SISTEMAS OPERACIONAIS

CAPÍTULO 4 - E SE EU DESLIGAR O COMPUTADOR, PERCO TUDO?

Osvaldo de Souza

Introdução

Quando os primeiros microcomputadores surgiram, eles não possuíam muitos recursos de memória. A memória de acesso randômico (RAM) era pequena, em alguns casos chegava a 4 *kilobytes*. O SO tinha funções limitadas e a interface era muito pobre em termos de facilidades para o usuário.

Ao ligar o microcomputador, o usuário deveria aguardar que o sistema completasse a inicialização e entrasse no modo de "pronto". Esse modo era caracterizado pelo surgimento de um *prompt* ou "sinal de pronto", a forma do SO indicar que já se podia executar as tarefas.

Durante o uso, se algo desse errado e o microcomputador fosse desligado, tudo era perdido e o usuário teria de recomeçar do zero. Será que situações como essa ainda ocorrem dessa forma? Como isso poderia ser evitado?

Esse passado contrasta muito com o que se percebe atualmente. Para salvaguardarmos nossos dados, temos à disposição discos rígidos de *terabytes*, dispositivos de conexão rápida, como os discos rígidos portáteis e os *pendrives*. Se isso tudo não for suficiente, ainda há a opção de salvar nossos dados nas nuvens.

Então, voltando à questão título do nosso capítulo: e se eu desligar o computador, perco tudo? A resposta é fácil: não. Tudo estará lá quando você voltar, desde que antes de desligar, você peça para que os dados sejam gravados.

Vamos ver mais detalhes sobre o assunto no desenvolvimento deste capítulo. Bons estudos!

4.1 Gerenciamento de memória cache

Vimos que no passado, o microcomputador era um equipamento precário, com poucos recursos, mas que evoluiu bastante, baseado em tecnologias que permitiram que os componentes eletrônicos ficassem cada dia menores e mais baratos.

Naquela época, como o usuário poderia pedir para que o SO carregasse algum aplicativo? De fato, essa questão envolvia um bocado de trabalho, pois o acesso às unidades de armazenamento externas era caro e limitado (DEITEL; DEITEL, 2011).

Basicamente havia três caminhos: fornecer o aplicativo a partir do teclado, ou seja, digitando todos os comandos necessários, todas as vezes que o aplicativo necessitasse ser executado; usar um gravador de fita K7, no qual tivesse gravado previamente uma cópia do programa; e usar um disco flexível, inserindo-o em alguma unidade leitora.

Funcionava deste jeito: você digitava o programa e o testava, se tudo corresse bem, você deveria retirar o disco de inicialização do SO (caso tivesse sido usado na inicialização do microcomputador), depois inserir um novo disco com espaço para salvar o seu programa. Após esse procedimento, você teria seu programa protegido em um disco flexível. Era algo parecido como se estivesse usando um gravador de fita K7, mas as chances de algo dar errado eram muito grandes. Vale lembrar que uma fita K7 permitia a gravação de uma pequena quantidade de *kilobytes*, enquanto um disco flexível de 3.1/2" pode chegar a 1.2 *megabytes*.

Dentre as evoluções surgidas, estão os equipamentos ligados à memória, seja de acesso randômico (RAM) que é a memória principal de um microcomputador, até as secundárias, tais como os discos rígidos e os discos rígidos portáteis.

VOCÊ O CONHECE?

Von Neumann é reconhecido no cenário da Computação como uma pessoa de grande importância, em especial com suas contribuições na área da arquitetura de computadores. De fato, os microcomputadores atuais, todos, sem exceção, possuem influências de seus trabalhos. Para conhecer mais sobre ele, a recomendação é a leitura do artigo (KOWALTOWSKI, 1996), disponível em: < http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-40141996000100022>.

No Quadro a seguir, podemos ver a atual hierarquia de memórias e uma relação de equipamentos de entrada e saída, de forma a ter um panorama geral da questão.

No topo, vemos os registradores e cache nível 1 e nível 2. Esses três itens, que são três tipos de memória, estão integrados, fazem parte da CPU e de dois grandes grupos: a memória volátil, memória que mantém o estado quando alimentados por energia elétrica - registradores, cache e RAM; e a memória persistente ou não-volátil: memória que mantém o estado mesmo sem energia elétrica – HD, *pendrive*, disquete, dispositivos de armazenamento.

Após as memórias cache encontramos a RAM, ali caracterizada como a memória física (real) e a memória virtual. Na continuação do Quadro, temos os equipamentos de armazenamento de dados e, por fim, os equipamentos de entrada de dados.

Os registradores e as memórias cache são ligados e diretamente relacionados à CPU (DEITEL; DEITEL, 2011), além de apresentarem um tempo de funcionamento extremamente pequeno, portanto são dispositivos que permitem uma transferência de dados em grande velocidade (DEITEL; DEITEL, 2011). Os registradores normalmente são acessados cerca de 1 ns (nanosegundo), já a memória cache tem velocidade de acesso típico próximo de 2 ns. Por fim, a memória RAM apresenta um tempo de acesso na casa de 10 ns. Perceba pela diferença de tempos que as memórias caches são extremamente rápidas e é necessário que sejam assim para que possam ser utilizadas pela CPU sem que seus respectivos tempos afetem o desempenho.

Observe que existe uma grande quantidade de tipos de memórias, subdivididas entre as voláteis e as não-voláteis. E entre esses tipos de memórias também há diferenças, em relação à velocidade. A escolha de qual será instalada no microcomputador é importante, pois afetará o desempenho da máquina.

VOCÊ SABIA?

