

Fragensammlung für die Prüfung der Betriebssysteme (VU - VU Betriebssysteme (Pflicht) Fall [VWVW] Verfügbare 2013-2023] 2010]

Foliensatz 11 - Betriebssysteme und Einführung

Nennen Sie zwei Abstraktionen, die ein Betriebssystem dem Benutzer bietet. Welche Aufgabe führt das Betriebssystem durch, um diese Abstraktionen realisieren?

- „ungestörte“ Programmabarbeitung Prozessmanagement
- „unendlich“ großes Speicher-Betriebssystem führt Speichergewaltung durch
- „private“ Maschine - Betriebssystem ist für Zugriffsschutz und Datensicherheit zuständig

Foliensatz 22 - Prozesse

Zeichnen Sie das Zustandsdiagramm eines Prozesses beginnend mit der Entstehung bis zur Beendigung des Prozesses. Geben Sie die Namen zu den Zuständen und beschreiben Sie kurz jeden Zustand und Übergang.



New: Zustand eines neuen Prozesses, der Prozess ist jedoch noch nicht bereit zur Ausführung

- **Ready:** bereit zur Ausführung in Ready Queue, anwartend auf Zuteilung CPU durch CPU
- **Running:** Prozess läuft gerade auf CPU
- **Blocked:** Prozess wartet auf Event (Bsp: Opt/IO Operation), sodass er weiter arbeiten kann.
- **Ready, suspend:** Ready, angegelaufen Swap in einem Sekundärspeicher
- **Blocked, suspend:** Blocked, angegelaufen Swap in einem Sekundärspeicher
- **Exit:** Zustand wird durch Terminierung der rechts Prozessinformationen/Tabelle werden gelöscht, obwohl nicht benötigt

Übergänge:

- **Admit:** Prozess aus New ist bereit zur Ausführung
- **Suspend:** Auslagnet des Prozesses im Sekundärspeicher (zur Effizienteren Nutzung des Speicherplatzes)
- **Activate:** Reintegrieren Primärspeicher (entweder in Blocked-Zustand)
- **Dispatch:** Abrufen des Prozesses Ready-Zustand Run in den Running-Zustand (CPU))

- Timeout bezeichnet die Zeit, die der Dispatcher gemäß Scheduling Strategie einen neuen Prozess reicht und dass dieser über die bereitliegenden Prozesse in die Ready Queue stecken.
- Eventwait für den Prozess bspw. I/O Operation wartet geblockt um in dieser Zeit die CPU anderweitig zu nutzen.
- Event occurs in einem bestimmten Event (I/O Operation) hat der Prozess nun wieder ready.

Wodurch unterscheidet sich die Prozessstände Ready, Ready und Blocked?

- Ready: jederzeit bereit wieder die Dispatchervorwahl zu warten und auf CPU zu arbeiten
- Blocked: wartet auf bestimmtes Event (bspw. I/O Operation um weiter ausgeführt werden zu können)

Was ist Swapping? Wann wird es angewandt?

- Swapping ist das Auslagern von Ready/Blocked Zuständen hinzu gleichartigen Zuständen auf Sekundenbasis (Festplatte suspendiert (Ready, suspendus Blocked, suspend))
- wird angewandt wann zwei Prozesse im Hauptspeicher (RAM) sind dies führt zu einer Verschlechterung der Performance

Was versteht man unter einer Prozess Control Block? Beide befinden sich in Teilen der PCB beschränkt welche Information in diesen Teilen jeweils verwendet werden.

Process Control Block ist Teil des Prozesses, Images, PCB enthält Daten, die das OS benutzt um den Prozess zu verwalten

- Process Identification (PID) des Prozesses, der Ursprung des Pads des Parent-Prozesses
- Process State Information Zustand des Prozesses, speichert Registerinhalt, Kontroll- und Statusregister und Stackpointer
- Process Control Information Zustand des Prozesses, speichert Scheduling und Priorität informationen, Queues weisen auf andere Prozesse hin, Prozesskommunikation, Memorymanagement, welche Ressourcen wurden verwendet, privilegierte, Privilegien

Geben Sie die MMU Mechanismen an, an mit denen moderne Mikroprozessoren die Aufgabe des OS unterstützen! Charakterisieren die Funktionsfides MMU Mechanismus kurz.

- Process Switching Wechsel vom Prozess zu einem anderen Prozess gemäß Scheduling Strategy
- I/O und Hardware Access Management zugehörige Mittel unter OS und I/O
- Basic Memory Management und damit unter stützende Speicher verwaltung

Erklären Sie die Aktionen, die von Betriebssysteme beim Prozess Switch durchgeführt werden. Welche Aktionen führen Ergebnisse können zu einem Prozess Switch führen?

Process Switch ist der Wechsel des aktiven Prozesses, findet statt wenn der Besitz der CPU entfällt durch System Call (interne Funktion des OS aufgerufen) oder Fehler (interner Fehler im Prozess auslösung) interne externe Störung)

- PCB muss gespeichert werden
- CPU Register und Stackträge werden gespeichert, sodass Prozess an genau dieser Stelle weiterarbeiten kann.
- Durchführen einer Interrupt/Trap/Aktion
- Umschalten der aktiven Prozesse gemäß Scheduling
- später wird Prozess wieder geladen, während alter Prozess und Prozess vorzustand wieder hergestellt

Erklären Sie die Begriffe Prozess Switch und Modus switch sowie deren Beziehung zueinander! Welche dieser beiden Konzepte steht höher im Spektrum der Klassifizierung von Ereignissen? Warum?

Prozess Switch ist der Wechsel des aktiven Prozesses statt Wechsel des Besitzes der CPU entscheidend. System Call (intern Fehler, Funktionstand der OS, aufgerufen mit Trap (interner Fehler im Prozess anweisung) interne Fehler (externer Störung))

Modus Switch (wechselt über System Call oder Interrupt zwischendurch das Wechseln vom User Modus in einen Privileged Modus. User Mode und Kernel Mode sind keine Prozess Switches möglich, insofern bedingen Modus Switches nicht unbedingt, ob es sich um einen Process Switches Modus Execution Modus Wechsel (Execution Mode existiert zum Schutz der Datensstrukturen des Betriebssystems)

Worin liegt die grundsätzliche Unterschied zwische Prozess Thread? Welcher Vorteil ergibt sich aus der Einführung von Threads für den Benutzer und was für den Benutzer achtet?

Prozesse kümmern sich sowohl die Ressourcenverwaltung als auch das Dispatching (kurzfristiges Scheduling) mit der Einführung von Threads entkoppelt man diese beiden Aufgaben. Threads kümmern sich nun um das Dispatching.

- Prozesse in virtueller Adressraum mit Prozessor, Speicher, I/O, Files
- Thread derzeitig aus Ausführung zu Stand, Stack, Kontext, thread lokale Variablen, besitzt Zugriff auf Prozessspezifische Ressourcen

Vorteile:

- Thread Erzeugung/Termination benötigt weniger Zeit,
- Umschaltung zwischen Threads schneller als Prozessswitch
- Kommunikation innerhalb Kernel
- Benutzer muss beachten Synchronisation zwischen Threads bei einer Aufgabe!

