Sistemas Operacionais

Conceitos de um Sistema Operacional

Modo usuário e Modo Kernel

- Como já vimos são ambientes de execução diferentes no processador
- Há um conjunto de funções privilegiadas acessadas apenas em modo Kernel
- Um programa de usuário pode usar destas funções apenas indiretamente através de chamadas de sistema(*system call*)
- Uma chamada sistema é normalmente executada através de uma biblioteca do S.O.

Processo

- Um dos conceitos chave dos Sistemas Operacionais
- Um processo é um programa em execução
- Associado ao processo está um conjunto de informações sobre os recursos que este tem direito
  - Espaço de endereços de memória
  - Arquivos Abertos
  - Contador de Programa
  - Registradores
  - Etc
- Uma definição mais abangente seria: Um processo é um conteiner que armazena todos os dados informação sobre um programa ativo

Processos

- Todo SO tem seus processos internos e processos de usuários
- Todo processo tem um número identificador chamado PID(*Process IDentification*)
- Os processos internos são executados pelo kernel do SO
- O mecanismo mais comum para criar processos em um SO multitarefa é:
  - Um processo existente crie um novo processos chamado processo-filho

Espaço de endereços

- O espaço de endereçamento de um computador é o conjunto de endereços de memória que este computador é capaz de acessar
- Comumente o espaço de endereçamento é maior que a quantidade de memória real de armazenamento
- Há técnicas para resolver esta questão.
  - Paginação e suas variações
  - Memória virtual

Sistemas de Arquivos

- O usuário está acostumado ao uso de arquivos no dia-a-dia
- Esta abstração é muito útil para o usuário final como também para os processos do sistema
- Dispositivos de armazenamento são complexos e diferentes em seus tempos de acesso e métodos de acesso
- Um sistema de arquivos permite abstrair todos esses detalhes
- Acesso a arquivos no S.O. é comumente realizado através de chamadas de sistema(*system call*)
- Exemplos de chamadas de sistema são:
  - **`read(), open(), write()`**

