OS AARE 2S2020

Questões gerais relativas ao livro:

Todos os trechos de códigos apresentados no livro são extremamente importantes e portanto devem ser entendidos completamente. Façam os testes de mesa com cada um deles e após compilem e executem nas máquinas Linux que vocês devem ter instaladas. Esse é a malhor maneira de aprender.

Para aqueles que desejam aprender com profundidade o assunto, é muito recomendável que façam as atividades previstas ao final de cada um dos capítulos. A priori elas não serão cobradas com relação à nota final a ser atribuída à disciplina, mas são muito interessantes para aprofundamento do aprendizado.

Semana 1

Capítulo 4 – Processes

Perguntas que devem ser respondidas ao ler o capítulo:

1. O que é um processo?
2. Quais os estados de um processo?
3. É possível determinar em que ordem processos executam em um SO? Essa ordem pode mudar de execução em execução, memso se os mesmos processos estejam executando todas as vezes?
4. Quais os dados devemos armazenar para poder gerenciar processos? Determine um exemplo de estrutura com esses dados

Capítulo 5 – Process API

Perguntas que devem ser respondidas ao ler o capítulo:

1. Quais as três chamadas principais da API de um sistema UNIX são utilizadas para gestão de processos? Como cada uma delas funciona?
2. Como funciona o fluxo de um programa que crie um processo? Como diferenciar se estamos no processo pai ou no processo filho? Como se dá o fluxo de cada um deles? Dica: a questão de fluxo tem relação com o estudado no capítulo anterior.

Capítulo 6 – Mechanism Limited Direct execution

Perguntas que devem ser respondidas ao ler o capítulo:

1. Em que consiste o Limited Direct execution Protocol? Quais as questões que em que ele atua?
2. Quais as maneiras de se mudar o contexto, ou seja, o processo que está em execução?
3. Como salvar e restaurar contexto? Quais dados são importantes?

4) 4.1) Um processo pode ser considerado similar a um programa em funcionamento.

4.2) Running / Ready / Blocked

4.3) Eles seguem a TaskList

4.4) lista de processos prontos e informações adicionais p/ processos rodando e bloqueados.

5) 5.1) fork(), wait(), exec()

passo

5.2) LOAD → ALOCAÇÃO de memória → iniciar a MAIN

PID

FORK() → ALOCAÇÃO → EXEC()

6) 6.1) Olhar resumo LDE | Executar instruções diretamente pelo CPU.

6.2) Cooperativa → System Call

Ñ coop → timer interrupt

6.3) Para salvar um contexto, o S.O executa um código low-level em assembly para salvar o registrador PC e o ponteiro kernel stack do processo em andamento, depois restaura esse registrador. Os dados dos registros são importantes.