Análise do Sistema Operacional Windows 10

Jorge Luiz Andrade, Ana Carolina Lopes, and Matheus Castro

***Abstract*—aa aaa[1]**

I. Introdução

II. Windows 10

A. Arquitetura

B. Gerência de memória

O Windows 10 estabelece 4GB como limite para memória física em versões 32-bits e 2TB em versões 64-bits, com exceção da versão Home, que possui limite de 128GB em sua versão de 64-bits [2].

A memória física pode ser dividida em [2]:

- **Reservada para o Hardware**: armazena drivers de hardware que devem sempre permanecer na memória física, não estando disponível para uso do gerenciador de memória.
- **Em uso:** É a memória em uso por todos os processos em execução, *kernel* do SO e *drivers*.
- **Modificada:** É a memória de páginas que foram modificadas em processos que ficaram em espera. Os dados anteriores são escritos em disco, mas facilmente recuperados.
- **Em espera:** É a memória que estava alocada em processos que terminaram normalmente. O gerenciador de memória mantém os dados em memória como uma espécie de cache para arquivos usados recentemente. A memória em espera está disponível para alocação, mas suas páginas são classificadas de 0 a 7, sendo as páginas com menores valores usadas primeiro.
- **Livre:** É a memória que ainda não foi alocada ou que retornou para o gerenciador de memória por um processo que foi terminado.

O gerenciador de memória do Windows 10 faz parte do Windows executive, uma porção em baixo nível do seu *kernel*, residindo no arquivo *Ntoskrnl.exe*. É responsável, entre outras funções, por [3]:

- Alocar, desalocar e gerenciar a memória virtual, que em sua maior parte está exposta por meio da API do Windows ou de interfaces para drivers de dispositivos em modo kernel;
- Garantir que processos não acessem regiões a que não possuem permissão;

O ambiente Windows, de modo geral, utiliza o conceito de espaço de endereçamento virtual para um processo, sendo este o conjunto de endereços da memória virtual que esse processo tem acesso. O espaço de endereçamento é privado e não pode ser acessado por outros processos que não o compartilhem [3].

O espaço de endereçamento em versões 32-bits do Windows é de até 4GB, dividido em uma partição para o processo e outra para uso do sistema. Versões 64-bits do sistema suportam endereçamento em modo usuário de até 8TB [2].

Assim como todos os componentes do *Windows executive*, o gerenciador de memória é totalmente reentrante, ou seja, pode executado novamente antes que a execução anterior tenha sido concluída, e suporta execução simultânea em sistemas multiprocessados. Isso permite que duas ou mais *threads* adquiram recursos de forma que seus dados não sejam corrompidos [3].

Uma funcionalidade importante introduzida no Windows 10 é a compressão de memória. Normalmente, quando uma página está inativa por um longo tempo o gerenciador de memória a move para a área de memória modificada. Se essa página continuar a não ser referenciada mas o sistema necessitar de memória adicional para outros processos a página é então escrita em disco em um arquivo denominado *pagefile* [2].

O Windows 10 introduz o conceito de que, antes da página ser enviada ao disco, o gerenciador de memória irá comprimir páginas não utilizadas. A compressão de memória chega a ocupar apenas 40% do tamanho original, reduzindo a necessidade de acesso ao disco a 50% do que era necessário em versões anteriores do Windows [4].

C. Gerência de processos

No Windows, processos são detentores de recursos que armazenam informações sobre o espaço de endereçamento virtual, os manipuladores que referenciam os objetos no modo kernel e as threads. As threads, por sua vez, são abstrações do kernel para realizar o escalonamento da CPU. Cada uma possui uma prioridade baseada no processo ao qual elas fazem parte e também podem ter afinidades com certos processadores. Cada thread possui duas pilhas, uma para que ela seja executada em modo kernel e outra para execução em modo usuário [5].

Comunicação entre Processos: Se dá pela comunicação entre threads, que pode ser feita por meio de pipes, pipes nomeados, mailslots, sockets, chamadas de procedimento remotas e arquivos compartilhados [5].

Um pipe é um tipo de pseudo-arquivo que pode ser usado para conectar dois processos. Mailslots e sockets são similares aos pipes. No entanto, diferente dos pipes, a comunicação não é bidirecional em mailslots e os sockets costumam conectar processos em máquinas diferentes.

Sincronização: O Windows fornece vários mecanismos de sincronização, incluindo semáforos, mutexes, eventos e regiões críticas [5].

Escalonamento: O sistema de escalonamento do Windows é completamente preemptivo e guiado por prioridades. O kernel do Windows não possui uma thread de escalonamento central. Por isso, quando uma thread não pode mais executar, ela entra em modo kernel e executa o escalonador. Para que uma thread execute o escalonador, uma das seguintes situações deve ocorrer [5]:

- A thread atual bloqueia em um semáforo, mutex ou evento de E/S;
- A thread sinaliza um objeto;
- O seu quantum acaba.

O escalonador também pode ser chamado quando uma operação de E/S ou uma espera temporizada termina.

Para implementar o escalonamento, o sistema mantém uma lista com 32 entradas, tal que cada uma contém um conjunto de threads com prioridade correspondente ao número da entrada. O algoritmo de escalonamento busca o vetor desde a prioridade 31 até 0 por uma entrada não vazia. Quando encontra, seleciona a primeira thread e esta é executada durante um quantum.