Was versteht man unter einer Prozessabbrechung? Erläutern Sie die verschiedenen Phasen in Process Image beschreibt.

- Beschreibt den derzeitigen Status/Aufbau des Prozesses, wird vom Betriebssystem Speicher verwaltung in RAM
- Process Tables zeigen auf Process Images
- im Virtual Memory
- Besteht aus:
 - Process Control Block (Prozess Identifikation, Processor state information, Prozessschichtinformation)
 - User Program enthält alle unveränderlichen Information (Programcode, etc.)
 - User Data (System Stack, User Variables gespeichert werden, und Heap zur Variablenallokation)
 - System Stack speichert Parameter und Adressen der System Calls [Shared Address Space]



Was versteht man unter einem Microkernel? Welche zentralen Services im Microkernel zur Verfügung stehen? Welche Vorteile hat der Verteilungskern gegenüber Verwendung eines Mikrokernels?

- Nur die benötigten Basiservices befinden sich im Kernel, nicht zentrale Services des Betriebssystems als Server Prozesse auf Prozessebene.

- Vorteil: ist flexibel, einfach und portabel
- Nachteil: Microkernel nicht definiert, unterschiedliche Anzahl an Services)
- Zentrale Services: Process Switching, Memory Management, Interrupts, Hardware Access und Nachrichtenaustausch. Es wird Kontrolle zwischen Prozess und Kernel

Erklären Sie den Begriff Multithreading. Geben Sie Probleme bei Verwendung von Threads an.

- Multithreading bezeichnet mehrere parallel laufende Threads pro Prozess bzw. Dies bedeutet, dass jeder Prozess mehrere Threads (Aufgaben) bearbeitet werden kann, während der Prozess selbst durch das Ressourcenmanagement und die Verwaltung betreut wird.
- Problem: Zugriff auf gemeinsam genutzte Ressourcen kann zu Wartezeiten führen.
- Thread Control Block: Thread besitzt Kontrolle über Thread

Nennen Sie die drei Kategorien von Ereignissen, mit denen die Betriebssystem die Kontrolle über das Computer-System übernimmt. Geben Sie für jede Kategorie ein Beispiel an.

- System Call: expliziter Aufruf des Programms Betriebsystem (IO, Fork, etc.)
- Trap: bedingt durch aktuelle Fehler im Programm (Fehlerhaftes Code, etc.)
- Interrupt: Ursache liegt außerhalb des Prozesses

Welcher Vorteile ergibt sich aus der Einführung von Threads für Benutzer? Warum durch kommt es zu dieser Menge von aufwändigen Benutzerebenen Programmierung von Threads ab?

Vorteile:

- Thread Erzeugungswise: Thread-Termination benötigt weniger Zeit, da dies kommt daher, dass Threads diese Ressourcenverwaltung prozessübergreifend teilen. Überlassen, Threads kümmern sich nur um das Dispatching (Prozesse übernehmen dabei das daher langsamer)
- Umschalten zwischen Threads: schneller Prozess-Switch nötig
- Kommunikation zwischen Threads: ist Kernel-Kommunikation möglich, jedoch ist eine Synchronisierung zwischen Threads nicht endig

Der Benutzer muss bei der Programmierung auf die Thread-Handhabung achten, dass die Synchronisation seine Hände liegt.

Was unterscheidet eine Kernel-Level Thread (KLT) und eine User-Level Thread (ULT)? Beschreiben Sie die beiden Arten Thread-Implementation und charakterisieren Sie deren Unterschiede.

- **Kernel Level Thread:**
 - werden vom Kernel gesteuert und verwaltet mittels Kernel API
 - Threadswitching durch Kernel (in kleinen Kernels möglich)
 - blockiert einen einzelnen Thread möglich, ganzer Prozess wird blockiert.
 - Bei mehreren Kernel-KL kann in jedem Kernel als UL ausgeführt werden, gleichzeitig!
- **User Level Thread:**
 - Threads sind für den Kernel unsichtbar
 - Threadswitching/management durch Thread Library Modus möglich
 - applikations-spezifische Scheduling

- wird Prozessortakt getaktet, bloß hier erhält Thread 1 von ULT einen blockierten Systemcall (IO/I/O), auf geweckter Basis Thread 2 des Prozesses gestoppt
- Synchronisation ist über User Mode
- kein Ausführen auf mehreren Kernen gleichzeitig

Wie unterscheidet sich das Blockieren von Kernel Threads und User Level Threads?

- ULT: Threads werden als ganzes geblockt, sobald Prozess geblockt wird
- KLT: einzelne Threads können geblockt werden, nach Priorisierungsanweisung

Beschreibe die drei verschiedenen Arten, wie Prozess-Bereich-Betriebssysteme getrennt werden können!

- Nonprocess Kernel:
 - strikte Trennung von Prozessor und Kernel
 - Prozesse sind nur Benutzerprogramme,
 - BS arbeitet getrennt von Prozessen
 - Kernel ist zentrale Schnittstelle zwischen HW/SW
- Ausführungsbereiche Betriebssystems und User-Prozessen:
 - BS ist Sammlung von Routinen, in die Prozesse sich ausführen können
 - Verlässiger Prozessmodus zum Prozessswitching
 - jeder Prozess hat eigene Kernelstack
- Prozess-basiertes Betriebssystem:
 - Betriebssystem ist Ansammlung von Prozessen, die für die Prozesse selbst
 - nur Basiservices sind keine Prozesse, alle anderen BS-Services sind eigene Prozesse

Welche Kernarchitekturvorgänge gibt es?

- Monolithic OS nutzt kleine Betriebssystemenge Mengen an Prozeduren die sich gegenseitig belieben verhalten
- Layered OS geschichtete Systeme auf hierarchische Architekturen mit benachbarten Schichten können miteinander engagiert aufbauen
- Modular OS verschiedene Ansätze realisieren die wichtigsten Funktionalitäten vom Betriebssystem, Hardware Abstraktion in einer untersten Schicht

Folienblatt 33 - Scheduling

Was unterscheidet Long-Term Scheduling, Mid-Term Scheduling und Short-Term Scheduling?

