
Guia de Implementação e Análise: Sistemas Operacionais II

Docente: Marivaldo Alcantara

Foco: Engenharia de Recursos, Concorrência e Gerenciamento de Baixo Nível

1. Escopo Técnico

Esta atividade foca na camada de abstração entre o hardware e o espaço do usuário (*user-space*). O objetivo é dissecar como o kernel gerencia o estado da CPU, a hierarquia de memória e o escalonamento de tarefas sob carga.

---

2. Pesquisa de Campo e Investigação Técnica

Instrução: As respostas devem ser fundamentadas tecnicamente, evitando descrições superficiais. Analise o impacto no hardware e na latência do sistema.

Parte 1 — Ciclo de Vida de Execução (Processos e Threads)

1. **Estrutura de Processo:** Descreva a composição de um processo no Linux (focando na estrutura task_struct) e no Windows (EPROCESS). O que define o isolamento de memória entre processos?
2. **Multithreading de Baixo Nível:** Diferencie Threads de Usuário de Threads de Kernel. Como o TCB (*Thread Control Block*) se relaciona com o PCB (*Process Control Block*) durante a execução?
3. **Mecânica do Context Switch:** Detalhe o overhead gerado pela troca de contexto. O que acontece com os registradores, o Program Counter e o Stack Pointer quando o escalonador interrompe uma tarefa?
4. **Máquina de Estados:** Analise a transição entre os estados *Running*, *Ready* e *Blocked*. O que diferencia um processo *I/O-bound* de um *CPU-bound* nessas transições?

Parte 2 — Escalonamento e Performance da CPU

1. **Arquitetura do Escalonador:** Explique como o SO decide qual tarefa assume o próximo ciclo de clock.
2. **Análise Comparativa de Algoritmos:**
   - **Round Robin:** O impacto do *time slice* (quantum). O que acontece se o quantum for curto demais ou longo demais?
   - **SJF/SRTF:** A problemática da previsibilidade do próximo burst de CPU.
   - **Prioridade Dinâmica:** Como o sistema evita a inanição (*starvation*) através de técnicas como *aging*?
3. **Métricas de Vazão:** Defina *Turnaround Time* e *Waiting Time* sob a perspectiva de

otimização de throughput em servidores.

Parte 3 — Subsistema de Memória e Paginação

1. **Unidade de Gerenciamento de Memória (MMU):** Como ocorre a tradução de endereços lógicos para físicos através das *Page Tables*?
2. **Hierarquia e Page Fault:** Descreva o fluxo de exceção (trap) do processador quando uma página não está presente na RAM. Como o sistema operacional gerencia a busca no storage?
3. **Fragmentação e Segmentação:** Analise por que a paginação se tornou o padrão em relação à segmentação pura em arquiteturas x86-64.
4. **Subsistema de Swap:** Qual o impacto da latência de E/S (*Disk I/O*) no desempenho global quando o sistema entra em *thrashing*?

---

3. Laboratório Prático: Simulação e Monitoramento

Atividade A: Análise de Carga Real

Utilize ferramentas de inspeção do sistema para coletar dados reais:

1. **Linux (htop / top) ou Windows (Monitor de Recursos):**
   - Identifique a diferença entre a carga de CPU por núcleo (*Core usage*) e a média de carga (*Load Average*).
   - Observe o consumo de memória VIRT (Virtual) vs. RES (Residente). Por que a memória virtual é geralmente maior?
   - Localize um processo com múltiplas threads e verifique como o uso de CPU é distribuído entre elas.
   - *Entrega: Screenshot da análise com um breve relatório sobre o processo de maior consumo identificado.*

Atividade B: Simulação Algorítmica (Hands-on Online)

Acesse o simulador CPUScheduling.com (ou similar baseado em JS) para validar a teoria:

1. Configure três processos com os seguintes bursts: **P1: 12ms, P2: 4ms, P3: 7ms**.
2. Execute a simulação utilizando **SJF (Shortest Job First)** e **Round Robin (Quantum = 4)**.
3. **Desafio:** Calcule manualmente o Tempo de Espera Médio para ambos e compare com o resultado do simulador.
4. **Questão:** Em qual cenário o Round Robin superou o SJF? Justifique com base no tempo de resposta (*Response Time*).

---

4. Diagnóstico de Engenharia (Avaliação Crítica)

Responda com foco em arquitetura de sistemas modernos:

1. **Abstração de Kernel:** Por que sistemas operacionais modernos (Windows/Linux) não utilizam mais microkernels puros, preferindo arquiteturas híbridas ou monolíticas modulares?
2. **Concorrência vs. Paralelismo:** É possível ter concorrência em um processador de um único núcleo? Explique o papel do escalonador preempitvo nesse cenário.
3. **Deadlock em Sistemas Distribuídos:** Além das condições de Coffman, como a latência de rede complica a detecção de impasses em sistemas modernos?
4. **Volatilidade:** Por que o gerenciamento de RAM e Memória Virtual é o maior gargalo para aplicações de Big Data e Real-time Analytics hoje?