Vorlesung Betriebssystemarchitektur SS 2006
Aufgabenblatt 2 vom 15. November 2007
(Vorstellung der Lösungen bei den Tutoren bis zum 29.11.2007)

Aufgabe 2.1: (20 Punkte)
Untersuchen Sie Eigenschaften des Betriebssystems Windows XP mit Hilfe des Performance Monitors (`perfmon`)! Beobachten Sie die Zähler für *privileged time*, *interrupts/second*, *processor time* und *user time*!

- Geben Sie Auskunft über die aktuelle Messung. Welche Charakteristiken des Betriebssystems werden durch diese Werte erkennbar?
- Erklären Sie die Änderungen die durch das Ausführen einer Shell (`cmd.exe`), Maus- oder Tastaturereignisse auftreten. Versuchen Sie die gemessenen Werte zu maximieren.

Aufgabe 2.2: (10 Punkte)
Untersuchen Sie die Eigenschaften des Betriebssystem Linux! Lesen Sie dazu die Manualseiten: *top(1)* und *ps(1)*!

- Woran erkennt man die Prozesse die als Dämonen laufen?
- Wo lassen sich die von *ps* angezeigten Informationen im Dateisystem finden?

Aufgabe 2.3: (20 Punkte)
Kernel-Objekte repräsentieren Windows-Ressourcen, die von Applikationen verwendet werden können.

- Erläutern Sie für die Objekte *File*, *Mutant*, *Directory* und *Event* welche Betriebsmitteln diese repräsentieren und welche Funktionalität sich dahinter verbirgt.
- Was ist ein Handle? Erläutern Sie, warum Handles nicht ohne Weiteres über Prozessgrenzen hinaus verwendet werden können. Benennen Sie mindestens ein Windows-API-Funktion, mit der dies doch möglich ist.
- Der Kernel-Adressraum ist in zwei große Heap-Bereiche unterteilt. Benennen Sie diese und erläutern Sie den Unterschied. Durch den Verbrauch welchen Heaps kann das System zum Quasi-Stillstand gebracht werden?
- Schauen Sie sich die Datei `handle.c` des Windows Research Kernels an. Ermitteln Sie durch Code-Review die maximale Anzahl von Betriebsmitteln, die ein Prozess besitzen kann. Benennen Sie außerdem die Konstante, die diesen Wert enthält.
- Nennen Sie 5 Aufgaben des Moduls Object Manager!

Aufgabe 2.4: (20 Punkte)
Eine einfache Instrumentierung des Windowskerns kann durch Debug-Ausgaben (`DbgPrint`) realisiert werden. Instrumentieren Sie den WRK, so dass bei Auftritt der folgenden Kern-Ereignisse die entsprechenden Informationen auf der Debug-Konsole ausgegeben werden!

- Prozesserzeugung PID, PID+TID vom erzeugenden Thread
- Threaderzeugung PID+TID des neuen Threads, PID+TID vom erzeugenden Thread, Start-Adresse des Threads
- Prozessbeendigung PID des beendeten Prozesses
- Threadbeendigung PID+TID des beendeten Threads

Demonstrieren Sie Ihrem Tutor Ihren geänderten Windows-Kern und beantworten Sie die folgenden Fragen!

- Welche Kernel-Funktionen müssen geändert werden?
- Welche Funktionen werden (direkt nach dem Starten) durch die ersten drei erzeugten Threads ausgeführt?
- Es existiert ein Treiber, der ständig Threads erzeugt. Diese Threads existieren nur für eine kurze Zeit. Wie heißt dieser Treiber bzw. welche Funktion führen diese Threads aus?

Aufgabe 2.5: (30 Punkte)
Implementieren Sie eine Windows Usermode-Bibliothek (`myntdll.dll`), die die Funktion

```
SetThreadPriorityExplicit(HANDLE tHandle, ULONG prio)
```

bereitstellt!

Berücksichtigen Sie dabei die folgenden Teilaufgaben:

- Welche Systemaufrufnummern sind der Funktion `NtSetInformationThread` zugeordnet (Windows XP und Server 2003)?
- Beschreiben Sie die Signatur der Funktion `NtSetInformationThread`!
- Welche Bedeutung hat die Aufzählung `THREADINFOCLASS`?
- Welchen Wertebereich hat die Thread-Priorität?
- Verwenden Sie `NtSetInformationThread` um die Priorität eines Threads auf den übergebenen Wert zu setzen! Ergänzen Sie dazu die Assembler-Routine im bereitgestellten Programmrahmen!
- Testen Sie Ihre DLL mit Hilfe des bereitgestellten Programmrahmens! Zeigen Sie mit Hilfe von `perfmon`, den zeitlichen Verlauf der Thread-Priorität!

Demonstrieren Sie das Programm Ihrem Tutor!

Verpacken Sie die Source-Code Dateien (inklusive Makefile) in eine ZIP-Datei (blatt2.zip). Diese ZIP-Datei muss mindestens 24 Stunden vor dem Tutoriumstermin in das Abgabesystem eingestellt werden.