Long Term Scheduling:

- bei der Kriegerung von Prozessen aktiv
- befasst sich mit der Frage, ob ein neuer Prozess in die Ready Queue oder in die Ready-Suspend Queue kommt, also ob der Prozess von den Medium Term Scheduler oder den Short-Term Scheduler übergeben wird
- bestimmt Parallelitätgrad

Mid Term Scheduling:

- Scheduling für das Eind- und Auslageren im Sekundärspeicher (Swapping)

Short Term Scheduling:

- bestimmt, welcher Prozess aus der Ready Queue als nächster in die CPU geladen wird
- = Dispatcher

- wird bei **Interrupt** OS-Signale aufgerufen
- **Interrupt kann durch I/O Operation oder durch Timer (max Zeit zum Abarbeitung)** ausgelöst werden

Beschreibung Sie die folgenden Strategien für das CPU-Scheduling mit den folgenden Eigenschaften: FCFS, SJF, Round Robin, SPS, PS, RR, SRT, High Response Ratio, Round Robin Feedback Scheduling

First Come First Serves:

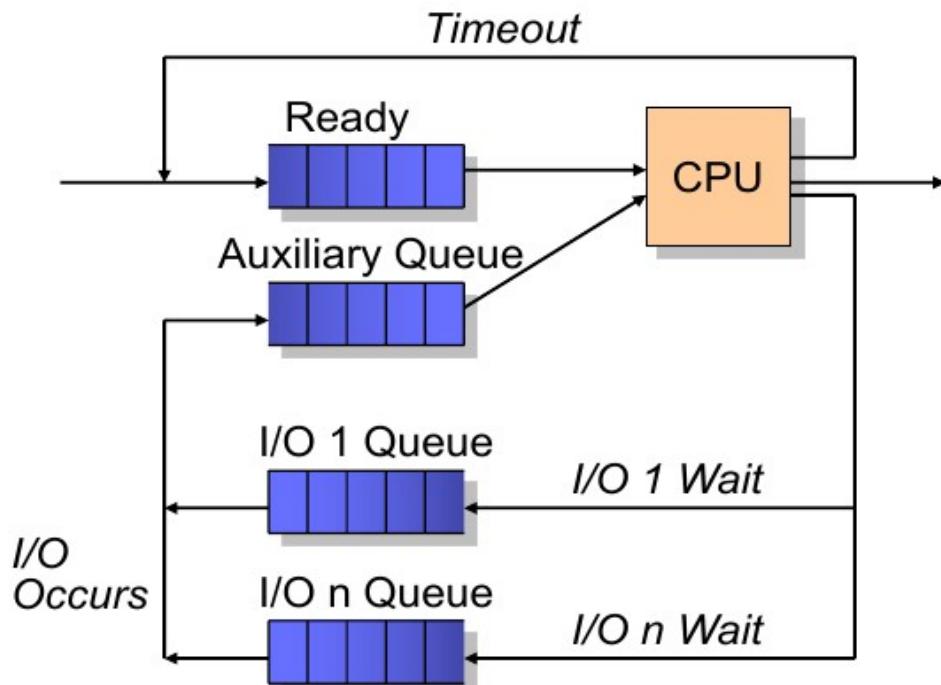
- Selection Function: **wer zuerst kommt, ist zuerst dran**
- Decision Mode: **Non Preemptive**
- begünstigt lang laufende CPU-intensiven Prozesse, während I/O intensive Prozesse immer wieder an Ende der Ready Queue geschoben werden
- ermöglicht Starvation und Endlesschleifen

Round Robin:

- Selection Function: wie bei First Come First Serves ältester Prozess in Ready-Queue wird ausgewählt
- Decision Mode: **Preemptive (unterbrechend)**
- jeder Runningprozess bekommt den gleichen (kurzen) Zeitslot, sobald Zeit abgelaufen - Prozessschwitzzy (zyklische Weiterverschiebung von Zeit slots)

Virtual Round Robin:

- Round Robin behandelt **I/O intensive Prozesse**, diese können die Zeitslots nicht voll ausschöpfen; sie verdrängen blockeende I/O vor CPU-intensiven überholt
- daher **Virtual Round Robin mit Auxiliary Queue** (besitzt höhere Priority als Ready Queue)



Shortest Process Next (SPN):

- Selection Function wählt Prozess mit **geringster CPU-Dauer aus**

- **Decision Model: Non Preemptive**
- **kurze Prozesse werden begünstigt**
- **besser Response Times als FCFS**
- **Probleme: Starvation möglich, bei Berechnung der verwarteten Programmlänge schwierig, nicht preemptiv erreichbar Blockieren der CPU**

Shortest Remaining Time (SRT):

- **Selection Function wie bei Shortest Process Next, wobei im Prozess mit der geringsten CPU Dauer aus**
- **Decision Model: Preemptive (unterbrechend)**
- **fairer im Bezug auf kürzere Prozesse als Shortest Process Next (SPN)**
- **Probleme: Starvation möglich, bei Berechnung der verwarteten Programmlänge schwierig**

Highest Response Rate Next:

- **Selection Function RR (w/ SJF) vs. [gesamte bis her gewartete Dauer / geschätzte Service Time] Priorität höher je größer RR**
- **Decision Model: Non Preemptive**
- **fairer im Bezug auf kürzere Prozesse**
- **keine Starvation jedoch bei Berechnung der verwarteten Servicetime nötigsten**

Feedback Scheduling:

- **Selection Function basiert auf bisher erzielten Ausführungszeit (je mehr CPU Time ein Prozess konsumiert hat, desto niedriger wird seine Priorität, eigene Queue für jede Priority)**
- **Decision Model: preemptive**
- **Problem: Starvation möglich, Abhilfe mittels anhebender Priority**

Erklären Sie die Begriffe Deadlock, Lifelock und Starvation. (4)

- **Deadlock durch zyklische Abhängigkeit beginn Zugriff auf Ressourcen kann keiner der beiden Prozesse die notwendigen Ressourcen anfordern, aber gibt sie auch (nicht mehr ab preemption). - Programme kann nicht weiter laufen**
- **Lifelock: Prozess wird den Eintreten des kritischen Abschnitts verzögert, kein Fortschritt möglich**
- **Starvation: ein Prozess kann auf bestreiter Ressourcen zugreifen, während immer andere vorher abgearbeitet werden**

Nennen Sie die Arten von Optimierungssystemen, die eine individuelle Prozess Scheduling verfolgen kann und geben Sie jeweils ein Beispiel an.

- **Durchsatz deshalb besser Schaltung, da ständig mehr Prozesse können nebeneinander bearbeitet werden - SRT**
- **Fairness alle Programme müssen irgendwann beendet werden, wodurch ein Round Robin**
- **Response Time: möglichst schnelle Antwort B. zB Shortest Process Next (SPN)**
- **Einhalten von Deadlines Alle (periodischen) Prozesse müssen spätestens dann beenden, wenn gefragt EDF)**
- **Prozessauslastung möglichst wenig Leerlauf, immer Arbeit Virtual Round Robin**

	User-Oriented	System-Oriented
Performance	Response Time, Turnaround Time, Deadlines	Throughput, Processor Utilization
Other	Predictability	Fairness, Resource Balance, Priorities

Bei welcher und des folgenden Schiedlungsstrategien kann Starvation auftreten? (a) FCFS, (b) Shortest Job First (c) Round Robin (d) Priority Scheduling. Begründen Sie jeweils Ihre Antwort. (4)

- a) Nein, keine Starvation, jedoch sehr lange Wartezeiten möglich
- b) Ja, da eventuell (falls es mehrere Prozesse gibt), im Verhältnis zu vor anden sein könnten
- c) Nein, da jeder Prozess eine zyklische Zeitschleife erhält. Pausiert Prozesse sind jedoch se benachteiligt daher RR
- d) Ja

Was versteht man unter Realtime-Schedulung?

