# VO Betriebssysteme – Fragenausarbeitung April 2016

von Alexandra H.

## Was versteht man unter einem Microkernel? Welche Services stellt ein Microkernel zur Verfügung? Welche Vor- bzw. Nachteile ergeben sich bei der Verwendung eines Microkernel-Betriebssystems? (5)

Basisservices befinden sich im Kernel (Interrupts, Hardware Access I/O, Process-Switching, Basic Memory Management, Nachrichtenaustausch & -kontrolle), nicht zentrale Services sind als Server-Prozesse realisiert
Vorteile: einheitliches Interface, Flexibilität, Erweiterbarkeit, Portabilität, Unterstützung von Verteilung
Nachteile: mehr Kontext-Wechsel → langsamer als monolithisch, komplexe Synchronisation

## Was versteht man unter einem Process Control Block? Beschreiben Sie, aus welchen Teilen der PCB besteht und welche Informationen in diesen Teilen jeweils verwaltet werden. (6)

PCB ist Teil des Process Images (für Execution Context)

1. Process-Identification: Process-Identifier (eindeutige Prozessnumer, Index in Primary Process Table), User-Identifier (Benutzer, dem Prozess gehört), Parent-Process-Identifier (Nummer des Prozesses, der den Prozess generiert hat)
2. Prozessor-Zustand: Registerinhalte, Kontroll- und Statusregister, Stack-Pointers
3. Prozess-Kontrollinformation: Verweise auf andere Prozesse (z.B. Parent/Child), Scheduling- und Zustandsinformationen (Prozesszustand, Priorität, Ereignis auf das Prozess wartet), Interprozesskommunkation, Ressourcen, Privilegs, Memory Management

## Worin liegt der grundlegende Unterschied zwischen Prozessen und Threads? Welcher Vorteil ergibt sich aus der Einführung von Threads für den Benutzer und worauf muss der Benutzer achten? (4)

Prozess: Einheit für Ressourcenverwaltung (virtueller Adressraum mit dem Process Image)
Thread: Einheit für Dispatching (teilen sich Adressraum und PCB)
Mehrere Threads pro Prozess sind möglich

Thread-Erzeugung, -Terminierung und das Umschalten zwischen den Threads schneller, Threads können ohne Kernel miteinander kommunizieren, aber es ist eine Synchronisation notwendig

## Was versteht man unter Deadlock Avoidance? Geben Sie zwei Strategien für Deadlock Avoidance an und beschreiben Sie diese. (4)

Strategie um Deadlocks zu vermeiden, selektives Vergeben von Ressourcen

- Process Initiation Denial: Prozess wird nicht gestartet wenn seine Anforderungen zu einem Deadlock führen können
- Ressource Allocation Denial: Ressourcenanforderungen eines Prozesses werden verwehrt wenn sie zu einem Deadlock führen könnten z.B. durch Banker's Algorithm

## Was versteht man unter einem Monitor zur Prozesssynchronisation? Nennen Sie die wichtigsten Komponenten und Eigenschaften des Monitors. (4)

= Softwaremodul, sorgt für Mutual Exclusion, kein explizites Programmieren notwendig

Komponenten: Prozeduren/Zugriffsfunktionen, lokale Daten (gemeinsamer Zugriff), Bedingungsvariable/Monitorvariable (regelt Monitoreintritt)

Eigenschaften: gemeinsame Objekte sind gekapselt, max. 1 Prozess zu einem Zeitpunkt im Monitor, Eintritt in den Monitor erfolgt über Monitorprozeduren, Zugriff auf lokale Variable

## Was versteht man unter dem Begriff Relocation? Wofür ist Relocation von Bedeutung? (3)

Relocation = Positionierung von Code oder Daten im Speicher
Der Speicherbereich eines Prozesses ist nicht statisch fixiert (unterschiedliche Speicherbelegung zum Prozessaufruf, Swapping etc.), Daten müssen also an verschiedenen Speicherstellen positioniert werden können und Referenzen auf den physikalischen Speicher veränderbar sein. Logische Adressen werden in physikalischer Adressen übersetzt.

## Beschreiben Sie, wozu und wie eine Page Table verwendet wird. Geben Sie weiters an, welche Informationen in den Tabelleneinträgen einer Page Table gespeichert werden. (4)

Pro Prozess gibt es eine Page Table. Sie enthält:
- Present Bit: befindet sich die Page im Hauptspeicher?
- Frame Number: die aktuelle Frame-Nummer für jede Page d.h. wo Seite zu finden ist
- Modified Bit: wurde die Seite seit dem Laden verändert?
- Control Bits (z.B. Kernel oder User Page?)

Mit ihrer Hilfe kann der Prozessor eine logische Adresse (Page Number, Offset) in eine physikalische Adresse (Frame Number, Offset) übersetzen.

