Sistemas Operacionais: Gerenciamento de Processos e Threads

Processos e Threads - As Unidades de Execução

- **Processo:** Pensem em um processo como um **programa em execução**. Ele é uma unidade **independente** , com seu **próprio espaço de memória virtual isolado** e seus próprios recursos (arquivos abertos, registradores da CPU, etc.). Por ser completo e isolado, é considerado "pesado" para criar e para trocar de contexto entre eles. A falha de um processo geralmente não afeta outros.
- **Thread:** É uma **unidade de execução "leve" dentro de um processo**. Múltiplas threads podem existir dentro de um único processo. Elas **compartilham o mesmo espaço de memória virtual** e a maioria dos recursos do processo pai (como arquivos abertos). No entanto, cada thread tem sua **própria pilha de execução** , contador de programa e conjunto de registradores.
    - **Vantagens das Threads:** São mais rápidas de criar e de ter troca de contexto, pois são "mais leves". O compartilhamento de memória facilita a comunicação entre elas. Permitem maior responsividade em aplicações (se uma bloqueia, outras continuam) e **paralelismo real** em CPUs multi-core.
    - **Riscos das Threads:** O maior risco é a **sincronização** (condições de corrida e deadlocks) devido à memória compartilhada. O código multi-threaded é mais complexo de depurar.
    - **Falha de Thread:** Se uma thread falha, geralmente **todo o processo é encerrado**.
    - **Quando usar Processos vs. Threads:** Use processos quando precisar de **isolamento robusto** (segurança, estabilidade), para programas independentes, ou para controlar estritamente recursos. Use threads para tarefas **concorrentes dentro de uma única aplicação**, onde o compartilhamento de dados é frequente e a velocidade é crucial.
    - **Threads Nível de Usuário vs. Kernel:**
        - **Nível de Usuário (ULTs):** Gerenciadas por uma biblioteca de usuário; o kernel só vê o processo. São rápidas, mas se uma bloqueia em uma chamada de sistema, todo o processo para. Não há paralelismo real em multi-core.
        - **Nível do Kernel (KLTs):** Gerenciadas diretamente pelo kernel. Mais lentas (precisam de chamadas de sistema), mas permitem paralelismo real e não bloqueiam o processo inteiro.
        - **Modelos de Mapeamento:**
            - **N:1 (Muitos para Um):** Muitas ULTs para uma KLT.
            - **1:1 (Um para Um):** Cada ULT para uma KLT.
            - **N:M (Muitos para Muitos):** Várias ULTs para um pool menor ou igual de KLTs, buscando um equilíbrio.

Gerenciamento de Memória - Onde os Dados Residem

- **Memória Virtual:** É uma técnica que dá a cada processo a ilusão de ter seu próprio espaço de memória grande e contíguo. É fundamental para o **isolamento entre processos** , pois cada um tem sua própria tabela de páginas que mapeia endereços virtuais para endereços físicos, impedindo acesso não autorizado.

- **Paginação:** É o mecanismo base da memória virtual. Divide o espaço lógico em **páginas** e a memória física em **frames** (quadros). A **tabela de páginas** de cada processo guarda o mapeamento de páginas para frames. Isso permite que a memória de um processo seja não contígua na RAM e facilita a troca de páginas com o disco.
- **Copy-on-write (COW):** Uma otimização da memória virtual. Ao invés de copiar tudo de imediato (ex: ao criar um processo filho), pai e filho compartilham páginas de memória como "somente leitura". A cópia real da página só acontece se um deles tentar **escrever** nela. Economiza memória e acelera a criação de processos.
- **Cache de Arquivos:** O sistema operacional usa uma parte da RAM para armazenar dados de arquivos frequentemente acessados. Isso melhora muito o desempenho de I/O, pois acessar a RAM é muito mais rápido do que acessar o disco.