Sistemas de arquivos

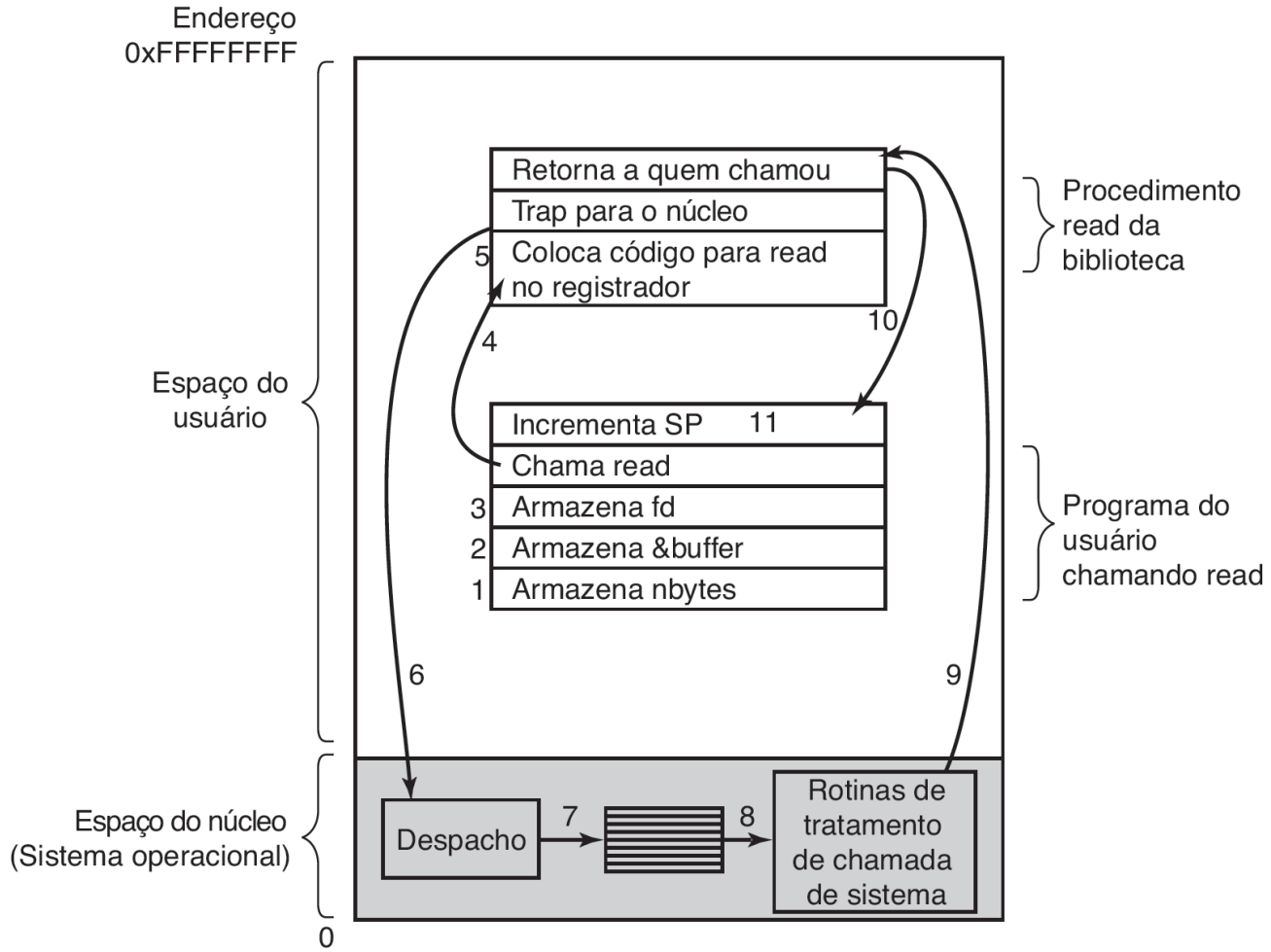


Figura 1.17 Os 11 passos na realização da chamada de sistema read (fd, buffer, nbytes).

Sistemas de Arquivos

- São comumente organizados em uma hierarquia
- Pastas contendo arquivos e outras pastas
- Há uma preocupação com permissões de acesso e acesso simultâneo de arquivos
- Arquivos comumente tem um caminho (*path*) de onde estão armazenados
  - Ex.:**`/home/tadeu/arquivos/prova.txt`**

Sistemas de Arquivos

- Em sistemas baseados em UNIX e em vários outros sistemas há alguns tipos especiais de arquivos:
  - Arquivo de blocos
  - Arquivo de caracter
  - *pipes*
- Apesar da abstração de arquivos, internamente o modo como os arquivos são efetivamente armazenados pode ser muito diferente
- Essas diferenças nos tipos de sistemas de arquivos tem relação com o objetivo de cada sistema de arquivo:
  - Otimizado para leitura, escrita, arquivos grandes, arquivos pequenos
  - Otimizados para um tipo de armazenamento: flash, HD, RAM, etc.

