Sistemas Operacionais – Pesquisa

Guilherme Coimbra – 02221070

1. Defina e Cite Exemplos de **Sistemas monolíticos**;

   Sistema operacional monolítico é um termo, oriundo dos estudos referentes a sistemas operacionais, que designa o modelo em que o sistema operacional é codificado por via de uma coleção de procedimentos, onde é permitido a qualquer um deles em qualquer parte do programa interagir livremente com outro procedimento.

   A arquitetura monolítica pode ser comparado com uma aplicação que contém vários procedimentos que são compilados separadamente e depois linkados, formando um grande e único programa executável, onde todos os módulos podem interagir livremente.

   A arquitetura monolítica é a arquitetura de sistema operacional mais antiga e mais comum conhecida. Ela se caracteriza pelo fato de cada componente do sistema operacional ser contido no núcleo (kernel) e pode comunicar-se diretamente com qualquer outro componente (utilizando chamadas à função), justamente por isso o núcleo normalmente tem acesso irrestrito ao sistema de computador.

   No núcleo/kernel monolítico, o sistema operacional é escrito como se fosse um conjunto de rotinas, de forma que cada rotina pode chamar, ou ainda, se comunicar, com outra rotina, sempre que for necessário. Esse sistema possui um núcleo grande e complexo que engloba todos os serviços. Todos os componentes funcionais do kernel têm acesso a todas as suas estruturas de dados e rotinas internas, com isso, um erro em uma rotina pode comprometer todo o núcleo.

   Dessa forma, o sistema é estruturado de maneira que seja possível a interação livre entre as rotinas (a comunicação uma com as outras). Para a utilização dessa técnica é necessário que cada uma das rotinas possuam uma interface bem organizada, com parâmetros e resultados bem definidos.

   O sistema monolítico é estruturado em um único arquivo binário, em apenas um processo que executa inteiramente em modo protegido.

Exemplos:

- Os UNIX-likes (como Linux, Solaris, AIX, HP-UX, BSDs e seus derivados);
- Microsoft MS-DOS e Windows 9x;
- Apple Mac OS nas versões abaixo do 8.6.
- Atualmente, apenas sistemas operacionais embutidos usam essa arquitetura, devido às limitações do hardware sobre o qual executam.

2. Defina e Cite Exemplos de **Sistemas micronúcleo**;

   Micronúcleo, ou microkernel, é uma arquitetura de núcleo (kernel) de um sistema operativo cujas funcionalidades são quase todas executadas fora do núcleo, em oposição a um núcleo monolítico. Os processos se comunicam com um núcleo

mínimo, usando o mínimo possível o "espaço do sistema" *(kernel space)*. Neste local os aplicativos tem acesso a todas as instruções e a todo o hardware e deixando o máximo de recursos rodando no "espaço do usuário" *(user-space)* em que o software tem algumas restrições, não podendo acessar algumas hardwares, nem tem acesso a todas as instruções).

Basicamente o Microkernel executa a maioria dos processos fora do kernel, ele carrega o minimo de processos possíveis no kernel space. O restante dos módulos são executados como processos de usuário comuns para o sistema.[1]

Um dos principais benefícios de utilizar a arquitetura dessa maneira, com divisão de tarefas entre módulos, é uma menor preocupação com os erros, já que sistemas operacionais são muito sujeitos a eles. Isso não quer dizer que erros não vão acontecer, porém, erros que provavelmente causariam uma falha e derrubariam todo o sistema, agora irão apenas causar uma falha naquele módulo em específico, trazendo alta confiabilidade.

Um bom exemplo de aplicação dessa arquitetura é o MINIX 3, que utilizou muito da modularidade no seu sistema operacional. O seu micronúcleo tem cerca de 12000 linhas em C e 1400 linhas de assembly para funções de nível muito baixo, como gerenciamento de interrupções e processos de chaveamento. O código C gerencia os processos e controla a comunicação entre eles, além de permitir as chamadas para o núcleo, permitindo o sistema operacional funcionar. O driver de dispositivo para o relógio também fica no núcleo, porque o escalonador interage proximamente com ele. Os demais drivers do dispositivo operam como processos do usuário.

Além do núcleo, o sistema é estruturado em mais três camadas de processos, todas essas camadas executando em modo de usuário. A camada inferior é responsável pelos drivers de dispositivos. Como essa camada executa em modo usuário, ela não tem permissão para acessar fisicamente e emitir comandos de E/S. Ao invés disso, é necessário construir uma estrutura com as informações das portas que irão ser utilizadas e gerar uma chamada ao núcleo. Isso permite que o núcleo verifique oque o driver está lendo ou escrevendo, não deixando que, por exemplo, um dispositivo defeituoso escreva acidentalmente no disco.

Acima dos drivers está a camada que contém os servidores. Essa camada faz a maior parte do trabalho do sistema operacional, já que gerencia operações importantes do mesmo, como gerenciamento de arquivos ou gerenciamento de processos, por exemplo. Os programas de usuário enviam requisições aos servidores do que for necessário para o funcionamento do mesmo. Acima dessa camada estão os programas de usuário, que utiliza chamadas nas demais camadas para funcionar corretamente.