Escalonamento da CPU - A Dança dos Processos

- **Escalonamento de Processos:** É a tarefa do sistema operacional de decidir **qual processo (ou thread) vai usar a CPU e por quanto tempo**. É crucial para o **multi-tarefas**, para manter a CPU ocupada (utilização) e para garantir que todos os programas tenham uma chance de rodar (justiça) e respondam rápido.
- **Escalonador vs. Despachante:**
  - **Escalonador:** A parte que **decide** qual processo vai rodar em seguida, baseando-se no algoritmo.
  - **Despachante:** O módulo que **executa** a decisão do escalonador, realizando a troca de contexto.
- **Troca de Contexto:** É o processo de salvar o estado (contexto) do processo/thread atual e carregar o estado do próximo. É puro **overhead** , pois a CPU não faz trabalho útil durante esse tempo. Envolve salvar registradores, ponteiros para tabelas de páginas e pode invalidar caches. A troca entre threads é mais rápida que entre processos.
- **Critérios de um Bom Escalonador:** Maximizar a **utilização da CPU** e o **throughput** (número de tarefas por tempo). Minimizar o **tempo de turnaround** (tempo total até a conclusão) e o **tempo de espera**. Para usuários, é crucial um bom **tempo de resposta** (primeira reação do sistema). Deve ser **justo** e evitar starvation.
  - **Starvation (Inanição):** Um processo fica eternamente sem acesso à CPU ou a um recurso, mesmo que ele esteja disponível.
- **Tipos de Escalonadores:**
  - **Preemptivo:** Pode interromper um processo em execução para dar a CPU a outro (ex: por expiração de quantum ou chegada de um processo de maior prioridade). Essencial para interatividade.
  - **Não Preemptivo:** Uma vez que um processo começa, ele roda até terminar ou bloquear voluntariamente. Mais simples, mas pode gerar longas esperas.
- **Algoritmos de Escalonamento Comuns:**
  - **Round Robin (RR):** Preemptivo. Cada processo recebe um pequeno tempo de CPU, o **quantum**. Se não terminar, volta para o final da fila. Bom para sistemas de tempo compartilhado, garante justiça e resposta.

    - **Shortest Job First (SJF):** Escolhe o processo com o menor tempo de execução estimado. Ótimo para minimizar tempo médio de espera e turnaround , mas sofre com starvation para jobs longos e exige predição do tempo de execução.
    - **Por Prioridade:** A CPU é dada ao processo com a maior prioridade. Problema de starvation, resolvido com **aging** (envelhecimento: aumenta a prioridade de processos antigos).
    - **Múltiplas Filas com Realimentação (MLFQ):** Combina múltiplas filas com diferentes prioridades e quanta. Processos que usam muito CPU são "rebaixados" (feedback), e os que esperam muito são "promovidos" (aging). Versátil para diferentes tipos de carga.
    - **Lottery Scheduling:** Atribui "tickets" aos processos; o que tem mais tickets tem mais chance de rodar. Bom para compartilhamento proporcional e para evitar starvation.
    - **Sistemas de Tempo Real:** Precisam garantir que as tarefas cumpram seus prazos. Algoritmos comuns: **Rate Monotonic Scheduling (RMS)** (prioridade estática por taxa) e **Earliest Deadline First (EDF)** (prioridade dinâmica pela deadline mais próxima).

Conectividade de Rede: DNS e DHCP

DHCP - O Gerenciador de Endereços IP

- **Função Principal:** O DHCP (Dynamic Host Configuration Protocol) **automatiza e centraliza a configuração de IPs e outros parâmetros de rede** para dispositivos. Elimina a configuração manual, que é propensa a erros e tediosa em grandes redes.
- **Prevenção de Conflitos:** Evita conflitos de IP gerenciando um pool central de endereços e atribuindo-os via "leases" (concessões). Também permite **reservas de IP** (MAC para IP fixo) e exclusões de faixas.
- **Por que usa UDP?** O DHCP usa UDP (User Datagram Protocol) porque, na fase inicial de descoberta, o cliente ainda **não tem um IP** , então não pode estabelecer uma conexão TCP (que exige IPs para o handshake). Além disso, o UDP suporta **broadcast** (necessário para o cliente encontrar o servidor) e é mais leve/rápido.
- **Processo DORA (4 Passos):**
    1. **DHCPDISCOVER:** Cliente (sem IP) envia um broadcast para encontrar servidores DHCP.
    2. **DHCPOFFER:** Servidor DHCP responde com uma oferta de IP e configurações.
    3. **DHCPREQUEST:** Cliente solicita o IP oferecido (aceitando a oferta e recusando outras) ou para renovar um IP existente.
    4. **DHCPACK (Acknowledgement):** Servidor confirma a atribuição do IP e o lease.
- **Lease de IP:** É a **atribuição temporária de um IP** por um período definido (o "tempo de leasing").
    - **Importância em Redes Móveis:** Essencial para a **eficiência de IP em redes com dispositivos móveis**. Como esses dispositivos se