Para fins didáticos, 1 ns é igual a 0,00000001 segundos, e o ms é igual a 0,001 segundos. Veja a seguinte comparação: você caminhando normalmente tem uma velocidade cerca de 4 km por hora, e um avião viaja a cerca de 850 km por hora. Fazendo as contas, percebemos que o avião é cerca de 200 vezes mais rápido do que você. Mas se a sua velocidade fosse de 1 milisegundo e a do avião 1 nanosegundo, o avião seria 100.000 (cem mil) vezes mais rápido. Logo, a memória cache é 100.000 vezes mais rápida do que a memória RAM.

Tendo em vista que a memória cache é mais rápida que a RAM, ela assume uma função relacionada ao armazenamento de itens (código ou dados) que estão sendo muito utilizados, de maneira a acelerar a execução dos processos. Quando um dado ou parte do código de um processo não se encontra na memória cache, ocorre um novo acesso à RAM.

Tipo	Nome
Memória Volátil	Registradores da CPU
Memória Volátil	Cache L1
Memória Volátil	Cache L2
Memória Volátil	RAM
Memória Volátil	Memória Virtual
Memória Não-Volátil	ROM / BIOS
Memória Não-Volátil	Disco rígido
Memória Não-Volátil	CD / DVD
Memória Não-Volátil	Dispositivos portáteis USB
Dispositivo de Entrada	Teclado
Dispositivo de Entrada	Apontador (*mouse*)
Dispositivo de Entrada	*Scanner*, câmera, microfone, etc.

Figura 1 - Hierarquia dos tipos de memória e dispositivos de E/S, no início do quadro, os registradores e as memórias cache que pertencem à CPU.
Fonte: Elaborado pelo autor, 2018.

Figura 2 - Itens que estão cada vez mais caminhando para a obsolescência, ou já nem são mais utilizados, como o disco flexível de 3.1/2", à direita.
Fonte: Moon Light PhotoStudio, Shutterstock, 2018.

Dessa forma, quando um processo é escalonado para a execução, uma parte significativa é carregada para a memória RAM. Todavia, vimos que a memória RAM tem um atraso de 10 ns para ser acessada. Isso seria como se um avião tivesse que parar nos semáforos. Assim, para minimizar a perda de tempo e o poder de processamento da CPU, quando um processo é lido, parte (ou todo) do fragmento lido é copiado (de uma vez) da memória RAM para a cache.

4.1.1 Conceitos e funcionalidade

Quando a CPU for executar a próxima instrução do processo, ela busca essa instrução da memória cache (que, como vimos, é muito mais rápida do que a RAM). Se a próxima instrução não estiver na cache L1, verifica-se se está na cache L2, se não estiver, então ela é recuperada na RAM, e não apenas ela, mas um outro fragmento do processo, que é armazenado na L1 inicialmente. Mas quando há algo na cache que não está sendo utilizado (código ou dados), ele sai da L1 e fica na L2. Se também não for usado por muito tempo na L2, então é descartado. Portanto, a função das memórias caches é manter por perto e à disposição da CPU os códigos e dados que estiverem sendo muito utilizados. Assim, evita-se um custoso acesso à memória RAM e, pior ainda, à alguma memória secundária.
Com relação à velocidade da memória secundária, enquanto as memórias caches têm acesso na casa dos 2 ns e o acesso à memória RAM em torno de 10 ns, o acesso ao disco rígido ocorre em torno de 10 ms (milissegundos) (DEITEL; DEITEL, 2011).
O próximo Quadro apresenta uma classificação dos tipos de memória de um típico microcomputador.

Tempo de acesso	Tipo de uso	Capacidade Típica
1 ns	Registrador	< 1 *kilobyte*
2 ns	Cache	~ 4 *megabytes*
10 ns	RAM	1 - 8 *gigabytes*
10 ms	Disco rígido	1 - 4 *terabytes*

Figura 3 - As diferenças de tempos entre os dispositivos de armazenamento de dados do tipo memória.
Fonte: Adaptado de TANENBAUM; BOS, 2016.

Qual o impacto de um microcomputador não ter uma memória cache? Seria a diminuição drástica da velocidade de processamento do conjunto, algo em torno de 10 vezes mais lento (DEITEL; DEITEL, 2011), tendo em vista que a diferença entre a velocidade da RAM e das memórias cache é cerca de 10 vezes.

VOCÊ QUER LER?

Nos primeiros computadores, a programação, da forma que conhecemos e entendemos, não existia. O equipamento era produzido para um fim específico, portanto era altamente especializado em uma única tarefa. Hoje, nossos microcomputadores realizam uma infinidade de tarefas sendo programáveis. A preocupação em produzir um equipamento capaz de ser programado remonta ao ano de 1945, no contexto do projeto EDVAC. Para conhecer essa história, acesse o relatório confidencial do projeto (ECKERT; MAUCHLY, 2006) disponível em: < http://archive.computerhistory.org/resources/text/Knuth_Don_X4100/PDF_index/k-8-pdf/k-8-u2736-Report-EDVAC.pdf>.

Então, por que não usamos apenas memória cache ao invés de memória RAM? Essa é uma pergunta comum e a resposta é simples: o preço das memórias caches em relação à RAM acompanha a diferença de velocidade.
Considere agora o impacto que há quando o sistema operacional opera com características de paralelismo. Como sabemos, um dos grandes trabalhos realizados pelo SO é relativo à troca de contexto entre os processos. Os processos são trocados em função das decisões do algoritmo de escalonamento que estiver sendo usado pelo sistema operacional. Em sistemas com preempção por interrupções e compartilhamento de tempo, o nível com que o sistema operacional troca de processo em execução é muito alto.
Considere agora um volume alto de trocas de contexto que requerem que os dados e o código de programa dos processos sejam constantemente carregados para execução, salvos na troca de contexto e novamente carregados para continuar a execução.
Sem o uso e apoio das memórias cache, o desempenho seria muito afetado e certamente não teríamos os equipamentos rápidos como os que nós costumamos ver hoje em dia.
Ainda assim, até o momento, não conseguimos fugir da necessidade de utilizar memórias secundárias em disco rígido, que são absurdamente lentas se compararmos com a velocidade da RAM e das memórias cache.