- Echtzeit-Schedulung bezieht sich auf die WEC auf alle Deadlines:

 - Soft Deadline: Versäumt nicht kritisch (Treppeat (Tempo) atursensor)
 - Hard Deadline: Deadline muss unbedingt eingehalten werden, sonst Katastrofe (FCS Ato)

- üblicherweise Preemptive
- oft periodische Prozessen
- Schedulability Test überprüft Task Schedule ist also ob alle Prozesse rechtzeitig bearbeitet werden können

Erklären Sie Earliest Deadline First Schedulung!

- Ist ein Realtime-Scheidlungsverfahren
- Selection Function Task mit frühestem Deadline (Deadline abgelehnt) angegeben
- Decision mode: preemptive
- Optimierungsziel: Minimierung der verpassten Deadline
- Schedulability Test: $\sum_{i=1}^n C_i \leq n \cdot T$ ($C_i \leq 11$ (CDauer, T Periode))
- Wird immer geschaut, welches Task näher an den Deadline hat gewirkt, ausgeführt, danach wieder geschaut usw.

Foliensatz 44: Mutex & Semaphoren

Was versteht man unter einem Monitor zur Prozesssynchroisation? Nennen Sie die wichtigsten Komponenten und Eigenschaften eines Monitors!

- Der Monitor ist ein Softwaremodell bestehend aus Prozeduren, lokalen Daten und Initialisierungscode

- Prozeduren benötigen Zugriff auf Warten auf bestimmte Bedingungen (not full, not empty etc.)
- Pro Prozedur kann nur ein Prozess zugreifen
- Eigenschaften:
 - Monitor sollte für MtxEx muss nicht explizit programmiert werden
 - Shared Memory und Monitor gehen nicht zusammen
 - Zugriff auf eine lokale Variable ist mit Monitorprozedur
 - Eintritt eines Prozesses in den Monitor mittels Monitorprozeduren
 - max. 11 Prozesse gleichzeitig Monitorprozeduren
 - Bedingungsvariable ist über Monitorvariable
 - Bedingungsvariable lokal im Monitor zu greifen (wait / signal potential speichert nicht)

Für die Lösung des Brüderproblems sei gegeben, dass es sich um einen kritisch abschnitt handelt. In diesem Abschnitt werden drei Eigenschaften gefordert: (a) Wenn Sie diese Eigenschaften erklären, welche Sie deren Bedeutung. (b) Welche Wiedwendung die drei Eigenschaften gewährleistet?Semaphore zum Schutz eines kritischen Abschnitts wird verwendet?

- a) Mutual Exklusion → nur ein Prozess darf in dem kritischen Abschnitt rein
Progress → jeder Prozess muss wieder in den kritischen Abschnitt rein und darf nicht auf ewig verzögern (Keine Starvation!)
- Bounded Waiting → nach Requests für kritisches Abschließen gibt es nur eine limitierte Anzahl von Personen, die wiederein in einem
- b) Semaphore besteht aus einem Wert und einer Queue. Es wird implementiert, ob der Wert ≤ 0 oder > 0 ist, somit wird Mtx bei mehreren Prozessen verzeugt. Falls ≤ 0 , kommt der Prozess in eine (B)Q. Queue, welche die Bedingungen Prozess und Bounded Waiting sicherstellt

Gegenseitigkeit in Computer systemen durch Synchronisationsobjektum Konsolidation von Prozessen und Nachrichten zur Verfügung gestellt (gibt es eigene Semaphore oder andere Synchronisationskonstrukte). Nennen Sie zwei verschiedene Möglichkeiten, wie Sie in diesem Computersystem ein konsistentes Datenbaustein zwischen parallelen Prozessen sicherstellen können.

- blockierend und nicht blockierend Message-Passing
- Blockierend Bei Empfangen/Senden eines Messages wird solange gewartet, bis sicher gestellt werden kann, dass der Messagepartner alles empfangen hat
- Nicht-Blockierend Prozesse senden Nachrichten weiter ohne auf Antwort zu warten

Folienblatt 55 - Deadlocks

Welche Strategie zu Vorgehung gegen Deadlocks zu verwenden, um Deadlocks zu verhindern? (5)

- Deadlock Prevention beschreibt das Verhindern einer der 4 Deadlockbedingungen
 - No Hold (nicht hält) kann nicht vorkommen, wenn es ist möglich aufgrund unserer Aufgabenstellung
 - Hold & Wait (hält und wartet) Prozesse fordern Ressourcen auf, während sie Blockieren bis alle das sind. → lange Verzögerung, Prozess braucht Wissen über Ressourcen, die er Verwendet wird
 - ...

- No Preemption (indirekt) zugewiesene Ressourcen werden nicht weggenommen, Prozess gibt Ressourcen frei wenn er sie nicht bekommen kann und ist dabei bei leicht speicherbaren Ressourcen (Process Switch)
- Circular Waiting verhindert (direkt) Rückstaus durch striktere Übereinordnung bzgl. Ressourcenart, der der Forderung dem Anford. angegengesetzter die unter der Grenze liegen
- Deadlock Avoidance verhindert Deadlock, 1. diskrete Vergabe von Ressourcen
 - Process Initiation Denkt Prozess wird nicht gestartet, wenn seine Anforderungen zu einem Deadlock führen könnten $R_i \leq R_{i+1} \dots R_n < R_1$ (Claim Matrix) sehr defensiv
 - Resource Allocation Denkt Ressourcen danach freigegeben, verwendet, wenn sie zu Deadlock führen könnte. Banker's Algo (ergibt Safe State)
 - Voraussetzung: Ressourcenbedarf muss bekannt sein
- Deadlock Detection Ressourcenanforderungen werden gewährt, sofern vor anden, Algorithmus und Deadlock erkennen (ähnlich zu Banker's Algo) Strategie zur Deadlock Detektion (Recovery); Selbst Constraintiv

Bei der Deadlock Vermeidung spricht man von einer Safe State bzw. Unsafe State. Erklären Sie die Bedeutung dieser Begriffe.

Bei Deadlock Avoidance, von Resource Allocation Denkt für den sog. Banker's Algorithmus welche als Ergebnis ein "Safe State" oder ein instabile "Unsafe State" liefert. Safe: es gibt die Möglichkeit alle Prozesse abzuarbeiten, Deadlock: Deadlock ist Deadlock möglich, aber nicht zwingend möglich

Nennen Sie die Bedingungen für das Entstehen eines Deadlocks und erklären Sie diese.

