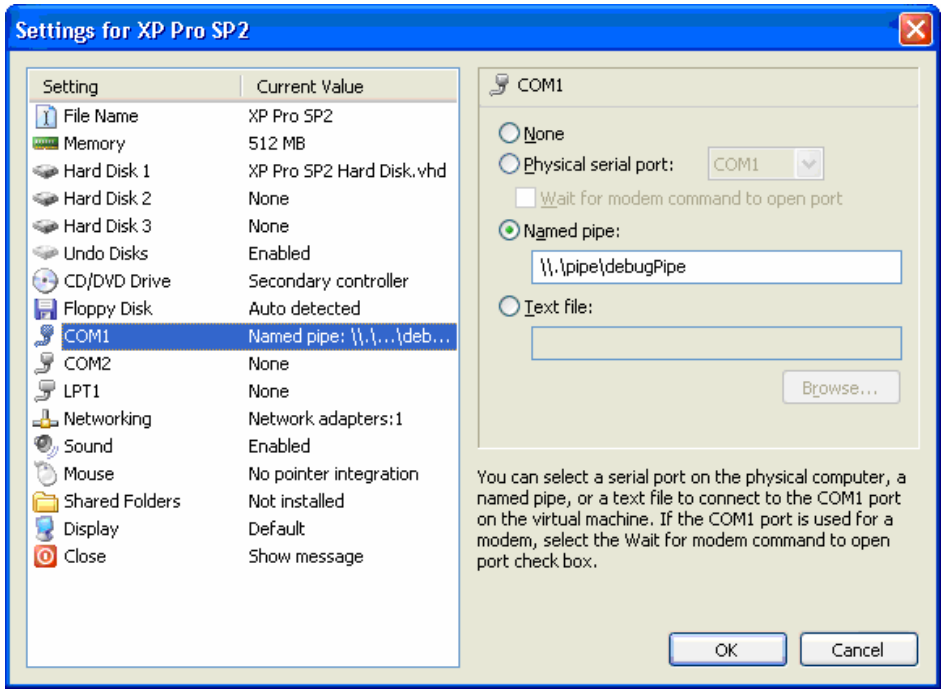


- **COM1 Port → auf Named Pipe abbilden**
  - `\\.\pipe\debugPipe`

**VMWare**




**VirtualPC**



# Ausführung & Debugging
# WinDbg Konfiguration



- **Verbindung herstellen & Debug-Symbole laden**

```
set wrksymbols=%WRKPATH%\base\ntos\BUILD\EXE\

set dbgpipe=\\.\pipe\debugPipe
set dbgargs=-k com:pipe,port=%dbgpipe%,resets=0,
            reconnect -y %wrksymbols%

"<pfad>\windbg.exe" %dbgargs%
```

# Ausführung & Debugging
# Kernel-Debug Session (I)



Kernel 'com:pipe,port=\\.\pipe\debugPipe,resets=0,reconnect' - WinDbg:6.6.0003.5

File Edit View Debug Window Help

Command

```
Microsoft (R) Windows Debugger  Version 6.6.0003.5
Copyright (c) Microsoft Corporation. All rights reserved.

Opened \\.\pipe\debugPipe
Waiting to reconnect...
Connected to Windows Server 2003 3800 x86 compatible target, ptr64 FALSE
Kernel Debugger connection established.
Symbol search path is: C:\MichaelSchoebel\code_svn\WMK\base\ntos\BUILD\EXE\;SRV*c:\MySymbols\*http://msdl.microsoft.com/download/symbols
Executable search path is:
Windows Server 2003 Kernel Version 3800 MP (1 procs) Free x86 compatible
Built by: 3800.WRKP1.2(daveprobert)
Kernel base = 0x80800000 PsLoadedModuleList = 0x808a1398
System Uptime: not available
```

Debuggee not connected

Ln 0, Col 0 | Sys 0:<None> | Proc 000:0 | Thrd 000:0 | ASM | OVR | CAPS | NUM

# Ausführung & Debugging
# Kernel-Debug Session (II)



Kernel 'com:pipe,port=\\.\pipe\debugPipe,resets=0,reconnect' - WinDbg:6.6.0003.5

File Edit View Debug Window Help

C:\MichaelSchoebel\code_svn\WMK\base\ntos\ps\psquery.c

```
    PETHREAD Thread;
    PEPROCESS Process;
    ULONG LastThread;
    KPROCESSOR_MODE PreviousMode;
    NTSTATUS st;
    THREAD_BASIC_INFORMATION BasicInfo;
    KERNEL_USER_TIMES SysUserTime;
    PVOID Win32StartAddressValue;
    ULONG DisableBoost;
    ULONG IoPending ;
    ULONG BreakOnTerminationEnabled;
    PETHREAD CurrentThread;
    ULONG ThreadTerminated;
    //
    // Get previous processor mode and probe output argument if nec
    //

    PAGED_CODE();

    CurrentThread = PsGetCurrentThread ();

    PreviousMode = KeGetPreviousModeByThread (&CurrentThread->Tcb).

    if (PreviousMode != KernelMode) {
        try {
            //
            // Since these functions don't change any state thats r
            // in the error paths we only probe the output buffer i
            // This improves performance by not touching the buffer
            // And only writing the portions of the buffer that cha
            //

            ProbeForRead (ThreadInformation,
                          ThreadInformationLength,
                          sizeof(ULONG));

            if (ARGUMENT_PRESENT( ReturnLength)) {
                ProbeForWriteUlong (ReturnLength);
            }
        } except (EXCEPTION_EXECUTE_HANDLER) {
            return GetExceptionCode ();
        }
    }

    //
    // Check argument validity.
    //

    switch (ThreadInformationClass) {

    case ThreadBasicInformation:
```

Locals

Typecast | Locations

| Name | Value |
|---|---|
| ⊞ ThreadHandle | 0x000001d4 |
| ThreadInforma... | ThreadBasicInformation (0) |
| ⊞ ThreadInformation | 0x0086f984 |
| ThreadInforma... | 0x1c |
| ⊞ ReturnLength | 0x00000000 |
| ⊞ BasicInfo | struct _THREAD_BASIC_INFORMATION |
| PreviousMode | 88 'X' |
| ⊞ SysUserTime | struct _KERNEL_USER_TIMES |

Command

```
*****************************************************************
*
*   You are seeing this message because you pressed either
*       CTRL+C (if you run kd.exe) or,
*       CTRL+BREAK (if you run WinDBG),
*   on your debugger machine's keyboard.
*
*                    THIS IS NOT A BUG OR A SYSTEM CRASH
*
* If you did not intend to break into the debugger, press the "g" l
* press the "Enter" key now.  This message might immediately reappe
* does, press "g" and "Enter" again.
*
*****************************************************************
nt!RtlpBreakWithStatusInstruction:
8086215c cc              int     3
kd> g
Breakpoint 1 hit
nt!NtQueryInformationThread+0xf:
8092e18f 648b1524010000   mov     edx,fs:[00000124]
```

kd>

Ln 3161, Col 1 | Sys 0:KdSrv:S | Proc 000:0 | Thrd 000:0 | ASM | OVR | CAPS | NUM

# Ausblick

- **Quellcode Analyse**
- **Usermode → Kernelmode Übergang**
- **Kernel Instrumentierung**
- **Erweiterung vorhandener Systemaufrufe**
- **Implementierung neuer Systemaufrufe**