VOCÊ QUER VER?

Como você deve saber, o computador não surgiu da forma como o vemos hoje, houve um enorme processo de evolução que culminou nos modernos equipamentos que estamos acostumados a utilizar. Grande parte dessa evolução tem relação com o processo de fabricação, que também foi modernizado. Você já viu como é feito uma unidade de processamento central? Vale a pena conferir o *Processo de fabricação do Intel Core i5* (INTEL, 2014) em: < https://www.youtube.com/watch?v=2W6kCXwSZnk&feature=youtu.be>.

Se houvesse um novo tipo de dispositivo de armazenamento secundário que pudesse ser usado em substituição aos discos rígidos e que apresentasse uma velocidade semelhante à das memórias RAM e cache, estaríamos em um outro patamar de desempenho da computação.

4.1.2 Segmentação

O conceito de paginação com o qual trabalhamos até este ponto consiste em uma abstração da memória virtual, dividida em frações e fornecidas aos processos. Todavia, essa estratégia pode representar dificuldades quando para um processo foi alocado uma página de memória virtual e durante o processamento passa a ser necessária mais memória. Para resolver essa e outras situações, uma nova abstração foi introduzida na qual a memória é segmentada e cada segmento pode possuir espaços de endereçamentos completamente independentes. Dessa forma, um segmento pode ter tamanho arbitrário, isto é, de zero até algum valor máximo permitido.

Usando-se essa abstração, as porções de memórias alocadas aos processos podem crescer ou diminuir de tamanho em função das solicitações dos processos.

Segmentos podem ser criados simplesmente para alocar estruturas de dados, tais como pilhas, variáveis, listas etc. de forma que na conversão dos códigos-fontes dos aplicativos, essas estruturas de dados sejam criadas de forma que possam ser suportadas em diferentes segmentos de memória.

A seguir, vamos conhecer o que é e como funcionam o gerenciamento de entrada e saída.

4.2 Gerenciamento de entrada e saída

Para que os processos sejam executados, quase sempre é preciso que eles tenham acesso a recursos, necessários para a escrita ou leitura de dados. A leitura de dados está relacionada ao conceito de entrada de dados, enquanto a escrita fica associada à saída de dados. São verdadeiros desafios ao sistema operacional e a forma de gestão compete ao gerenciamento de entrada e saída.

Com o aperfeiçoamento dos microcomputadores, houve também uma evolução paralela de certos tipos de dispositivos, como os de entrada e saída de dados para registro em massa. Por registro em massa estamos nos referindo a grandes quantidades de dados. A Tabela a seguir apresenta alguns desses dispositivos usados desde o surgimento do microcomputador e do sistema operacional.

A capacidade de armazenamento mudou muito em relação aos quesitos velocidade, confiabilidade e, principalmente, quantidade de dados que podem ser guardados em tais dispositivos. De fato, muito do que se faz atualmente seria impossível se não houvesse equipamentos que suportassem a enorme quantidade de dados que é necessária, por exemplo, para a simples existência de um filme guardado no microcomputador (DEITEL; DEITEL, 2011).

4.2.1 Interrupção, acesso direto à memória

Os diversos dispositivos que permitem a realização da entrada e saída de dados são controlados pelo sistema operacional, mas esse controle é parcial, pois alguns operam de maneira completamente separada do microcomputador. No caso desse tipo de equipamento, é necessário algum mecanismo que permita uma forte relação entre o equipamento de entrada e saída, o sistema operacional e o *hardware*.

Esse trio deve ser sincronizado, é preciso que o equipamento externo possa comunicar a CPU que ele precisa transferir dados, que ele está pronto para ser usado, ou que acabou de ocorrer alguma situação que merece atenção do sistema operacional.

Uma solução é permitir e disponibilizar mecanismos sólidos para que, quando um dispositivo precise de atenção do sistema operacional, ele possa lançar mão desse recurso.

Nos microcomputadores modernos, este mecanismo é conhecido por **interrupção,** sendo, de fato, uma requisição de interrupção ou *interruption request* (IRQ) (TANENBAUM; BOS, 2016). Quando uma IRQ ocorre é sinal de que algum fato relevante aconteceu em alguma parte do *hardware* e precisa de atenção. Ignorar uma IRQ, isto é, não dar o tratamento adequado para uma solicitação de interrupção, pode levar o equipamento a uma perda de dados ou até o travamento total.

Os discos rígidos também utilizam as interrupções sempre que concluem uma operação de escrita ou leitura e para informar ao sistema operacional que a tarefa foi finalizada. Assim, quando um processo precisa de dados que estão em um disco rígido, ele solicita ao sistema operacional que coloca o processo no estado de **esperando**. Depois disso, o sistema operacional providencia a solicitação de leitura de dados ao disco rígido. Enquanto o disco rígido providencia os dados solicitados pelo sistema operacional, este estará livre para escalonar outro processo. Quando, enfim, os dados estiverem disponíveis, o disco rígido gerará uma IRQ que notificará o sistema operacional, que então, obterá os dados e os disponibilizará para o processo que havia solicitado, ocasião na qual terá seu estado alterado de **esperando** para **pronto** e será escalonado pelo sistema operacional de acordo com o algoritmo que estiver em uso.