Sistemas de Arquivos

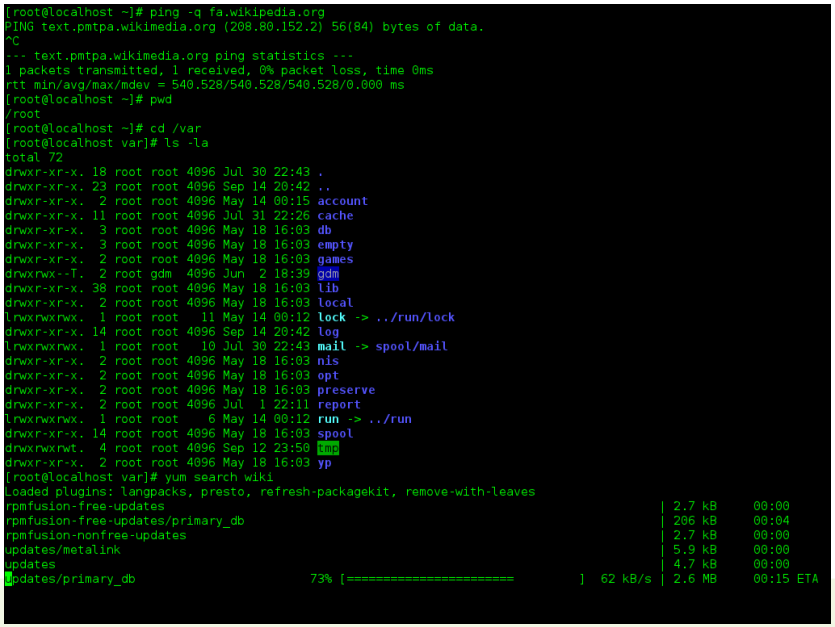
- NTFS – Sistema de arquivos do Windows
- EXT4 – Sistema de arquivos padrão no Linux
- ReFS – Novo sistema de arquivos do windows
- YAFFS – Yet Another Flash File System
- HFS – Sistemas de arquivos do OSX

Entrada e Saída

- Grande parte do que o computador faz é ler dados de entrada e escrever dados em uma saída
- Essas atividades são gerenciadas pelo S.O. usando as bibliotecas de E/S *I/O(input/output)*
- Há muitas questões que precisam ser tratadas pelo sistema de I/O
  - Velocidades dos dispositivos
  - Métodos de acesso
  - Concorrência de acesso
  - Proteção do hardware

A interface

- Hoje a maioria dos sistemas operacionais de usuário final inclui uma interface com o usuário
- Essa interface pode ser gráfica *Graphic User Interface (GUI)*, ou pode ser em modo texto *Command Line Interface(CLI)*



A interface

- Há outras maneiras de interagir com um sistema operacional
- Um exemplo são as interrupções de hardware
- Sistemas Operacionais que cuidam de sistemas robóticos podem ter interfaces apenas com os sensores instalados no equipamento

System Call

- São funções do S.O. acessíveis pelos processos de usuário
- Funcionam como chamada a funções comuns de qualquer biblioteca
- Porém são executadas pelo Kernel do S.O. e normalmente exigem estar em modo Kernel para tal

Exemplos de System Call

- **`fork()`** - Cria um novo processo
- **`open()`** - Abre um arquivo
- **`write()`** - Escreve em um arquivo
- **`mkdir()`** - Cria uma pasta no FS
- **`read()`** - Lê de um arquivo

Estrutura de um S.O.

- Há várias maneiras de se organizar um Sistema Operacional
- Cada modelo tem um conjunto de vantagens e desvantagens é preciso compreender estas diferenças
- Esta é uma decisão de projeto do S.O. mas que influenciará na maneira como processos e recursos são tratados

Estrutura de um S.O.

- S.O. monolítico
- S.O. multi camadas
- Microkernels
- Cliente-Servidor
- Máquinas Virtuais
  - JVM
- Exokernels

S.O. monolítico

- O modelo mais comum para a construção de um kernel
- Todo o código de kernel é executado como um único programa
- Essa organização pode tornar o código que executa em modo kernel muito longo e passível de mais erros
- Como tem menos camadas este tipo pode ser mais eficiente

S.O. multi camadas

- Uma generalização do S.O. monolítico
- A ideia é separar as funções do S.O. em várias camadas
- Usado no sistema THE de 1968
- Permite uma separação/segurança maior

Camada	Função
5	O operador
4	Programas de usuário
3	Gerenciamento de entrada/saída
2	Comunicação operador–processo
1	Memória e gerenciamento de tambor
0	Alocação do processador e multiprogramação

Tabela 1.3 Estrutura do sistema operacional THE.