- No Mutex (indirekte Deadlockvervention) kann nicht verhindert werden, ist nötig aufgrund unserer Aufgabenstellung
- Hold & Wait (indirekte Deadlockvervention) zeigt Prozess kann Ressourcen erhalten, während er auf andere wartet
- No Preemption (indirekte Deadlockvervention) zugewiesene Ressourcen werden nicht weggenommen bei leicht speicherbaren Ressourcen (Process Switch)
- Circular Waiting verhindert (direkt) Deadlockvervention) ergibt eine Kette von Prozessen, welche jeder mindest eine Ressource hält die der anderen Prozess benötigt

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

Ressourcenvergabe z.B. Deadlocks führen, könnten jedoch gefährlich bedroht (Bedingungen 1 sind erlaubt, Circular Waiting nicht). Höherer Parallelität lässt Deadlock Prevention je mal muss jedes Wissen bezüglich des Ressourcenbedarfs haben

- Process Initiation Denkt Prozess wird nicht gestartet, wenn seine Anforderungen zu einem Deadlock führen könnten $R_i \leq R_{i+1} \dots R_n < R_1$ (Claim Matrix) sehr defensiv (Claim Matrix beschreibt maximalen Ressourcenbedarf jedes jeden Prozesses)
- Resource Allocation Denkt Ressourcen danach freigegeben, verwendet, wenn sie zu Deadlock führen könnte. Banker's Algo (ergibt Safe State)

Folienatz 66 - Memory Management

Was versteht man unter Virtual Memory Management? Welche Mechanismen benötigt man zur Realisierung von Virtual Memory Management? Welche Vorbeileiste? (5)

- Unter Virtual Memory Management versteht man dynamische Adressübersetzung: logische Adressen referenzieren, virtuelle Adressen übersetzen bei fehleriger Ausführung
- Aufteilung des Speichers Pages / Segmente (richtig zusammenhängend)

- erlaubt es die nur für den Prozess relevanten Pages in der RAM zu laden
- Segementable für Überlappung
- Falls angeforderte Page den Zeit nicht im RAM (Page Fault) aus Sekundärspeicher (kann zu Thrashing führen)
- Residenz Set des Prozesses, die in RAM sind
- Vorteil: Prozess kann größer als RAM sein und darf alles gleichzeitig geladen werden sein muss eine flache Adressübersetzung

Nennen Sie Möglichkeiten, um ein Page Fault Speicherbeschreibungen zu verhindern? (3)

- Bound Check logischer Adressbereichskontrolle hinzugefügt man überprüft ob Adressen innerhalb der (maximale) Hälfte des Adressbereichs liegt.
- Bei Speicherbereichshierarchie Protection Keys
- Variante 1 jeder (physische) Frame hat einen Key, jede Prozess hat einen Key, bei Adressierung auf Frame Key Key abgleichen
- Variante 2 jeder Prozess hat Mengen von Keys, in TLB ist jeder logischen Adresse ein Key hinzugefügt, wenn der Prozess Key mit TLB Key übereinstimmt - Zutritt konsumiert Speicherbelegungsverletzung

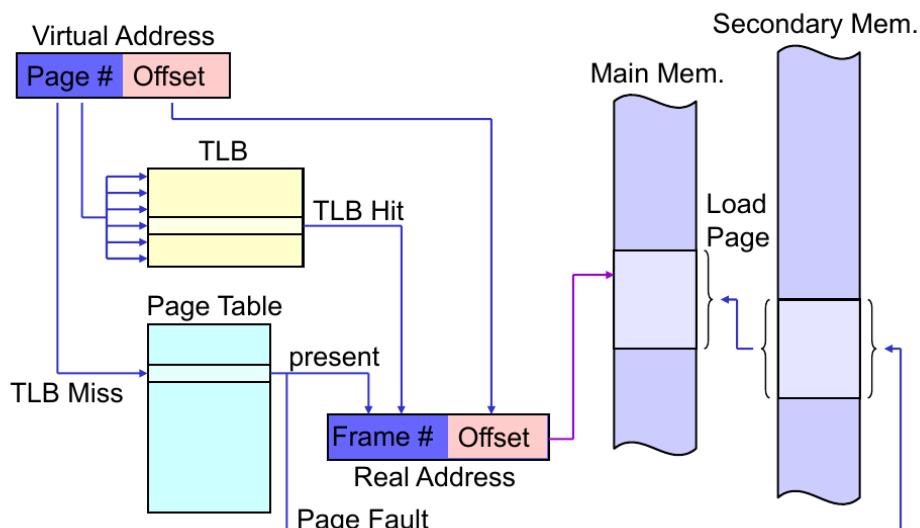
Beschreiben Sie Aufbau und Funktion eines Translation Lookaside Buffer auf Worauf hat man bei der Betriebssystemimplementierung bei einem Prozess Switch zu achten, wenn man einen Transaction Locks im Buffer verwendet?

Aufbau:

- Cache für Eingänge der Seiten (Page, Page Frame) der zuletzt verwendeten Seiten
- 16-512 Eingänge
- assoziativer Zugriff
- Löscherbeigemtem Context Switch

Funktion: möchte man die Adressübersetzung von virtuellen zur physischen Adresse, schaut man zuerst mit dem ersten Teil der Virtual Address (#Page) in der TLB nach und sucht nach der passenden Page Number. Findet man sie (TLB Hit) so hat man das assoziative #Frame (Frame number) und muss nicht mehr die Page Table schauen, da Offset bleibt gleich.

Translation Lookaside Buffer (TLB)



Bei einem Prozessswitch muss TDB gelöscht werden, da am Anfang des neuen Prozesses nicht viele Einträge vorhanden (unwahrscheinlich, TDB Hit zu treffen)

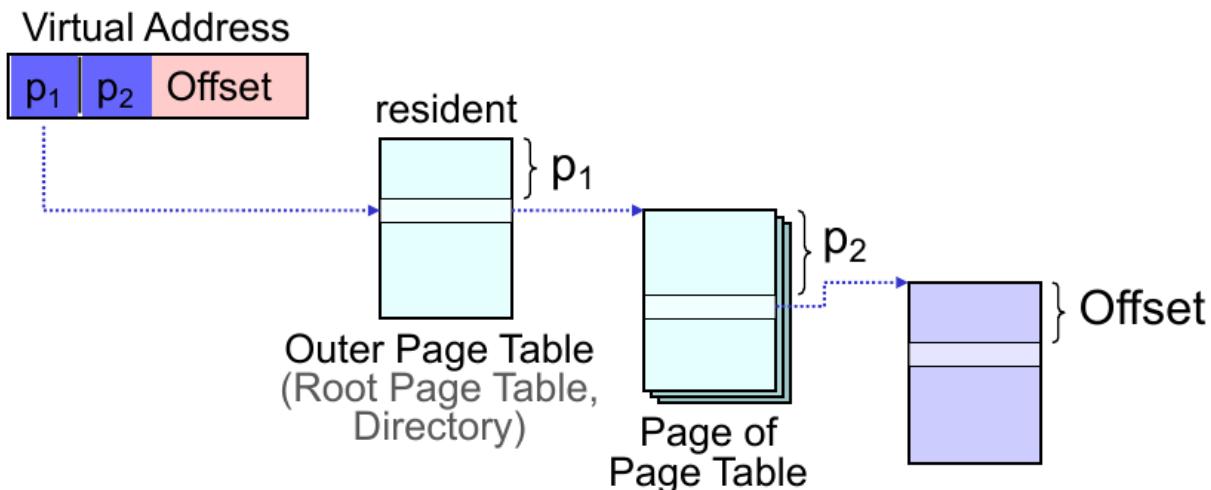
Was versteht man unter der Diff Relocation? Worin liegt die Bedeutung?