Isso que acabamos de apresentar é parte da comunicação e notificação entre o equipamento de entrada e saída e o sistema operacional. A outra parte refere-se propriamente à transferência dos dados.

Se uma CPU precisasse ficar totalmente envolvida na transferência de dados, digamos entre um disco rígido e a memória, haveria uma enorme sobrecarga de processamento e o tempo de transferência de dados iria “roubar” o tempo de processamento de fato. Na verdade, os primeiros microcomputadores funcionavam dessa maneira e durante as leituras e escritas de grandes volumes de dados, a máquina, literalmente, não fazia outra coisa e o usuário tinha a impressão de travamento.

Nome do Dispositivo	Período de uso aproximado	Capacidade predominante	Custo	Tempo de acesso	Confiabilidade
Fita K7	Década de 1980 e parte da década de 1990	Poucos *kilobytes*	Preços mínimos	Dependia da velocidade de gravação. Lento	Pouco confiável
Disco 5.1/4"	Década de 1990	160 ou 360 *kilobytes*	Preços mínimos	Cerca de 30 *kilobytes* por segundo	O disco era facilmente danificado
Disco 3.1/2"	Década de 1990 e posterior	1.44 *megabytes*	Preços mínimos	Cerca de 60 *kilobytes* por segundo	Vários anos, se manuseado adequadamente
Disco Compacto	Década de 1990 e atualmente	Variável, em torno de 700 *Megabytes*	Preços mínimos	Taxas de transferência variáveis para escrita e leitura. Medianamente rápido	Teoricamente vários anos, mas a mídia era muito frágil e tinha de ser manuseada com muito cuidado
Disco rígido	Década de 1990 e atualmente	1 GB - 1990 200GB - 1990 1000GB - 2007	Variáveis, dependendo da capacidade, podendo chegar a milhares de reais	Taxas variando de 1.7 até cerca de 30 *megabytes* por segundo. Rápido	Durável, e por permanecer dentro do microcomputador, bem protegido

Quadro 1 - Evolução dos principais dispositivos de armazenamento de dados usados nos microcomputadores, destacando-se as respectivas capacidades de armazenamento de dados.
Fonte: Elaborado pelo autor, 2018.

VOCÊ QUER LER?

Quando tratamos do escalonador de tarefas e da próxima instrução a ser executada pela CPU, estamos nos referindo, de fato, às instruções que são "compreendidas" pelo processador. No processador, as instruções são representadas por códigos numéricos e há um número bem pequeno delas. Para entender o assunto, a recomendação é o livro "*Programming Ground Up*" (BARTLETT, 2003) sobre a linguagem *assembly*, disponível em: <http://nongnu.askapache.com/pgubook/ProgrammingGroundUp-0-9.pdf>.

Nos equipamentos modernos foi introduzido o conceito de acesso direto à memória, ou DMA, que permite que tais dispositivos utilizem um canal exclusivo para a transferência de dados, desocupando parcialmente a CPU. O resultado é que as transferências de dados quase não ocupam tempo de processamento, portanto o microcomputador, como um todo, apresentará um desempenho idêntico, mesmo durante a execução de processos que façam intenso uso de dados.

4.2.2 Deadlock

Você já tem conhecimento das condições de corrida que ocorrem quando processos diferentes disputam o mesmo tipo de recurso e das estratégias para resolver os problemas de acesso às regiões críticas dos processos, relacionados às condições de corrida. Todavia, essa não é a única situação em que a competição por recursos, por parte dos processos, pode resultar em problemas.

Vamos supor que X necessita dos recursos A e B, dos quais ele já possui privilégio de uso de um dos recursos e, neste caso, digamos do A. Como o processo X precisa de A e B, e ele já possui A, resta aguardar até que o recurso B seja liberado e alocado pelo sistema operacional a X. Assim, X poderá realizar o processamento pretendido. Mas considere a situação particular na qual o processo Y já possui o acesso ao recurso B e está dependendo do recurso A que está bloqueado.

O resumo da situação é: X depende que Y libere A, enquanto Y somente liberará B quando X liberar A. Isso leva os processos a uma situação bloqueante sem solução. Ambos estão condenados a nunca serem concluídos.