## Beschreiben Sie das Ziel von Disk Scheduling. Nennen Sie drei „intelligente" Disk-Scheduling Algorithmen und beschreiben Sie diese kurz. (5)

Ziel: Erhöhung der durchschnittlichen Zugriffsgeschwindigkeit durch Ordnen der I/O-Request zur Minimierung der Seek Time (Ts)

- FIFO (First In, First Out → erster eingegangener I/O-Request als erstes ausgeführt)
- LIFO (Last In, First Out → letzter eingegangener I/O-Request als erstes ausgeführt)
- Priority (I/O-Requests haben Prioritäten, höhere Prioritäten zuerst)
- Shortest Service/Seek Time First (SSTF → Request mit der Shortest Seek Time als erstes)

## Was versteht man unter Buffering? Welche Vorteile bietet es, wo liegen seine Grenzen und worauf hat man bei der Verwendung von Puffern bei der Betriebssystemimplementierung zu achten? (5)

= Zwischenspeichern von Daten

Vorteile: erhöht Performance und Flexibilität, weniger Overhead durch Zusammenfassen von Daten

Aber erhöhte Kosten: Man benötigt Puffermanagement (Zuordnung: Puffer – Prozesse) und Speicher. Buffer-Overflow!

Grenzen: Die Langzeit-Performance ist nie besser als die Device-Performance.

## Was versteht man unter einer File Allocation Table? Wie ist diese organisiert? (2)

Wird bei Indexed Allocation verwendet (= Strategie zur Block-Allokierung)
FAT ist eine Tabelle im Speicher, die Pointer auf Folgeblöcke einer Datei enthält. Da die Pointer in einer eigenen Tabelle gespeichert werden und nicht in den Blöcken der Datei selbst (wie bei der Chained Allocation), ist mehr Platz für Nutzdaten vorhanden. FAT benötigt jedoch viel Platz im Arbeitsspeicher.

## Was beschreibt das Modell von Bell und LaPadula? Geben Sie die vom Modell geforderten Eigenschaften an. (4)

Definiert Regeln für Informationsfluss (für Protection):
Objekte und Subjekte haben Securtiy Classifikations SC (z.B. top secret, secret, public)
Gibt verschiedene Arten von Zugriffs-Operationen (von Subjekt auf Objekt):
- read only (keine Veränderungen)
- append = hinzufügen (ohne read)
- execute (ohne read und write)
- read-write

Subjekt S darf Objekt O nur
- lesen, wenn: SC(S) >= SC(O)
- append, wenn SC(S) <= SC(O)
→ read-write, wenn SC(S) = SC(O)

## Erklären Sie die Aktionen, die vom Betriebssystem bei einem Process Switch durchzuführen sind. Welche Arten von Ereignissen können zu einem Process Switch führen? (3)

Aktionen: Umschalten des aktiven Prozesses, alten Prozess entweder auf Blocked-Queue setzen oder terminieren, Prozess aus Ready-Queue holen, neuem Prozess CPU zuteilen

Führt zu Process Switch (Umschalten des aktiven Prozesses):
1) Supervisor Call: expliziter Aufruf durch Programm
2) Trap: durch aktuelle Instruktion (z.B. bei Fehler)
3) Interrupt: Ursache liegt außerhalb des Prozesses

## Beschreiben Sie die folgenden Strategien für das CPU Scheduling und vergleichen Sie deren Eigenschaften: Round Robin, Virtual Round Robin, Highest Response Ratio Next und Feedback Scheduling, FCFS, SPN und SRT. (6)

FCFS (First Come, First Serve): Prozess, der schon am längsten in der Ready-Queue ist, wird ausgewählt. Non-preemptive. Begünstigt lange und CPU-intensive Prozesse.

Round Robin: Wie FCFS mit Zeitscheiben fixer Länge, die nacheinander an Prozesse vergeben werden. Ist die Zeit abgelaufen und der Prozess noch nicht terminiert, kommt er wieder ans Ende der Ready-Queue → preemptive. Benachteiligt I/O-intensive Prozesse, da diese die Zeitscheiben nicht voll auskosten können.

Virtual Round Robin: Wie Round Robin, aber mit Auxiliary Queue, die höhere Priorität als die Ready Queue hat. Nach Ablauf der Zeitscheibe, kommt ein noch nicht terminierter I/O-Prozess in eine I/O-Queue, in der sich alle I/O-Prozesse befinden, die auf ein bestimmtes I/O-Event warten, während andere nicht terminierte Prozesse (z.B. CPU-intensive) wieder in die Ready Queue kommen. Tritt das I/O-Event ein, kommt der I/O-Prozess in die Auxiliary Queue und bekommt die CPU vor Prozessen in der Ready Queue → keine Benachteiligung für I/O-Prozesse

SPN (Shortest Process Next): Wählt Prozess mit kürzester zu erwartender CPU-Zeit. Non-preemptive. Hat bessere Response Time als FCFS aber auch größere Variabilität der Response Times (langer CPU-intensiver Prozess wird als letztes ausgewählt, egal wie lange schon da → Starvation langer Prozesse). Außerdem müssen die Abarbeitungszeiten geschätzt werden.

