AULA 4(19-11): PROCESSOS E SEU FUNCIONAMENTO NO CONTEXTO DE SISTEMAS OPERACIONAIS.

Na primeira parte da aula, foi realizada uma introdução ao conceito de processo, que pode ser entendido a bastante grosso modo como uma instância de um programa em execução. Durante a explicação, o professor abordou conceitos como:

Diferença entre processo e programa: em resumo, um processo é uma instância em execução de um programa. Ele representa não apenas o código (programa), mas também todo o contexto necessário para que a CPU execute suas instruções de forma organizada, segura e isolada. O processo é a unidade fundamental de execução em um sistema operacional moderno.

Pilha: a pilha é uma região de memória usada pelo processo para armazenar dados de curta duração e que seguem uma organização estrita: LIFO (Last In, First Out). Sempre que uma função é chamada, o sistema aloca um *frame* na pilha contendo variáveis locais, parâmetros e o endereço de retorno. Quando a função termina, esse frame é removido automaticamente.

Heap: o heap é uma região de memória destinada à alocação dinâmica, isto é, dados que não têm um ciclo de vida previsível e são criados em tempo de execução. Diferentemente da pilha, quem controla o heap é o próprio programa, por meio de operações como malloc/free (C) ou new/delete (C++/Java/etc). A responsabilidade de liberar memória também recai sobre o programa (ou um coletor de lixo, no caso de linguagens gerenciadas).

PCB: basicamente, o PCB(Process Control Block)é uma estrutura de dados fundamental mantida pelo sistema operacional para representar e controlar cada processo existente. Ele funciona como um "cartão de identidade" do processo, contendo todas as informações necessárias para que o sistema operacional saiba em que estado o processo está, quais recursos está usando e como retomar sua execução após uma interrupção ou troca de contexto. Além disso, informações como o PID fazem parte do PCB.

Estados de processo: Os estados de um processo representam as etapas pelas quais ele passa durante sua execução no sistema operacional. Um processo inicia em new (criação), segue para ready quando está preparado para usar a CPU, e entra em running quando efetivamente está sendo executado. Caso precise aguardar algum evento — como operação de E/S ou liberação de recurso — ele passa para blocked ou waiting. Quando o evento ocorre, retorna ao estado ready. Por fim, ao concluir sua execução ou ser encerrado pelo sistema, entra em terminated. Esses estados permitem ao sistema operacional controlar a concorrência e organizar a CPU de forma eficient**e.**

Após isso, assistimos ao vídeo "*The most succesful idea in computer science"*(https://youtu.be/LDhoD4IVElk?si=yQp3BunWwx5zi-Eh) do canal Core Dumped.

O vídeo complementou o tema da aula, oferecendo uma abordagem mais didática e concreta por meio de animações.

CONCLUSÃO

Em síntese, o estudo de processos revela como o sistema operacional organiza, controla e garante a execução segura e eficiente dos programas. A compreensão de elementos como PCB, estados de execução, pilha, heap, threads e mecanismos de escalonamento mostra que a operação de um SO vai muito além de simplesmente “rodar programas”: envolve gerenciar recursos, preservar isolamento, coordenar concorrência e manter o sistema responsivo mesmo diante de múltiplas demandas simultâneas. Assim, dominar esses conceitos é fundamental para entender o funcionamento interno dos sistemas modernos e para desenvolver aplicações mais robustas, eficientes e alinhadas ao comportamento real do ambiente de execução.

Aluno: João Victor Oliveira
Matrícula: 20240008468