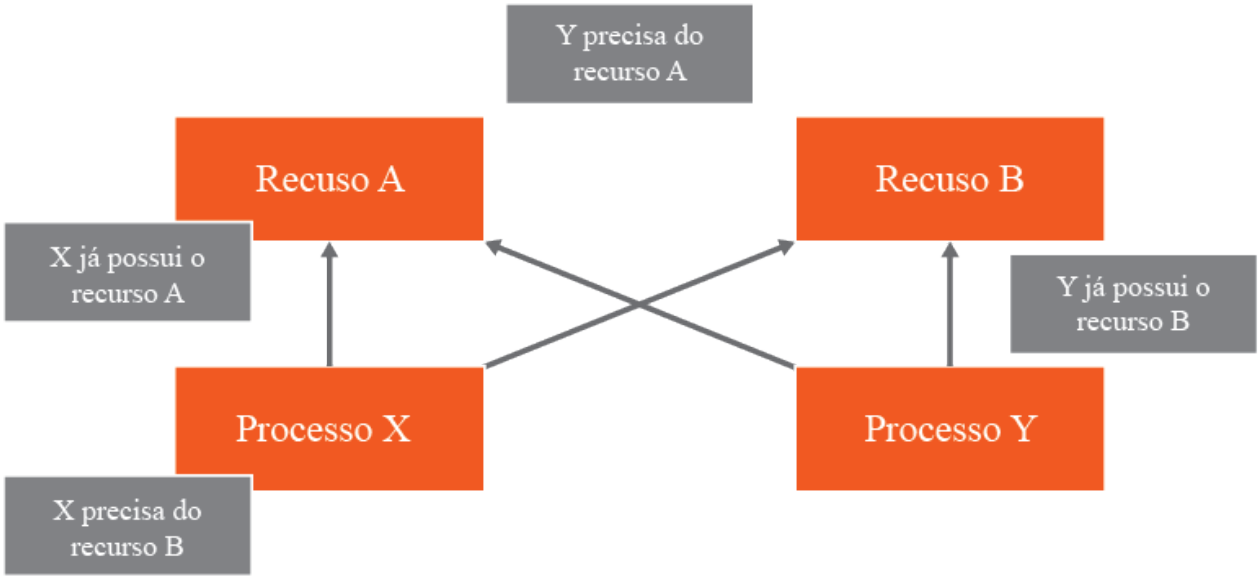

Figura 4 - Uma generalização da situação de deadlock em que há mútua dependência paralisante, na qual dois processos detêm e competem por recursos mútuos.
Fonte: Elaborado pelo autor, 2018.

Os projetistas dos sistemas operacionais têm algumas opções para tentar minimizar a ocorrência desse tipo de situação. A primeira é não fazer nada. Se isso acontecer, os processos terão que lidar com o fato, assim eles devem ter um tempo de espera, caso não tenham sucesso em bloquear todos os recursos, e liberar aqueles que já foram bloqueados e se encerraram, ou então recomeçarem novamente. Esta técnica é chamada de Técnica de Avestruz.

Outra opção consiste em forçar a preempção no uso dos recursos, dessa forma, arbitrariamente, o sistema operacional pode suspender o privilégio de acesso a um recurso por um determinado processo e entregar o controle para outro. Posteriormente, o controle voltará ao processo que inicialmente o estava utilizando. Esta

técnica é conhecida como Técnica de Preempção. Existem outras maneiras que envolvem a eliminação de um dos processos bloqueantes, ou que os processos informem antecipadamente quais recursos (e em que ordem) serão necessários para a realização de seus processamentos.
O sistema operacional Linux, por exemplo, não trata os problemas (isto é, não toma nenhuma providência e conhecimento) que ocorrem no espaço do usuário. Ele considera isso um problema a ser resolvido pelos processos. Já no espaço do núcleo, a ocorrência de *deadlocks* se dá eliminando as condições nas quais o problema possa ocorrer e se programando corretamente para que tais situações sejam anuladas.
Muito bem, continuando com nosso estudo, vamos para o próximo tópico: gerenciamento de arquivo.

4.3 Gerenciamento de arquivo

O gerenciamento de arquivo é uma responsabilidade do sistema operacional que está intimamente relacionada aos processos de entrada e saída (NEMETH; SNYDER; HEIN, 2007).
Um arquivo, como já sabemos, é uma abstração para uma região da memória secundária (externa em discos rígidos, por exemplo), nas quais os dados do usuário são guardados de maneira a não serem perdidos quando o equipamento for desligado. Essa discussão também é totalmente aplicada aos discos flexíveis, *pendrivers* e outros tipos de dispositivos para a guarda de dados.
Ocorre que os arquivos não podem simplesmente serem escritos no dispositivo, é preciso que haja um planejamento, um cuidado e uma estruturação de tal forma que seja possível a recuperação de qualquer arquivo em qualquer tempo pelo usuário e/ou pelos processos do usuário.
Para que essa organização seja possível, foram criados certos protocolos que governam a forma como os dados, os arquivos e as pastas de arquivos são criados, acessados e mantidos nos discos rígidos. Este protocolo recebeu o nome de **sistema de arquivo**.

Sistema operacional que o suporta	Sistema de arquivo	Principal uso
Windows NT e Windows XP	NTFS	Dados
Windows NT e Windows XP	FAT 32	Dados
Windows NT e Windows XP	FAT 16	Dados
LINUX	EXT2	Dados
LINUX	EXT3	Dados
LINUX	EXT4	Dados
LINUX	SWAP	Memória Virtual
LINUX	REISERFX	Dados
MacOS	HFS	Dados
MacOS	MFS	Dados
FreeBSD, OpenBSD, Solaris Sun	UFS	Dados
IBM Aix	JFS	Dados

Quadro 2 - Principiais tipos de sistemas de arquivos e os respectivos sistemas operacionais que os suportam.
Fonte: Elaborado pelo autor, 2018.

O que você precisa compreender é que todos sistemas de arquivos têm em comum as seguintes características: suportam a criação de pastas (diretórios); de arquivos; a exclusão de arquivos e pastas, bem como suas renomeações, completando as funcionalidades básica do sistema de arquivo, relacionadas às operações de leitura, escrita e execução de arquivo.

4.3.1 Tipos, sistemas e acesso

Alguns sistemas de arquivos suportam a recuperação de arquivos e/ou pastas apagadas (o que nem sempre é possível), outros não suportam (mesmo que parcialmente) esse tipo de operação e há aqueles que suportam a encriptação dos dados, de forma que apenas quem possuir chaves e senhas possa ter acesso aos dados.
Os sistemas de arquivos mais modernos suportam também a vinculação entre as pastas e os arquivos e seus proprietários, como o controle de tipo de acesso que pode ser realizado sobre tais arquivos, como de leitura, escrita etc. (NEMETH; SNYDER; HEIN, 2007).

VOCÊ QUER VER?