SRT (Shortest Remaining Time): Wie SPN, aber preemptive. Protokollieren der Zeiten nötig, Starvation möglich. Dafür werden kürzere Prozesse fair behandelt und es kommt zu weniger Interrupts als bei Round Robin.

Highest Response Ratio Next: Mittels Selection Function: RR=(w+s)/s (w…bisherige Wartezeit, s…geschätzte Service Time); Prozess mit größtem RR wird ausgewählt. Non-preemptive. Kürzere Prozesse werden fair behandelt, keine Starvation. Zeiten müssen aber wieder geschätzt werden.

Feedback Scheduling: Je mehr CPU-Zeit ein Prozess bisher konsumiert hat, desto niedriger wird seine Priorität → Queue für jede Prioritätsstufe. Preemptive. Starvation ist möglich, kann aber behoben werden, indem Priorität nach einer bestimmten Zeit angehoben wird.

Earliest Deadline First: Real-Time-Scheduling Strategie. Preemptive. Minimiert Anzahl der Deadline-Misses.

Wir betrachten ein System mit Virtual Memory Management. Diskutieren Sie, welche der folgenden Situationen beim Referenzieren einer virtuellen Adresse auftreten bzw. nicht auftreten kann: (a) TLB Miss ohne Page Fault, (b) TLB Miss mit Page Fault, (c) TLB Hit ohne Page Fault, (d) TLB Hit mit Page Fault. (4)

TLB = Translation Lookaside Buffer (Cache für Einträge der Page Table, enthält Nummern der zuletzt verwendeten Seiten)
Page Fault = Adresse wird referenziert, die sich nicht im Hauptspeicher befindet

a) kann auftreten z.B.: die Page existiert, deren Nummer befindet sich aber nicht im TLB
b) kann auftreten z.B.: die referenzierte Page existiert gar nicht

c) kann auftreten z.B.: Page wurde zuletzt verwendet und ist deshalb im TLB & existiert
d) kann nicht auftreten: Page, die im Hauptspeicher gelöscht wird, wird auch im TLB gelöscht

## Mit welcher logischen Struktur des I/O Systems versucht man bei der Realisierung von I/O Funktionen sowohl ein einheitliches Programmierinterface, als auch eine möglichst gerätespezifische Ansteuerung zu erreichen? (4)

Struktur von I/O-Funktionen hat 3 Schichten:
- Logical I/O: logische Bedienung (open, close, read, write,…)
  Bei File-System-I/O: Directory Management, File System, Physical Organisation
- Device I/O: Übersetzung von Operationen in Sequenz von I/O-Kommandos
- Scheduling and Control: Queuing, Interrupt-Handling, Verwaltung von I/O-Operationen, Lesen des I/O-Status

## Was versteht man unter Synchronous bzw. Asynchonous I/O? Beschreiben Sie die beiden Arten, I/O-Operationen durchzuführen. (3)

Synchronous: Prozess schickt I/O-Request und wartet, bis I/O-Response da ist (vom Zustand Running auf die BlockedQueue gestellt)

Asynchronous: Prozess schickt I/O-Request aber wartet nicht auf I/O-Response, sondern arbeitet derweilen weiter, I/O wird zu einem späteren Zeitpunkt ausgeführt, kein Blockieren

## Erklären Sie die Begriffe Access Control List und Capability List. Wozu und wie werden diese verwendet? (4)

Sorgen für autorisierten Zugriff auf Objekte. Zugriff nur, wenn Subjekt passende Rechte hat.

Access Control List: Zugriffsrechte werden direkt bei Objekten gespeichert, Differenzierung von Benutzergruppen

Capability List: Jeder Prozess hat eine Liste von Objekten mit erlaubten Zugriffsoperationen = Capability Tickets. Zugriff auf Objekt nur mit Ticket. Ticket kann vererbt und weitergegeben werden & sind verschlüsselt, um nicht gefälscht zu werden.