- Keine statische Fixierung des Speicherbereichs von Prozessinstanzen müssen an verschiedenen Speicherstellen (dynamisch) positioniert werden (bsp Swapping)
- Referenzen auf physikalischer Speicher müssen veränderbar sein
- Konzept: Unterscheidung zwis physisch (absolut) und logischer Adresse (Referenz unabhängig von Organisation des Speichers) [Relative Adresse oder Adresse von bekanntem Punktaus]

Wir betrachten ein System im Virtual Memory Management Diskussionen, welche der folgenden Situationen bei Referenzen an einen virtuellen Adressraum auftreten bzw. nicht auftreten kann (a) TDB Miss ohne Page Fault (b) BTB Miss Page Fault, TLB Hit ohne Page Fault, (c) TLB Hit mit Page Fault.

- a) möglich
- b) möglich
- c) muss auftreten
- d) kann nicht auftreten, TDB Hit auf statt physikalische Adresse gerade aufgerufen (und somit muss sie direkt Resident Set sein)

Beschreiben Sie den Aufbau und die Verwendung von Multilevel Page Tables (mit Skizze). Warum werden Multilevel Page Tables verwendet?



Anwendung große Prozesse mit Page Table falls Page Table nicht passt, in RAM passt, wird unterteilt in zwei Page Tables (zwingt zu mehr Zugriffen, länger)

Beschreiben Sie, wo zu und wie die Page Table verwendet wird. Geben Sie weiters an, welche Informationen in der Tabelle enthalten müssen, um eine Page Table gleich spezifiziert werden zu können. Prozesse werden in Pages unterteilt
RAM → in Frames

Author: www.00078.08.03242024

wird gemacht, dann der Prozessortritt in Hauptspeicher passiert. PageTable ist zur Übersetzung zwischen Prozessor (Pages) und RAM (Frames) da.

PageTable wird zur Adressübersetzung einer virtuellen Adresse in eine physikalische Adresse benutzt. Dabei besteht sie aus einer Menge von Adressen # (Page#) Offset Paaren. Der Offset steht dabei für die physikalische Adresse im Frame. Der Page# ist der Index in der PageTable. In diesem Index findet man dann den physikalischen Frame, unter dem Adresse Fall den Index in der PageTable vor. An diesem Index kommt es zu einer Page Fault, es wird die Seite aus dem Sekundärspeicher in den RAM geholt werden (Page Replacement).

Wozu wird die Dokumentation des Berichtsberichtsfunktionsweises?

- Ist eine Page Replacement Strategie
 - wird also angewandt um möglichst wenige Page Faults zu haben und die Optimalen Pages in den Frames zu haben
 - Funktion: Hat eine Zeiger, die auf die nächst vorerstende Seite zeigt. Seltener ist es, dass eine Seite aufgerufen wird, in der die Flag 1 aufgesetzt wird. Es wird dann erst das fügt das der Punkt zeigt. Falls eine Seite aufgerufen wird, die Flag 1 aufgesetzt hat, wird das旗 = 0 ersetzt.
 - Ist nicht viel schlechter als die LRU Strategie

Was versteht man unter der Working-Set-Strategie? Beschreiben Sie die Funktionsweise und erklären Sie, wie dieser Strategie Optimierung eines Paging-Systems genutzt werden kann.

- Ist eine Strategie, um möglichst wenig Pages zu erzeugen
 - Working Set ist $W(D, t)$, d.h. nicht zu einem Zeitpunkt allein, sondern **zurück und speichert** (variabel in der Anzahl) die in dieser Zeit mit benutzten Pages ab
 - basiert auf dem Lokalitätsprinzip
 - wächst schnell, aber stabilisiert sich & wieder schnell bei einem Prozess Switches
 - Resident Set soll dann immer bestimmen Zeitschritte Weichen Working Set orientieren
 - Problem optimales Durchkantje variert log. mit loggen sehr schwierig daher gängige Praxis
 - Beobachtung der PF/Zeitintervall und entsprechende Anpassung der Anzahl der Frames für den Prozess

Was ist IT-reshoring und woher kommt der Begriff? Wie kann das Betriebsergebnis von IT-reshoring optimiert werden?

Häufige Pageflats erzeugen beim Prozessor viele laufende Vorgänge im Sekundärspeicher in der RAM - führt zu unzuverlässigen Ergebnissen.

Befreiungsmöglichkeiten des Prozesses zu großes Residenzzeitintervall (so dass suspend und versetzen)

Erkennungsmöglichkeit zu den Seiten, die benötigt werden, um den Prozess auszuführen (Residentset zu klein)

Was versteht man unter einer Fragmentierung und externer Fragmentierung? Beschreiben Sie die Bedeutung des Begriffs mit Beispielen.

Interne Fragmentierung verschwendet Speicher im Rahmen der Partitionen z.B. bei fixer Partitionierung wird einem kleinen Speicherbereich eine große Partition zugewiesen - dadurch entsteht innerhalb der Partition mehrerer Sützen Speicherplatzes

extreme Fragmentierung oder **Zerstückelung** des Speicherbereichs ausserhalb von Partitionen, z.B. bei dynamischem Verbrauch des Speicherplatzes kann im Lesen bei Verlöschen Verteilen Speicher wiederum

dazu, dass zwischen den Partitionen kleine Bereiche überschreiten, die nicht weiter gefüllt werden können.

Beim Paging ist ein optimaler Seitenwechselalgorithmus bei Beschreibungsfehlern. Sie diesen und geben Ihnen an, warum es der Praxis nicht verwendet wird. (8)urd. (3)

OPT Algorithmus:

- erzeugt min. Anzahl Page faults
- ersetzt die am weitesten zukünftig wieder verwendet werden
- Praxis unmöglich da man nie weiß, in welcher Reihenfolge welche Pages wann verwendet werden
- Sinn nur in der Bewertung und Strategien.

Foliensatz 77 - I/O

Was versteht man unter Bufferring? Welche Vorteile bietet die gängige Gruppe und worauf kann man bei der Verwendung von Puffern im Betriebssystemmanagement zu achten?