Como você deve ter percebido, a memória tem um papel essencial nos microcomputadores atuais. Em destaque, temos os discos rígidos que são amplamente utilizados e existem em diversas capacidades e fabricantes. Uma delas é a Seagate, companhia internacional com vários produtos nessa área. E você sabe como funciona um disco rígido? A empresa publicou um vídeo explicativo sobre o tema, disponível em: <https://youtu.be/lpYfep68xnA>. (SEAGATE TECNOLOGY, 2016).

Vimos há diferentes tipos de sistemas de arquivos, alguns são compatíveis com determinados SOs e outros não. Isso em alguns momentos torna-se um problema para o usuário, que pode não ter acesso aos seus dados devido a eles terem sido escritos utilizando outro tipo de sistema de arquivo.
Há casos em que o dispositivo usado no acesso aos dados permite apenas a escrita, ou então apenas a leitura. Também existem particularidades que estão associadas à mídia usada em tais equipamentos. Um desses casos ocorre quando em um dispositivo de leitor de CD seja usada uma mídia que permite apenas a leitura. Uma tentativa de escrita resultaria em um erro, pois não seria uma operação permitida para aquele dispositivo.

4.3.2 Estrutura de diretório

Uma diretório (ou pasta) é um conceito que permite agrupar um conjunto de arquivos (e outras pastas) sob um mesmo nome. Essa abstração é utilizada para permitir uma melhor organização nos dados.
Assim, o usuário pode criar estruturas que sejam convenientes para agrupar arquivos de uma mesma natureza, tais como documentos, fotos, músicas etc.
Na implementação de uma estrutura de diretório são necessárias algumas informações para a localização no disco. Em alguns casos, a depender do tipo de sistema de arquivo que esteja em uso, pode ser utilizada uma referência ao disco inteiro (para sistemas de alocação direta), ou um número que indique uma determinada posição do diretório no disco (número do primeiro bloco) para estruturas encadeadas, por exemplo.
Esses dados precisam ser armazenados em um ponto no disco de tal forma que exista uma convenção sobre como localizá-los (MACHADO; MAIA, 2013). Geralmente, utiliza-se uma parte do disco, previamente conveniada, a qual tem uma função de tabela de alocação. De fato, o conceito dessa tabela é amplamente conhecido como *File Allocation Table* (FAT), sigla que, inclusive, tem forte relação com os nomes dos sistemas de arquivos da Microsoft, mas como conceito está relacionada a todos os sistemas de arquivos. Essa tabela de alocação precisará ser preenchida com várias informações, tais como o tamanho do diretório (ou do arquivo), seu nome, dados do proprietário, permissões de acesso etc. Já que estamos tratando de criação e acesso a diretórios, arquivos e pastas, uma pergunta é válida: somente o usuário tem acesso a esses recursos? É um questionamento relacionado aos mecanismos de segurança, nosso tópico a seguir.

4.4 Mecanismos de segurança

As permissões de acesso e nomes de usuários dos diretórios e arquivos são exemplos de mecanismos de segurança que são implementados pelo sistema operacional. Utilizando desses dados simples, dono e permissões, já é possível prevenir uma série de problemas, como um usuário copiar ou ler dados de outros ou ainda modificar tais dados sem o consentimento do dono. Mas há casos em que você deseja que outros usuários

possam ler ou até mesmo alterar seus arquivos, para isso as permissões de acesso serão de grande ajuda. Todavia os problemas de segurança são mais complexos e, portanto, devem existir mais mecanismos de proteção.

VOCÊ SABIA?

Uma grande parte da segurança de dados que existe hoje é vinculada ao sistema operacional. Se ele falha, várias outras ações de segurança deixarão de ser efetivas. No entanto, é muito difícil para o sistema operacional proteger seus dados sem a ajuda do *hardware*. Uma das ações básicas para proteger os dados do usuário é garantir que um aplicativo (processo) não tenha acesso irrestrito a toda memória real do microcomputador. Isso é necessário para impedir que um processo modifique dados de outro processo. No passado, as CPUs não tinham um chamado "modo protegido", agora, exceto alguns microprocessadores e microcontroladores, esse modo protegido já existe.

A memória do microcomputador, bem como os dispositivos que permitem que ele se comunique com outros equipamentos, também deve ser alvo de preocupações relativas à segurança. Muitas são as técnicas utilizadas para explorar brechas no gerenciamento de segurança, ou até mesmo funcionalidades essenciais de alguns tipos de aplicativos. A exemplo, podemos citar um aplicativo para a publicação de páginas na internet, denominado servidor *web*. Um servidor *web* necessita, essencialmente, comunicar-se através da internet e, para tanto, precisa de acesso à rede de dados do microcomputador, que requisitará esse tipo de recurso ao sistema operacional.

Ao permitir que o servidor *web* faça uso de comunicação com a internet, também passa a ser possível que usuários externos (desconhecidos ao sistema operacional) possam solicitar recursos. Esses recursos, em tese, deveriam se referir às páginas html (dentre outras) que seriam disponibilizadas pelo servidor *web*, todavia, caso existam falhas no *design* desse aplicativo servidor, ele pode ser utilizado como recurso para que sejam obtidos documentos que não deveriam ser acessados por terceiros, como, por exemplo, a relação de usuários registrados no sistema operacional.