## Wie berechnet sich die mittlere Zugriffszeit beim Lesen von Daten von einer mechanischen Festplatte? Geben Sie die charakteristischen Zeitparameter einer Festplatte an. Durch welche Strategien kann das Betriebssystem dazu beitragen, die mittleren Zugriffszeiten auf die Festplatte zu reduzieren? (5)

T(A) … Average Access Time (mittlere benötigte Zeit für Datenzugriff)
T(S) … Seek Time (Zeit, um Disk-Arm auf die gewünschte Spur zu setzen)
T(RD) … Rotational Delay (Verzögerung, bis Anfang des gewünschten Sektors gefunden)
T(TF) … Transfer Time (Zeit zum Übertragen der Daten)
b … Anzahl der zu übertragenden Bytes
N … Anzahl von Bytes pro Spur
r… Umdrehungsgeschwindigkeit

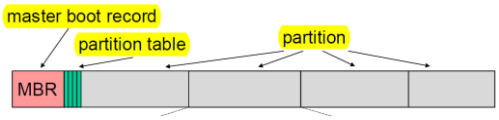
T(A) = T(S) + T(RD) + T(TF)

T(RD) = 1 / (2*r)
T(TF) = b / (r*N)

Reduktion der Zugriffszeit durch Ordnen der I/O-Requests (Minimierung der Seek Time): SSTF, SCAN, C-SCAN, FSCAN, N-Step-SCAN (Prioritäten, FIFO, LIFO reduzieren die Zugriffszeit nicht wirklich)

## Beschreiben Sie das typische Layout einer Disk bzw. eines Filesystems. Welche Rolle spielen die einzelnen Teile beim Hochfahren des Betriebssystems? (5)

Disk ist in Partitionen unterteilt. Master Boot Record (MBR) befindet sich im Sektor 0 der Disk: enthält den Boot Code und den Partition Table (start/end of partitions, active partitions).




Beim Hochfahren des BS wird der Code des MBR ausgeführt: Aktive Partitionen werden lokalisiert und geladen durch Ausführung des Boot Blocks.

## Wie ist ein i-node aufgebaut? Welche Informationen enthält er? (4)

i-Node (= Index Node): Ist eine Datenstruktur, für jedes einzelne File. Sie enthält die Attribute des Files und Referenzen auf die Fileblöcke. Unix: Erster i-Node verweist auf root-Directory.
Vorteil: i-Node wird nur im Memory gebraucht, wenn File verwendet wird.
Nachteil: Anzahl der Blockreferenzen ist begrenzt → viele indirekte Blöcke

## Nennen Sie die drei Kategorien von Security Threats und beschreiben Sie diese. Geben Sie für jede Kategorie an, welches grundlegende Security-Ziel dadurch bedroht wird. (4)

Denial Of Service (DOS): vorübergehende oder permanente Unterbrechung des Services durch Überlastung des Rechners, kann kein sinnvolles Service mehr zur Verfügung stellen
→ Availabilty wird bedroht

Exposure/Interception: nicht autorisierter Lesezugriff (oft unbemerkt)
→ Confidentiality wird bedroht

Modification/Fabrication: Veränderung der Datenintegrität durch Verändern oder Generierung von Daten, Ressourcen werden unautorisiert verwendet
→ Integrity wird bedroht

## Nennen Sie Design Prinzipien für die Konstruktion von sicheren Systemen. Geben Sie für jede Regel ein Beispiel an. (4)

- Open Design
- Default Einstellung: keine Berechtigung, Least Privilege
- Einfachheit der Sicherheitsmechanismen → Fehlervermeidung

- Acceptability: akzeptierbare Mechanismen
- Überprüfung der gegenwärtigen Berechtigung z.B. durch Authentification
- Kontrolle aller Zugriffe auf Ressourcen z.B. durch Logging

## Wodurch unterscheiden sich die Prozesszustände Ready und Blocked? (2)

Ready: Prozess ist bereit zur Ausführung und wartet auf Zuteilung der CPU
Blocked: Prozess ist nicht laufbereit, wartet auf Eintritt eines Ereignisses; tritt Ereignis ein, wird der Prozess auf die Ready-Queue gesetzt

## Wie unterscheidet sich das Blockierverhalten von Kernel Level Threads und User Level Threads? (2)

ULT: bei Blockierung durch System Call werden alle Threads blockiert
KLT: einzelne Threads können blockiert werden

(ULT: kein multi-threading möglich,
KLT: Thread-Switching benötigt 2 Mode Switches -> langsamer als ULT)

## Für die Lösung des Problems des geregelten Eintritts in einen kritischen Abschnitt werden drei Eigenschaften gefordert. (a) Nennen Sie diese drei Eigenschaften und erklären Sie deren Bedeutung. (b) Wodurch werden die drei Eigenschaften gewährleistet, wenn Semaphore zum Schutz eines kritischen Abschnitts verwendet werden? (5)

a:

1) Mutual Exclusion: verhindern des gleichzeitigen Zugriffs auf Ressourcen
2) Progress: kein Prozess im k.A. → mehrere Prozesse möchten in den k.A. eintreten: Entscheidung welcher Prozess eintritt darf nicht unendlich lang verzögert werden
3) Bounded Waiting: Prozess hat Request für k.A. abgegeben → Anzahl der Eintritte in den k.A. durch andere Prozesse ist abgeschrankt

b:

1) durch wait(S) und signal(S) wird Zugriff auf k.A. für Prozesse blockiert
2) wartende Prozesse werden in eine Blocked-Queue gestellt, Eintrittsreihenfolge muss festgelegt werden z.B. FIFO
3) ebenfalls durch Blocked-Queue: Prozesse, die schon vorher Request abgegeben haben stehen vor dem Prozess in der Blocked-Queue, nachfolgende Prozesse nachher

## Gegeben sei ein Computersystem, in dem Ihnen zur Synchronisation bzw. Kommunikation von Prozessen nur Nachrichten zur Verfügung stehen (d.h., es gibt keine Semaphore oder andere Synchronisationskonstrukte). Nennen Sie zwei verschiedene Möglichkeiten, wie Sie in diesem Computersystem einen konsistenten Datenaustausch zwischen parallelen Prozessen realisieren können. (4)

1) Zustandsnachricht mittels senden eines Speicherinhalts (non-blocking) und receive, das die alten Datenwerte durch neue ersetzt (nicht-konsumierend)
2) Ereignisnachrichten mittels Queue (receive konsumiert, send ist blocking oder non-blocking)

## Bei der Realisierung von Dateisystemen gibt es verschiedene Möglichkeiten, um die zu einer Datei gehörenden Datenblöcke zu organisieren bzw. auffindbar zu machen (Block-Allokierung). Nennen Sie vier verschiedene Strategien zur Block-Allokierung von Dateien und beschreiben Sie diese mit ihren Vor- und Nachteilen. (6)

Contiguous Allocation: Datei belegt aneinander grenzende Blöcke.
Vorteil: gute Performance
Nachteil: externe Fragmentierung, Platzprobleme bei Vergrößern einer Datei

Chained Allocation: Blöcke sind über Pointer auf den Folgeblock verkettet
Vorteil: keine externe Fragmentierung
Nachteil: langsamer Zugriff, Pointer brauchen selbst Platz im Block

Indexed Allocation: Wie Chained Allocation, aber Pointer befinden sich nicht direkt im Block, sondern in einer File Allocation Table (FAT) im Speicher.
Vorteil: Blöcke ganz für Nutzdaten verfügbar
Nachteil: FAT benötigt viel Platz im Arbeitsspeicher

i-Nodes (Index Nodes): Jedes File hat eine Datenstruktur, die Fileattribute und Pointer auf Fileblöcke enthält.
Vorteil: i-Node wird nur im Memory gebraucht wenn File verwendet wird
Nachteil: begrenzte Zahlt von Blockreferenzen (Pointern) → merfach indirekte Blöcke

## Wozu wird die Clock Policy verwendet? Beschreiben Sie deren Funktionsweise. (5)

= Strategie, die bestimmt, welche Seite beim Laden einer neuen Seite in den Hauptspeicher ersetzt wird
= Ringpuffer mit Frames, die für den Austausch in Frage kommen + Positionszeiger
Jedes Frame hat ein Use Bit, das auf 1 gesetzt wird, wenn die Seite geladen oder referenziert wird. Bei Page Fault: Der Zeiger wird auf den nächsten Frame im Puffer gesetzt. Ab der Zeigerposition wird der erste Frame gesucht, dessen Use Bit 0 ist → wird ausgetauscht. Die Use Bits mit 1 werden auf 0 gesetzt.

## Was versteht man unter der Working-Set Strategie? Beschreiben Sie deren Funktionsweise und erklären Sie, wie diese Strategie zur Optimierung eines Paging-Systems eingesetzt werden kann. (5)

Strategie, die bestimmt, wie viele Frames einem Prozess zugeordnet werden. Jeder Prozess hat ein Working Set = Menge an Seiten, die in einem letzten Zeitintervall referenziert wurden. Periodisch werden die Seiten gelöscht, die sich nicht im Working Set befinden. Wenn das ResidentSet weniger Einträge hat als das WorkingSet, werden Frames für den Prozess allokiert. Dadurch wird Paging optimiert, da zu wenig Frames viele Page Faults verursachen und zu viele Frames die Parallelität einschränken.