conectam/desconectam muito, um tempo de leasing curto permite que IPs não usados sejam rapidamente reciclados e reatribuídos, evitando o esgotamento do pool de endereços.
    - **Expiração da Concessão:** Se o cliente não renovar o lease antes do tempo expirar (tenta em 50% e 87,5%), ele **perde o IP** e não pode mais se comunicar na rede, tendo que reiniciar o processo DORA.
- **Parâmetros Configurados:** Além do IP, o DHCP pode configurar a **máscara de sub-rede, gateway padrão, IPs de servidores DNS**, tempo de lease, nome de domínio, servidores WINS/NTP, rotas estáticas, entre outros.
- **Agente Relay (IP Helper):** É um dispositivo (geralmente um roteador) que encaminha mensagens DHCP entre clientes e servidores localizados em **sub-redes diferentes**, pois roteadores não repassam broadcasts.

DNS - O Tradutor de Nomes da Internet

- **Função Principal:** O DNS (Domain Name System) é o "telefone da internet". Sua principal função é **traduzir nomes de domínio legíveis por humanos (ex: google.com) para endereços IP numéricos** (ex: 172.217.160.142), que os computadores usam para se comunicar.
- **Poluição de Cache (Cache Poisoning):** É quando um atacante **injeta registros DNS falsos no cache de um servidor DNS**, fazendo com que ele retorne IPs incorretos para domínios legítimos. Isso pode ser explorado para **redirecionar usuários para sites maliciosos (phishing)**, distribuir malware ou causar negação de serviço.
- **Servidor Autoritativo vs. Recursivo:**
    - **Autoritativo:** O servidor que **detém os registros originais e definitivos** para um domínio específico (a "fonte da verdade" para aquele domínio).
    - **Recursivo (Resolvedor):** Atua como um intermediário para os clientes. Ele recebe a consulta do cliente e se encarrega de encontrar a resposta (consultando outros servidores, como raiz, TLD e autoritativos) e depois a retorna ao cliente. Ele também faz cache das respostas.
- **Processo de Resolução de Nomes (Caminho completo):**
    1. Cliente (navegador) tenta acessar um domínio.
    2. O SO verifica seu cache DNS local. Se não achar, envia uma **consulta recursiva** ao resolvedor DNS local configurado (ex: DNS do ISP).
    3. O resolvedor local consulta um **servidor raiz** (resposta iterativa: "pergunte ao TLD").
    4. O resolvedor local consulta o **servidor TLD** (`.com`, `.org`, etc.) (resposta iterativa: "pergunte ao autoritativo").
    5. O resolvedor local consulta o **servidor DNS autoritativo** para o domínio específico, que finalmente tem o IP.
    6. O resolvedor local recebe o IP, o armazena em **cache** e o envia ao cliente.
    7. O cliente usa o IP para se conectar.
- **Consultas Recursivas vs. Iterativas:**
    - **Recursiva:** O solicitante (geralmente o cliente) pede que o servidor (resolvedor) **encontre a resposta completa** para ele.
    - **Iterativa:** Um servidor responde com a **melhor informação que tem**, indicando outro servidor para o qual o solicitante deve fazer a próxima

consulta. (É o que os resolvedores fazem ao consultar raiz, TLD e autoritativos).

- **Zona Reversa:** Mapeia **endereços IP de volta para nomes de domínio** (usando registros PTR). É muito usada para **anti-spam em e-mails** (verificação de PTR) e para leitura de logs.
- **Registros de Recursos (RRs):** São os "tipos de entradas" no DNS.
  - **A Record:** Mapeia nome de domínio para **IPv4**.
  - **MX Record:** Especifica os **servidores de e-mail** para um domínio.
  - **NS Record:** Identifica os **servidores autoritativos** para um domínio.
  - (Outros: AAAA para IPv6, CNAME para alias, PTR para reverso, TXT para texto livre).
- **DNSSEC (Domain Name System Security Extensions):** Extensões de segurança para o DNS que fornecem **autenticação criptográfica** dos dados. Protege contra **poluição de cache** e adulteração, garantindo que as respostas DNS são autênticas e não foram alteradas.