CASO

A segurança de dados nasce no sistema operacional e se estende até o comportamento do usuário. Um caso que recebeu muita atenção da mídia foi o do *Wikileaks*, cujo fundador, Julian Assange, está no centro da divulgação de dados, por exemplo, do partido Republicano dos Estados Unidos. Atualmente, Assange está refugiado em uma embaixada sem tratado de extradição. O caso é notório pela forma como os dados iniciais (cerca de 92 mil documentos sobre operações militares de diversos países) foram obtidos. Há hipótese de que foram a partir de um sistema militar de correio eletrônico usado pelo exército. Para descobrir quem esteve envolvido no processo de cópia não autorizada dos documentos foram utilizados os registros de históricos (*logs*) do sistema operacional, que permitiu visualizar os usuários que fizeram uso de determinados equipamentos no período em que ocorreu o vazamento. Isso reforça que é sempre importante manter o controle de acesso, com o uso exclusivo de credenciais, nunca cedendo a terceiros, como também de uma política de guarda e verificação dos históricos do sistema operacional.

Percebe-se que o sistema de arquivo é um ponto de vulnerabilidade em qualquer sistema operacional, pois mesmo que existam várias ações para tornar o SO seguro, se as credenciais dos usuários forem roubadas, de nada valerá qualquer ação tomada anteriormente. Uma maneira de tornar um sistema operacional mais seguro é melhorar o seu ambiente de funcionamento.

4.4.1 Ambiente, controle de acesso aos recursos

Suponha que você coloque o seu sistema operacional para funcionar como um grande processo dentro de outro sistema operacional. Isso seria possível? Tornaria seu sistema um pouco mais seguro?

A resposta para as duas questões é: sim. Este conceito, na verdade, é conhecido como **virtualização** e pode ser utilizado para dar mais segurança a um sistema operacional, ou ainda para que um microcomputador potente possa parecer ser mais de um.

Pela virtualização é possível que em um mesmo microcomputador você tenha executado vários sistemas operacionais ao mesmo tempo. Isso é uma vantagem quando, por exemplo, você precisa ter duas máquinas com sistemas operacionais diferentes para realizar tarefas que dependem de aplicativos que funcionam somente em determinados sistemas.

Mas, retornando ao foco da segurança, através da virtualização, a máquina real, hospedeira, torna-se um pouco mais afastada dos usuários, podendo minimizar os riscos de perda de dados e criando máquinas virtuais para demandar necessidades específicas. Suponha que para o servidor *web*, nos criássemos uma máquina virtual e lá incluíssemos os documentos html que desejamos publicar. Caso essa máquina fosse alvo de alguma violação de segurança, no máximo, poderiam obter os documentos html (que já estão expostos) e a máquina real ainda estaria protegida.

Com a virtualização, pode-se controlar o acesso a certos recursos quando forem ofertados através de redes de dados. Mesmo que não se utilize o conceito de virtualização, é possível também tornar o ambiente do sistema operacional mais seguro. Para isso, os dados que se deseja proteger podem ser criptografados utilizando-se esquemas bem complexos. Nesses processos, os dados criptografados somente poderão ser lidos se o usuário possuir uma senha especial de acesso.

4.4.2 Noção básica de criptografia

A criptografia consiste em transformar os dados em outros, de forma que essa transformação seja controlada e reversível. Os dados transformados não farão sentido e nem poderão ser associados a algum processo específico. A reversão para os originais somente pode ocorrer quando o usuário possuir o algoritmo e uma chave de transformação. Daqui em diante, vamos usar o termo criptografia para nos referirmos à essa transformação.
A criptografia pode ser, em geral, de dois tipos: simétrica e assimétrica. A simétrica é aquela na qual existe apenas uma única chave de decriptação, sem a qual os dados estarão perdidos. Já a criptografia por chave assimétrica requer duas senhas, uma que é pública e todos podem conhecer, e outra que é secreta.

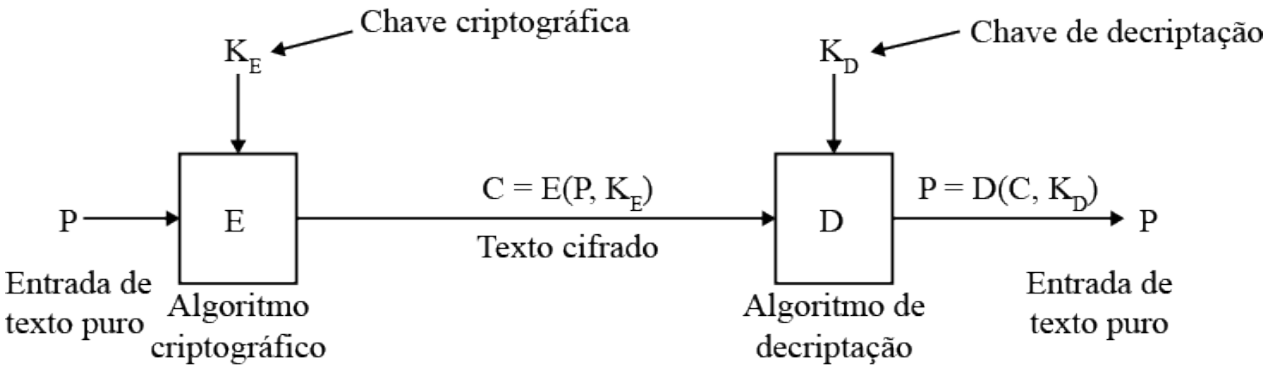

Figura 5 - O esquema básico mostra que a criptografia é dependente unicamente da chave, associada ao algoritmo.
Fonte: TANENBAUM; BOS, 2016, p. 429.

Na Figura anterior, podemos ver a forma como é feita a criptografia simétrica, em que há a possibilidade de uso de dois algoritmos diferentes, um para a criptografia e outro para a decriptação.

VOCÊ SABIA?