## Beschreiben Sie Aufgabe und Funktion eines Translation Lookaside Buffers? Worauf hat man bei der Betriebssystemimplementierung bei einem Process Switch zu achten, wenn man einen Translation Lookaside Buffer verwendet? (4)

= Cache für die Einträge einer Page Table: enthält die Nummern der zuletzt verwendeten Seiten → erhöht Effizienz: Da sich die normale Page Table im Hauptspeicher befindet, bedeutet jeder Zugriff auf das VM 2 Zugriffe auf den Hauptspeicher.
Achtung: Bei jedem Process Switch werden die Einträge des TLBs gelöscht!

## Was versteht man unter Long-Term Scheduling, Mid-Term Scheduling und Short-Term Scheduling? Erklären Sie jeden der Begriffe. (3)

Long-term Scheduling = Scheduling bei der Kreierung von Prozessen, bestimmt Grad der Parallelität, Mix von CPU- und I/O-intensiven Prozessen

Mid-term Scheduling = Ein- und Auslagern von Prozessen (Memory Management)

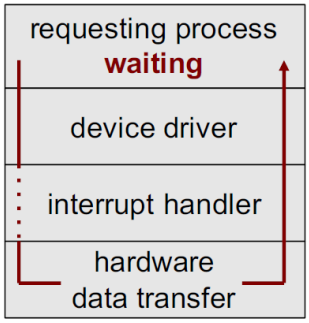
Short-term Scheduling = bestimmt nächsten Prozess der ausgeführt werden soll

## Beschreiben Sie drei Naming Strategien, die in Verteilten Dateisystemen (DFS) verwendet werden. (4)

Naming = Wie werden Namen gebildet? (Filename bleibt gleich auch wenn sich die physikalische Adresse ändert und gibt keinen Aufschluss über die physikalische Speicheradresse.)

1) Filename = Host Name + Local Name (→ eindeutig)
2) Einhängen von externen Directories in lokale Directory-Struktur (Mounten) → transparenter Zugriff auf alle Directories
3) Volle Integration: globale Namensstruktur für alle Files eines Systems

## Skizzieren Sie die Folge der Schritte, die bei der Abarbeitung eines Synchronen I/O Requests ablaufen. (5)




## Erklären Sie die Begriffe Process Switch und Mode Switch, sowie die Beziehung, in der diese beiden Konzepte stehen. Zählen Sie weiters die drei Klassen von Ereignissen auf, die einen Mode Switch nach sich ziehen. (4)

Process Switch: Umschalten des aktiven Prozesses
Mode Switch: Umschalten zwischen User Mode (eingeschränkter Zugriff) und Kernel Mode (auch Zugriff auf Kontrollregister etc.), zum Schutz von Datenstrukturen des BS

Mode Switch ist Basis für Process Switch, aber nicht jeder Mode Switch bewirkt einen Process Switch

1) Supervisor Call: expliziter Aufruf durch Programm
2) Trap: durch aktuelle Instruktion (z.B. bei Fehler)
3) Interrupt: Ursache liegt außerhalb des Prozesses

## Was versteht man unter Virtual Memory Management? Welche Vorteile bietet es? (5)

Verwendung von Speicher ohne Berücksichtigung der Größe des physikalischen Speichers, nur aktuell benötigte Seiten laufender Prozesse werden im Hauptspeicher gehalten (ResidentSet), Rest im Sekundärspeicher (Arbeitsspeicher)

Vorteile: mehr Prozesse passen in den Hauptspeicher, ein Prozess der größer ist als der Hauptspeicher passt hinein

## Welche Möglichkeiten kennen Sie, um in einem Paging-System Speicherschutz zu realisieren? (3)

= Überprüfung ob die Adresse im gültigen Bereich ist
Mittels Protection-Keys:

1) Ein Protection Key pro Frame. Prozess besitzt einen Protection Key. Bei Speicherzugriff verglichen: bei Ungleichheit → Fehler!
2) Pro TLB einen Key. Prozess besitzt eine Menge von Protection Key Registern. Bei Speicherzugriff verglichen: wenn kein Key gleich dem TLB Key → Fehler!

## Was ist Swapping? Wann wird es angewandt? (2)

= Auslagern von Prozessen in den Sekundärspeicher
wenn zuviele Prozesse im Hauptspeicher → Verschlechterung der Performance

## Nennen Sie die vier Layer eines TCP/IP Stacks und beschreiben Sie kurz deren Aufgaben. Geben Sie evtl. Beispiele an. (4)

Application Layer: entspricht im OSI-Modell Application- (Interaktion mit Applikation) + Presentation- (auflösen unterschiedlicher Formate) + Session-Layer (realisiert Verbindungen zwischen Prozessen),
verwendet z.B. FTP, SMTP, HTTP, telnet

Transport Layer: entspricht dem OSI-Transport-Layer (Transfer von Daten zwischen Knoten, Partitionierung von Nachrichten, Ordnen von Paketen),
verwendet TCP, UDP