Uma estratégia interessante ligada ao núcleo minimalista é utilizar processos de modo usuário para atribuir prioridades aos processos, desacoplando essa função do núcleo e reduzindo-o.

Exemplos:

- Hurd
- MINIX
- QNX
- L4

3. Defina e Cite Exemplos de **Sistemas em camadas**;

O **Sistema Operacional em Camadas** é um termo oriundo dos estudos referentes a sistemas operacionais, que designa os modelos de sistemas operacionais montados sobre uma hierarquia de camadas.

O sistema operacional é organizado em camadas construídas uma sobre a outra. O primeiro sistema construído dessa maneira foi o sistema criado no Technische Hogeschool Eindhoven, na Holanda, por E. W. Dijkstra (1968) e seus alunos. O sistema THE era um sistema de lote simples para um computador holandês, o Electrologica X8, que tinha 32K de palavras de 27 bits (bits eram caros naquela época).

O sistema tinha seis camadas:

- A **camada 0** (zero) lidava com a alocação do processador, alternando entre processos quando ocorriam interrupções ou quando os temporizadores expiravam. Acima da camada 0 (zero), o sistema consistia em processos sequenciais, cada um dos quais podia ser programado sem ser necessário preocupar-se com o fato de que múltiplos processos estavam executando em um único processador. Em outras palavras, a camada 0 (zero) proporcionava a multiprogramação básica da CPU.
- A **camada 1** fazia o gerenciamento da memória. Ela alocava espaço para os processos da memória principal e em um tambor (Antigo meio magnético de armazenamento de dados) com 512K de palavras utilizado para armazenar partes do processo (páginas) para os quais não havia lugar na memória principal. Acima da camada 1, os processos não tinha que se preocupar com o fato de eles estarem em memória ou no tambor, o software da camada 1 cuidava de assegurar que as páginas fossem levadas para a memória sempre que fossem necessárias.
- Já a **camada 2** fazia a comunicação entre o console do operador e cada processo.
- A **camada 3** gerenciava dispositivos de entrada e saída.
- A **camada 4** localizavam-se os programas de usuários.
- A **camada 5** era o usuário.

No entanto, a maioria dos sistemas de uso geral usa apenas duas camadas, mesmo que o hardware em que são executados forneça mais modos de CPU do que isso. Por exemplo, o Windows 7 e o Windows Server 2008 (e seus predecessores) usam apenas duas camadas, com o camada 0 correspondendo ao modo do núcleo e a camada 3 ao modo do usuário, porque as versões anteriores do Windows eram executadas em processadores que suportavam apenas dois níveis de proteção.

Muitas arquiteturas modernas de CPU (incluindo a popular arquitetura Intel x86) incluem alguma forma de proteção em camada, embora o sistema operacional Windows NT, como o Unix, não utilize totalmente esse recurso. O OS/2 o fez até certo ponto, usando três camadas: camada 0 para o código do núcleo e drivers de dispositivo, camada 2 para código privilegiado (programas de usuário com permissões de acesso de E/S) e camada 3 para código não privilegiado (quase todos os programas de usuário). No DOS, o núcleo, os drivers e os aplicativos normalmente são executados na camada 3 (no entanto, isso é exclusivo para o caso em que drivers de modo protegido e/ou extensores DOS são usados; como um sistema operacional de modo real, o sistema funciona efetivamente sem proteção), enquanto gerenciadores de memória 386, como o EMM386, são executados na camada 0.

4. Defina e Cite Exemplos de **Máquinas Virtuais**;

Na ciência da computação, **máquina virtual** consiste em um *software* de ambiente computacional, que executa programas como um computador real, também chamado de processo de virtualização.

Uma máquina virtual (*Virtual Machine* – VM) pode ser definida como "uma duplicata eficiente e isolada de uma máquina real". A IBM define uma máquina virtual como uma cópia isolada de um sistema físico, e esta cópia está totalmente protegida.

Máquinas virtuais são extremamente úteis no dia a dia, pois permitem ao usuário rodar outros sistemas operacionais dentro de uma única máquina física, tendo acesso a outros software existentes que podem ser instalados dentro da própria máquina virtual.

5. Defina e Cite Exemplos de **Sistemas de Contêineres**;

O que são os Containers?
Containers são uma metodologia de virtualização a nível de Sistema Operacional, que permitem rodar múltiplos sistemas isolados em um único sistema real.
Sua principal diferença da virtualização comum, é que os containers conseguem compartilhar um mesmo kernel do Sistema Operacional, poupando recursos.
Com o empacotamento da aplicação em um container, é possível utilizá-la em qualquer ambiente, pois tudo que é necessário para executá-la com sucesso já está no container
Muitas vezes, os containers são usados para empacotar funcionalidades individuais, por exemplo, Mysql, phpMyAdmin, etc. Essas funcionalidades são chamadas de microserviços, que são as pequenas partes de uma aplicação divididas em serviços especializados.