A criptografia já é usada pela humanidade há muito tempo. Durante o governo de Cesar, na Roma antiga, houve um esquema de criptografia que ficou conhecido como Cifra de Cesar, uma técnica simples, para os dias de hoje, mas que teve enorme importância no uso estratégico da comunicação militar naquele período. A técnica consistia em fazer uma troca de uma letra do alfabeto por outra, mas de maneira planejada, de forma que, posteriormente, fazendo-se a troca inversa, era possível recuperar a mensagem original.

Na criptografia por chave assimétrica os processos são bem mais complexos e demandam muito mais tempo, todavia o resultado é mais seguro. É o tipo mais utilizado na transferência de dados entre sistemas operacionais e em aplicações sensíveis, como as de acesso aos bancos.

Síntese

Neste quarto e último capitulo, pudermos conhecer um pouco mais sobre como o sistema operacional permite e controla o acesso aos vários recursos administrados por ele. A respeito desse controle, percebemos que a segurança dos dados requer várias ações do sistema operacional e do próprio usuário. Destacamos também a importância e as diferenças dos diversos tipos de memória que compõem o microcomputador, analisamos os conceitos de paginação e cobrimos os problemas relacionados aos conflitos no uso de recursos, em especial, os *deadlocks*.

Neste capítulo, você teve a oportunidade de:

- compreender a importância da proteção dos dados do usuário, mantidos em unidades de memória secundária, como, por exemplo, os discos rígidos;
- entender a relevância da arquitetura dos microcomputadores atuais que são dotados de memória cache e o impacto do uso dessa funcionalidade no desempenho do sistema;
- compreender a complexidade das abstrações usadas (arquivos e diretórios) para que os dados guardados nos discos rígidos possam ser facilmente recuperados;
- reconhecer a importância da segurança de dados como um fator para a perenidade dos dados do usuário e seu perfeito funcionamento.

Bibliografia

BARCELLOS, M. P. Programação Paralela e Distribuída em Java. In: 2ª Escola Regional de Alto Desempenho - ERAD 2002. São Leopoldo. **Anais...** São Leopoldo, p. 181-192, 2002. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/erad-rs/2002/007.pdf>. Acesso em: 31/05/2018.

BARTLETT, J. **Programming from the Ground UP**. Edição: Dominick Bruno, Jr. 2003. Disponível em: < http://nongnu.askapache.com/pgubook/ProgrammingGroundUp-0-9.pdf>. Acesso em: 11/07/2018.

CHRISTIAN, B.; GRIFITHS, T. **Algoritmos para viver:** a ciência exata das decisões humanas. São Paulo: Companhia das Letras, 2017.

DEITEL, H. **Sistemas Operacionais.** 3. ed. São. Paulo: Pearson, 2005.

_____; DEITEL, P. J. **C: Como programar**. 6. ed. São Paulo: Pearson Prentice Hall, 2011.

ECKERT, J. P. ; MAUCHLY, J. W. Automatic high-speed computing: a progress report on the EDVAC. 1945, 22 p. In: RANDALL, N. **Knuth Digital Archive Project.** History Box 2, abr 2006. Disponível em: <http://archive.computerhistory.org/resources/text/Knuth_Don_X4100/PDF_index/k-8-pdf/k-8-u2736-Report-EDVAC.pdf>. Acesso em: 11/07/2018.

FOROUZAN, B. A. **Comunicação de Dados e Rede de Computadores.** 4. ed. Porto Alegre: McGrawHill, 2010.

INTEL. **Processo de Fabricação do processador Intel Core i5**. Direção: INTEL. Produção: INTEL. Legendas: Kaike Iaralian. 2014. Disponível em: <https://youtu.be/2W6kCXwSZnk>. Acesso em: 11/07/2018.

MACHADO, F. B.; MAIA, L. P. **Arquitetura de Sistemas Operacionais**. 5. ed. Rio de Janeiro: LTC. 2013. Disponível na Biblioteca Virtual Laureate: <https://laureatebrasil.blackboard.com/webapps/blackboard/content/listContent.jsp?course_id=_198689_1&content_id=_4122211_1&mode=reset>. Acesso em: 24/05/2018.

MELO NETO, A. **Estrutura dos Sistemas Operacionais.** USP. 2014. Disponível em: <https://www.ime.usp.br/~adao/teso.pdf>. Acesso em: 17/05/2018.

NEMETH, E.; SNYDER. G.; HEIN, T. R. **Manual Completo do Linux**: Guia do Administrador. 2. ed. São Paulo: Pearson, 2007. Disponível na Biblioteca Virtual Laureate: <https://laureatebrasil.blackboard.com/webapps

/blackboard/content/listContent.jsp?course_id=_198689_1&content_id=_4122211_1&mode=reset>. Acesso em: 24/05/2018.
SEAGATE TECNOLOGY. **How a hard disk drive works**. Direção: Edson Tech Center. Produção: Edison Tech Center. EUA, 2016. Disponível em: <https://youtu.be/NtPc0jI21i0>. Acesso em: 11/07/2018.
SILBERSCHATZ, A.; GALVIN, P. B. **Fundamentos de Sistemas Operacionais**. 9. ed. São Paulo, LTC 2015. Disponível na Biblioteca Virtual Laureate:<https://laureatebrasil.blackboard.com/webapps/blackboard/content/listContent.jsp?course_id=_198689_1&content_id=_4122211_1&mode=reset>. Acesso em: 24/05/2018.
KOWALTOWSKI, T. Von Neumann: suas contribuições à Computação. **Estud. av.**, São Paulo, v. 10, n. 26, p. 237-260, abr 1996. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-40141996000100022>. Acesso em: 11/07/2018.