Internet Layer: entspricht dem OSI-Network-Layer (liefert Verbindungen, für Routing zuständig),
verwendet IP, ICMP

Network-Interface-Layer: entspricht Physical- (Übertragung von Bitstreams) + Data-Link-Layer (Übertragung von Frames)
verwendet Ethernet, Token Ring,…

## Was versteht man unter einem Kernel Level Thread und unter einem User Level Thread? Beschreiben Sie die beiden Arten der Threadimplementierung und charakterisieren Sie deren Unterschiede. (4)

ULT: Threads sind für den Kernel unsichtbar, Thread Management mittels Thread Library, applikationsspezifisches Scheduling,
Vorteil: für Thread Switching kein Mode Switch nötig
Nachteile: keine Verteilung auf Prozessoren möglich, Blockierung durch System-Call blockiert alle Threads

KLT: Thread Managment durch Kernel, Thread Switching durch Kernel, Scheduling auf Thread-Basis
Vorteil: Thread-weises blockieren, multi-threading
Nachteil: Thread-Switching benötigt 2 Mode-Switches → langsamer als ULT

## Nennen Sie die Arten von Optimierungszielen, die ein Scheduler beim Prozess-Scheduling verfolgen kann und geben Sie jeweils Beispiele an. (4)

Durchsatz (SPN?), Prozessor-Auslastung (RR?), Fairness (VRR), Response Time (SRT), Einhaltung von zeitlichen Vorgaben (EDF)

## Was ist Thrashing, wodurch kommt es dazu? Wie erkennt das Betriebssystem Thrashing? Wie kann dieses Problem beseitigt werden? (4)

Durch häufige Page Faults (z.B. durch zu kleine ResidentSets) verbringt der Prozessor viel Zeit mit dem Laden von Sekundärspeicher → Einbruch der Effektivität
Problem kann z.B. durch die Working Set Strategie behoben werden

## Was versteht man unter Blocking bzw. Non-Blocking I/O? Beschreiben Sie die beiden Arten, I/O-Operationen durchzuführen. (3)

Blocking: Prozess schickt I/O-Request und wartet, bis I/O-Response da ist (vom Zustand Running auf die BlockedQueue gestellt)

Non-Blocking: Prozess schickt I/O-Request aber wartet nicht auf I/O-Response, sondern arbeitet derweilen weiter, I/O wird zu einem späteren Zeitpunkt ausgeführt, kein Blockieren

## Beschreiben Sie das Prinzip einer Sicherheitsattacke durch Stack/Buffer Overflow. Wodurch kann man sich bei der Implementierung eines Betriebssystems vor einen solchen Angriff schützen? (4)

Wenn über das Pufferende hinaus geschrieben wird, kann der Stack verändert werden. Hier kann bösartiger Code platziert werden. Am Stack kann Returnadresse verändert werden, um zu bösartigem Code zu springen.
Kann einfach verhindert werden, indem auf Buffer Overflow geprüft wird.

## Was versteht man in einem Distributed File System unter Location Transparency bzw. Location Independence. Erklären Sie die beiden Begriffe. (2)

Distributed File System
= verteilte Implementierung eines File Systems (gemeinsamer Speicher)
= verwaltet verteilte Speichereinheiten, Naming (Bildung von Namen)

Location Transparency = Filename gibt keinen Aufschluss über physikalische Speicheradresse
Location Independence = Filename bleibt gleich, auch bei Änderung der physikalischen Adresse

## Was versteht man unter einem Process Image? Erklären Sie, aus welchen Teilen ein Process Image besteht? (4)

Primary Process Table verweist auf Process Image (befindet sich im Virtual Memory), für Prozessverwaltung vom BS benötigt

- User Program
- User Data (modifizierbarer Bereich des User Space)
- System Stack
- Process Control Block (PCB)

## Nennen Sie die Bedingungen für das Eintreten eines Deadlocks und erklären Sie diese. (5)

1) Mutual Exclusion – exklusiver Zugriff auf Ressourcen
2) Hold and Wait – Prozess kann Ressourcen halten, während er auf andere Ressourcen wartet
3) No Preemption – zugewiesene Ressourcen werden dem Prozess nicht weggenommen
4) Circular Wait – geschlossene Kette von Prozessen: jeder Prozess hält mindestens eine Ressource, die von einem anderen Prozess benötigt wird

## Bei welchen der folgenden Scheduling-Strategien kann es zur Starvation kommen: (a) FCFS, (b) Shortest Job First, (c) Round Robin, (d) Priority Scheduling? Begründen Sie jeweils Ihre Antwort. (4)

a) Nein, da Prozesse in Reihenfolge abgearbeitet werden. Langer Prozess kann jedoch die CPU monopolisieren.
b) Ja, Starvation langer Prozesse.
c) Nein, langer Prozess kann sehr lange brauchen, wird aber immer ein Stück weiter abgearbeitet.
d) Ja, indem Prioritäten immer weiter absinken. Kann gelöst werden, indem Prioritäten nach einer gewissen Zeit erhöht werden.