Microkernels

- É uma estratégia inversa do kernel monolítico
- O objetivo aqui é ter o mínimo de software executando em modo kernel
- Retirando a maioria dos serviços do kernel e colocando-os como aplicações
- O principal problema desta abordagem é a quantidade de chamadas de sistema e trocas de contexto necessárias para executar algumas tarefas
- Exemplos: Symbian, Minix, QNX

Cliente-Servidor

- É uma variação do modelo Microkernel
- Cada processo é um servidor ou um cliente
- A comunicação entre processos é feita através de mensagens
- É uma boa estratégia para computação distribuída como em clusters

Máquinas Virtuais

- O conceito de máquina virtual existe há muitas décadas o OS/370 da IBM implementava este conceito
- A ideia é que há um monitor de máquinas virtuais e cada processo receberá uma máquina virtual com recursos limitados pelo monitor
- A ideia de virtualização hoje mudou no sentido de que as máquinas virtuais executam cada uma o seu próprio S.O. compartilhando recursos
- Surge o conceito de hypervisor um software responsável pela gerência das VMs

Máquinas Virtuais

- Há várias maneiras de se implementar um hypervis
- Falaremos delas mais à frente
- Este modo é muito comur em datacenters
- Esta tecnologia é a base para os sistema de computação elástica com o AWS e AZURE

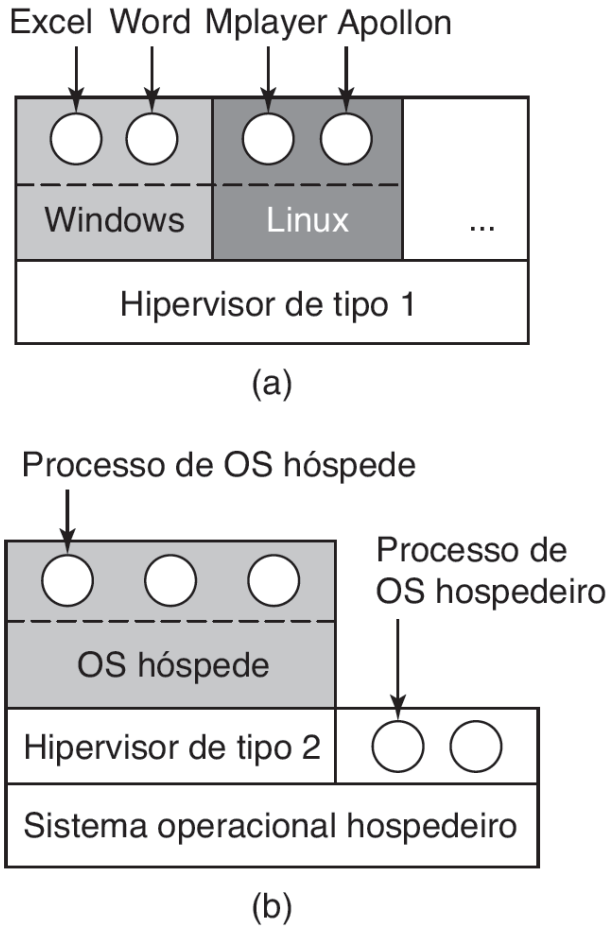


Figura 1.26 (a) Hipervisor de tipo 1. (b) Hipervisor de tipo 2.

Máquinas Virtuais

- A JVM Java virtual machine
- Estratégia criada pela SUN para que sua nova linguagem de programação pudesse executar em qualquer sistema
- É uma máquina capaz de interpretar um código intermediário e executá-lo em um sistema host qualquer
- Permite ainda que se isole aplicações JAVA de outros processos em execução.

Exokernels

- Uma outra estratégia para a separação de recursos do sistema são os exokernels
- Ao invés de solicitar que um hypervisor traduza todas as chamadas de S.O. para os S.O. convidados
- Os exokernels apenas dividem o sistema em partições e os sistemas operacionais acima que cuidam do acesso a estes dispositivos físicos
- O principal papel do exokernel é garantir que os processos não acessem recursos que não lhe pertencem

Atividade

- Descreva a diferença entre kernel Monolítico, Microkernel e Exokernel.
- Trocar o sistema de arquivos de um S.O. pode impactar sua performance? Por que?
- Defina processo.
- O que são *system calls*?Por que os processos devem usá-las para acessar o hardware?

Prazo:30/11/15 Endereço de entrega: https://goo.gl/Ih0DWq