- Zwischenkreispeicher bei I/O Transfer
- Vorteile Zusammenfassung von I/O Operationen (statt 1000 einzelne Operationen, nur 1 gesamte Entkopplung von Prozess und Bus), Swapping, Maskieren von Geschwindigkeitsunterschieden, Lastenpitzen
- Grenzen Speicherkapazität, Maskierung von Geschwindigkeitsunterschieden bei durchgehend hoher Last nicht möglich, Puffermanagement

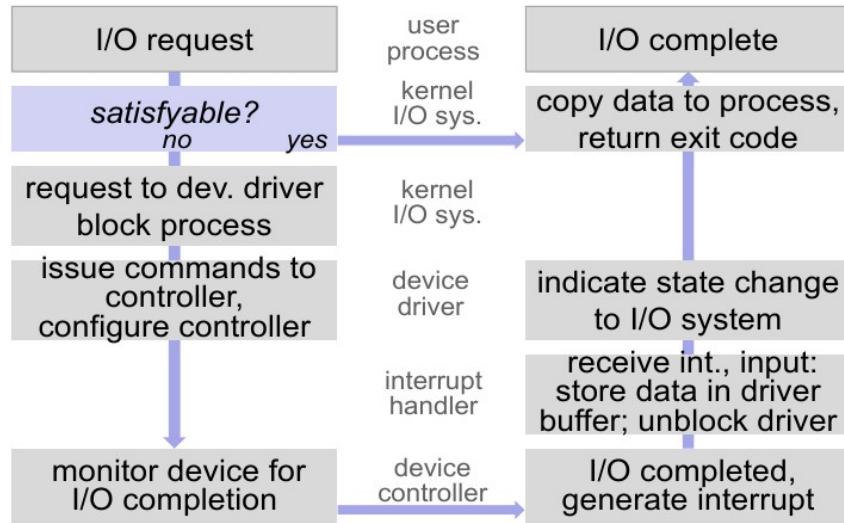
Eine Festplatte hat 500 Zylinder, die von 0 bis 499 nummeriert sind. Er wird gerade auf Zylinder 195 zugewiesen, die vorliegende Zugriffsrangfolge lautet: 195, 1470, 917, 1841, 906, 715181, 8154516206203, 17. Bei einer wie Zylinder sind die Schreibkopf vom aktuellen Position zu bewegen, um die Request folgendes zu den Strategien abzuarbeiten: (a) FCFS, (b) Elevator Algorithm (SC (SCAN) down (S-CAN))?

- (6)
- a) $1103 + (1470-92) + (1470-917) + (18417) + 91841 + 9678 - 9(17) + 181967 + (15) + 1816264 + (1620 - 1154) + (1620 - 173) = 666660$
- b) $(917 + 195) + (967 - 917) + (1456 - 967) + (10411570 + 11518 + 450) + (1670) + (1820 + 18318) + (1084 + 181620) + (1841 - 173) + (173 - 92) = 339395$
- c) 37100

Mit welcher logischen Struktur des I/O Systems kann man bei der Realisierung von I/O Funktionen sowohl die einfache als auch die Programmierfähigkeit als auch möglichst gerätespezifisch anstreben? (6)

- Schichtmodell mit 3 Schichten
 - Logical I/O: logische Bedienung der Geräte, leicht (einfach), Interface, User Input
 - Device I/O: teilt das I/O auf
 - Scheduling & control: Verwaltung I/O Operationen, maximierung der Performance

Skizzieren Sie die Erfolgsabfolge, mit der eine Abarbeitung von synchronen I/O Requests ablaufen.



Was versteht man unter Blocking bzw. Non-Blocking I/O?

Beschreiben Sie die beiden Arten, I/O-Operationen durchzuführen.

- **Blocking I/O:** Der Prozess wird vom Zustand **Running** sofort in den **Blocked** Zustand gestellt und blockiert.
- **Non-Blocking I/O:** Der Prozess wird sofort durchgeführt, liefert sofort Feedback der Operation, kann weiterarbeiten.

Was versteht man unter Synchronous bzw. Asynchronous I/O? Beschreiben Sie die beiden Arten, I/O-Operationen durchzuführen.

- **Synchronous:** Bei Synchronous wird der Ablauf des Prozesses blockiert, bis die I/O Operation wirklich durchgeführt wird (Prozess wartet bis Ausgabe am Bildschirm).
- **Asynchronous:** Hier kann der Prozess die Daten im General Buffer weitergeleben und parallel zur I/O Operation weiterarbeiten (obwohl noch nicht am Bildschirm arbeitet Prozess weiter)

Wie berechnet sich die mittlere Zugriffszeit im Lesen von Daten von mechanischen Festplatten? Gehen Sie dabei auch reale Zeitparameter einer Festplatte ein. Durch welche Strategien kann das Betriebssystem zu Leistungsfähigkeit, d.h. mit dem Zugriffszug auf die Festplatte zu verbessern?

- $T_a = T_s + T_{rd} + T_{tf}$
- $T_s \dots$ Seek Time benötigte Zeit, Disk Arm zu bringen Spur zu (Mittelwert)
- $T_{rd} \dots$ Rotationsdelay Zeitverzögerung Abstand Anfang des gesuchten Sektors gefunden $T_{rd} = 1/2r$
- $T_{tf} \dots$ Transfer Time benötigte Zeit Daten übertragen $T_{tf} = b/rN$ (b Anzahl der übertragen Bytes, N Anzahl der Bytes pro Spur, r Umdrehungsgeschwindigkeit)
- $T_a \dots$ Average Access Time (benötigte Zeit für Datenzugriff) Mittelwert

Strategien:

- **Disk Scheduling (EFO, LIFO, Priority, Shortest Service First, First Come First Served, Elevator Algorithm, C-SCAN, F-SCAN)**
- **Disk Caching** Teile des Disks sind im Hauptspeicher auszutauschen (Last-Access-Prinzip mittels Cache (Kombination LRU/LFU, B-Sections))

Beschreiben Sie das Ziel von Disk Scheduling. Nehmen Sie dort die Disk Scheduling

Algorithmen um beschreibt die Sie diskurz.

Disk Scheduling soll da helfen, die Anfragen auf Disk zu abarbeiten, dass die Seek Time möglichst kurz wird.

Intelligente AG:

- Elevator Algorithm
- C-Scan
- FSCAN

Foliensatz 88- File Management

Nennen Sie Möglichkeiten wie die Blocklegung von Dateien auf einer Festplatte repräsentiert werden kann.