## Was versteht man unter interner Fragmentierung und externer Fragmentierung? Beschreiben Sie die Begriffe und geben Sie je ein Beispiel an. (3)

Interne Fragmentierung: Daten füllen eine Partition nicht aus. So kommt es zu einer Speicherplatzverschwendung innerhalb der Partition. Beispiel: fixe Partitionierung (alle Partitionen gleich groß)

Externe Fragmentierung: Speicherplatzverschwendung zwischen den Partitionen durch Zerstückelung des Speicherbereichs (Beispiel: dynamische Partitionierung = Partitionen variabler Länge und Anzahl)

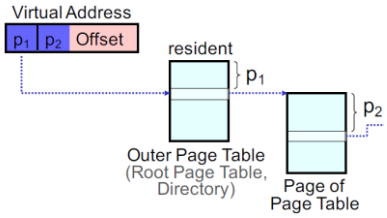
## Was versteht man unter dem Domain Name System (DNS)? Erklären Sie die Aufgabe von DNS und geben Sie ein Beispiel an, um seine Funktionsweise zu illustrieren. (4)

Löst Adressen von hinten nach vorne auf um zugehörige IP-Adresse zu erhalten (Name Server für letzten Domain - z.B. at - muss bekannt sein), schickt Anfragen an Name Server für jeden Domain → liefert Adresse des Name Servers für Sub-Domain

z.B. „tuwien.ac.at“: DNS fragt Name Server für „at“ (ist bekannt) nach Adresse von Name Server für Domain „ac.at“, fragt dann resultierenden Name Server nach Internetadresse von „tuwien.ac.at“ → liefert IP-Adresse

## Beschreiben Sie den Aufbau und Verwendung einer Multilevel Page Table (eventuell mit Skizze). Warum werden Multilevel Page Tables verwendet? (3)

Bei großen Prozessen besteht die Page Table selbst oft aus mehreren Seiten. Als Page Number speichert die virtuelle Adresse die Adresse zur Outer/Root Page Table. Diese enthält dann die Adresse der gesuchten PageTable-Seite, deren Nummer im Offset der virtuellen Adresse mitangegeben wird.




## Wodurch unterscheidet sich ein Network Operating System von einem Distributed Operating System? (4)

Network Operating System: Verteilung des OS für Benutzer sichtbar, hat spezielles User-Interface; z.B.: Datentransfer (FTP) oder Remote-Login (SSH, telnet)

Distributed Operating System: Verteilung unsichtbar (Data-, Computation-, Process-Migration)

## Nennen Sie drei Möglichkeiten, wie Distributed Mutual Exclusion realisiert werden kann. Charakterisieren Sie jede dieser Möglichkeiten kurz. (4)

1) Zentralisierter Ansatz mittels Serverprozess zur Koordination: Prozess sendet request an den Koordinator, erhält reply wenn Ressource frei ist und zugreifen darf, wenn Prozess fertig: sendet release
2) Vollständig verteilter Ansatz: Prozess sendet request an alle Prozesse, antworten mit reply wenn sie selbst keinen Request haben oder Request jünger ist, Eintritt in den k.A. sobald reply von allen Prozessen erhalten

3) Durch Token: Anordnung der Knoten in einer Ringstruktur, Token (=spezielle Datenstruktur) wird reihum weitergegeben, Besitzer darf k.A. abarbeiten

## Bei der Deadlock-Vermeidung spricht man von einem Safe State bzw. einem Unsafe State. Erklären Sie die Bedeutung dieser Begriffe. (4)

Safe State: Ressourcenbelegung ist „Safe“, wenn jeder Prozess fertiggestellt werden kann ohne einen Deadlock zu verursachen → Abarbeitungsreihenfolge erlaubt Fertigstellung aller Prozesse; bei einem Safe State ist kein Deadlock möglich

Unsafe State: keine solche Reihenfolge; Deadlock ist möglich, muss aber nicht eintreten

## Wie funktioniert das CPU-Scheduling nach dem Highest Response Ration Next Verfahren? Welche Vorteile bzw. Nachteile hat diese Schedulingstrategie? (4)

Mittels Selection Function: RR=(w+s)/s (w…bisherige Wartezeit, s…geschätzte Service Time); Prozess mit größtem RR wird ausgewählt. Non-preemptive.
Vorteile: Kürzere Prozesse werden fair behandelt, keine Starvation.
Nachteile: Zeiten müssen aber wieder geschätzt werden.