- unstructured sequence of bytes
- pile: Records variablen Längen werden in Reihenfolge des Ankommens gespeichert
- sequential file lautet Record sind fixem Format, Keyfield besagt die Position innerhalb der Dateiformat
- indexed sequence falls file in fixe fiktive Zonen Zugriff
- direct (hashed) file hash Funktion über Keyfield keine sequentielle Reihenfolge der Dateien

Erklären Sie die Begriffe absolute Pfadname und relativ Pfadname und geben Sie jeweils ein Beispiel an (4)

absolut identifiziert Datei durch Beschreibung des Pfads vom Root weg: Unix
/usr/hans/mailbox/file.txt

relativ: identifiziert Datei von CD (Current Directory) aus hans/mailbox/file.txt (aus CD Sicht, hier /usr)

Wie ist ein inode aufgebaut? Welche Information enthält er? (4)

- Jeder File besitzt einen oder mehrere Inode
- Speicher Flags zu Bestimmen von Permissions, Zähler wie viele Einträge im System auf dem Node verweisen, owner/gruppe Größe des Files, Speicheradresse, letzter Zugriff, letzte Änderung
- Aufbau: Attribute (File) und Referenzen auf die Datenblöcke der Datei

Was unterscheidet eine ein FAT Allocation Table wie ist sie organisiert? (2)

- speichert die Aufteilung der Files in die der Disk (siehe Strategie der Blockallokierung)
- (Filename, Anfangspunkt, Länge) gespeichert
- wird bei Indexed Allocation genutzt (Pointert auf Daten in Tabelle)
- Vorteil: sowohl direkte als auch indirekte Zugriffe
- Nachteil: großer Platzbedarf für FAT

Bei der Realisierung von Dateisystem gibt es verschiedene Möglichkeiten, mit die zu einer Datei gehörige Datenblöcke organisiert werden können (Blockallokierung). Nennen Sie verschiedene Strategien zur Organisation von Daten und beschreiben Sie dies mit Vor- und Nachteilen.

- Contiguous Allocation: Datei Mengen an aufeinander folgende Blöcke
 - Vorteil: gute Performance beim Lesen
 - Nachteil: Probleme bei Vergrößerungen einer Datei
- Chained Allocation: Belegung einzelner Blöcke die über Zeiger verketten werden (wird in Block gespiegelt)
 - Vorteil: keine externe Fragmentierung, leichter verwitbar

- Nachteil kein Lokalitätsprinzip, langsame Zugriff daher
- Indexed Allocation wie Chain Allocation jedoch verwendet die Pointertabelle einer Tabelle (FAT), und nicht die Blöcke.
 - Vorteil: sowohl direkt als auch sequenziell Zugriff, gütz unterstützt
 - Nachteil großer Platzbedarf im RAM
- I-Nodes Datenstruktur für das ganze File speichert sowohl Attribute als auch Pointer auf andere Blöcke des Files
 - Vorteil: I-Node wird nur gebraucht sobald benutzt
 - Nachteil begrenzte Anzahl der Blöcke pro Partition und daher Verwaltung doppelter oder dritter Block

Beschreiben Sie das typische Layout einer Disk Disk eines Rechenfilesystems. Welche Rolle spielen die einzelnen Teile bei der Initialisierung des Betriebssystems? (5)

- Disk ist in Partitionen unterteilt, umblumt hängt vom Filesystems
- Master Boot Record Sektor oder (Disk) enthält Boot-Partition-Table
- Systemstar BIOS exekutiert MBR und initialisiert und der erste Boot Block wird ausgeführt (laden der OS der aktiven Partition)

Welches Dateiformat ist ergänzt um Datei für OS benötigte Bedienungswerte? Warum? Und wie wird es erkannt?

Binary Files erhalten bei jedem Bit unterschiedliche (Dateien (.exe))

Wer kann erkennt mit einer bestimten Magic Number im Header, dass das ein .exe ist. (ist spezielle Bits unter den ersten Bytes des Headers).

Foliensatz 99 - Security

Die Implementierung einer Zugriffsmaatrikeln der Form von Access Control Lists oder Capability Lists erfolgt. Erklären Sie die Begriffe. (4)

- Access Control List Zugriffsberechtigung Objekte gespeicherte Spalte, es wird nach Benutzern gepeppelt differenziert werden ein Kernel Space gehalten
- Capability List präzise gibt es eine Liste mit Objekten und den dazugehörigen Rechten ebenfalls im Kernel Space Tickets regeln Zugriff (User muss Tickets für einen Zugriff auf Objekt besitzen)

Was beschreibt das Modell Bell-Bell und Padgar? Geben Sie die Modelle für den dritten Eigenschaften an! (4)

Diese Modelle beschreiben Regeln für den Informations-Hierarchie von Security Classification für Subjekts Objekte (top secret public, secret).

Nun können die Subjekts auf den Objekts Operationen ausführen, read-only, read-write, append, execute

gefordert Security Axiome:

- simple security property Read (SC(S) ⊇ SC(O)) (no) read up
- property append (SC(S) ⊇ SC(O)) (the write down)
- property read & write (SC(S) ⊇ SC(O))

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

- **Open Design**: keine Sicherheit durch besondere schwere Codes, Sicherheitssysteme müssen verständlich lieben
- **Default-Einstellung**: keine Berechtigung (bsp: User von Haus auf keine Admin Rechte haben)
- **Least Privilege**: so viele Rechte wie möglich, die nicht benötigt werden
- **Economy of Mechanism**: Fehlervermeidung durch Einhalten der Sicherheitsmaßnahmen und Einfache Implementierung
- **Acceptability System**: ist es tatsächlich Nutzer nicht nutzen möchten
- Überprüfung ob gegenwärtigen Berechtigungen einfach einzuhalten, ansonsten gleich Berechtigungen immer konstant sind
- **Complete Mediation**: Kontrolle aller Zugriffe auf Ressourcen, Ausnahme situationen

Nennen Sie drei Kategorien von Security Threats und beschreiben Sie diese! Geben Sie für jede Kategorie an, welche geugteleges Sicherheitsziel dadurch bedroht wird. (4) grundsätzliche Unterscheidung in passives Threaten (Abhören/Monitoring) ohne Wissen des Betroffenen) und Active Threats (System/Daten werden manipuliert)

- **Denial of Service** (Interruption) übergehende oder permanente Unterbrechung eines Services (durch Zensur/überlastet) → **AVAILABILITY**
- **Exposure** (inhalt autorisierte Leserichtung) → **CONFIDENTIALITY**
- **Modification**: Veränderung der Datenintegrität (Malware, Middleman) Daten werden verändert → **INTEGRITY**

Beschreiben Sie das Prinzip einer Sicherheit im Rahmen durch **Stack Overflow**. Wodurch kann man sich bei dem Prinzip sicherstellen, dass Betriebssysteme vor solchen Angriff schützen? (4)

Funktionsweise: Wir nehmen Funktion aus, die ausgeführt werden soll, die Angabe über den Aufrufer, und somit die Return Address überschreibt, dies sagt, was als nächster (nach Aufruf) Funktion aufgerufen werden soll. Nur ist das Ziel, stark schädlichen Code zu beschreiben, und die Return Address überschreibt, sodass statt der eigenen Funktionalität durch der schädliche Code (Remote Code Execution) durchgeführt wird.

Schützen Implementierer bei den Funktionen `gets()`, `strcpy()` Platz für User-Input vorhanden ist (Länge des Inputs!)

In welcher Art von Systemen wird ein Einsatz kryptographischer Verfahren zur Sicherung der Vertraulichkeit und Integrität von Daten empfohlen? Welche kryptographischen Verfahren sind diese Systeme sicher? (3)

in offenen Systemen (wie Internet) basiert auf dem Besitz von geheimen Schlüsseln Sicherheit und Verwendung von verschlüsselten Nachrichten (Confidentiality) und Sicherheit der Nachricht (Integrity/Authenticity)