A API do Windows organiza os processos de acordo com a classe de prioridade que eles recebem quando são criados. Essas classes são: tempo real, alta, acima do normal, normal, abaixo do normal ou ocioso. Feito isso, ela atribui uma prioridade relativa às threads individuais dentro de cada processo. Essas prioridades são: tempo crítico, mais alta, acima do normal, mais baixa e ociosa. No kernel, a classe de prioridade é convertida para uma prioridade-base e nela aplica-se um diferencial de acordo com a prioridade relativa da thread. Desta forma, processos possuem apenas uma prioridade-base, enquanto as threads possuem duas prioridades: atual e base.

As threads de tempo real nunca têm suas prioridades alteradas. As demais threads, no entanto, podem ter suas prioridades atuais alteradas. As seguintes situações podem acarretar na alteração da prioridade atual de uma thread:

- Uma operação de E/S é finalizada;
- Uma thread que esteja esperando um semáforo, mutex ou outro evento seja liberada;
- Uma thread de GUI desperta.

Se a thread gastar todo o seu quantum durante a execução, sua prioridade é rebaixada. Isso ocorre quantas vezes ela for executada, até que a sua prioridade iguale-se à prioridade do nível-base.

D. Gerência de arquivos

Assim como ocorre desde a versão 3.1, o *Windows* 10 utiliza o *NTFS*(*New Technology File System*) como seu sistema de arquivos padrão em ambientes domésticos, suportando volumes e arquivos de até 256TB quando utilizado o tamanho do *cluster* padrão de 64KB e até 2^{32}-1 arquivos por volume e pasta [6].

O NTFS foi desenvolvido de forma a incluir funcionalidades necessárias em sistemas de arquivos empresarias. Isso inclui integridade e recuperação de dados, proteção à informações sensíveis, redundância de dados e tolerância a falhas [3].

- **Integridade e recuperação de dados**: Modificações no sistema de arquivos são realizadas em operações atômicas, ou seja, toda a operação deve ser completada ou nenhuma parte dela o será.
- **Segurança**: Arquivos e diretórios são associados à um arquivo oculto de segurança contendo as informações de permissão. Assim que um processo tenta utilizar um arquivo, suas permissões são checadas, e seu acesso só é permitido se autorizado pelo administrador do sistema ou pelo dono do arquivo.
- **Redundância e tolerância a falhas**: O NTFS garante que o sistema de arquivos permaneça acessível após uma falha do disco, mas não garante integridade dos arquivos em si. Essa integridade, entretanto, é alcançada utilizando-se RAID 1 e 5.

O sistema NTFS não tenta evitar fragmentação de arquivos durante suas alocações. Entretanto, além de sua própria ferramenta, o Windows inclui uma API que permite o desenvolvimento de ferramentas de desfragmentação de terceiros, que permite que dados de arquivos sejam movidos de forma que ocupem *clusters* contíguos, possuindo como única limitação o impedimento da desfragmentação em arquivos de paginação e de logs do sistema NTFS [3].

E. Gerência de E/S

O gerenciador de Entrada/Saída do *kernel* do Windows 10 realiza a comunicação entre o sistema operacional e os *drivers* de dispositivos por meio de *IRPs*(*I/O request packets*, ou pacotes de requisição de E/S). Isso permite que *threads* individuais operem em múltiplas chamadas de E/S de forma concorrente [3].

Devido aos dispositivos operarem em velocidades diferentes daquela do sistema operacional, a comunicação IRP se assemelha a pacotes de redes, sendo passados do sistema operacional para um *driver* de dispositivo, e de um *driver* para outro, por meio do gerenciador de E/S. O sistema de E/S do Windows possui um modelo em camadas, ou pilha, onde cada controlador na pilha envia e recebe IRPs [7].

Além da criação e distribuição de IRPs, o gerenciador também possui código comum a diferentes controladores, facilitando a criação e utilização de *drivers* individuais de dispositivos.

F. Interrupções

G. Kernel

H. Suporte a threads

I. Segurança

A primeira camada de proteção do Windows 10 é o próprio hardware. Alguns dos recursos de segurança do Windows 10 tiram proveito de projetos modernos de hardware. Destacam-se [8]:

- **Unified Extensible Firmware Interface (UEFI)**: Interface de firmware que realiza funções anteriormente realizadas pela BIOS. É um aspecto chave da segurança do Windows 10, oferecendo Boot Seguro e suporte a dispositivos auto-encriptados. Quando o Boot Seguro está ativado, o usuário só é capaz de iniciar o computador usando um loader que use um certificado armazenado no UEFI.
- **Trusted Platform Module (TPM)**: Chip que dá suporte a encriptação de alto nível e previne adulteração de certificados e chaves de encriptação. A presença do TPM permite recursos como a encriptação de drive BitLocker, Measured Boot e Device Guard.

Além disso, o Windows 10 oferece suporte a dispositivos de hardware que permitem que usuários se identifiquem por meio de informações biométricas, como digitais e reconhecimento facial.

A encriptação é habilitada por padrão em todas as edições do Windows para dispositivos que incluem um TPM. As edições Pro e Enterprise podem ser configuradas com proteção BitLocker (que é um sistema de criptografia que codifica partições) adicional e capacidade de gerência. A encriptação é habilitada assim que um administrador local faz login com uma conta da Microsoft. A chave de recuperação é automaticamente armazenada no OneDrive do usuário para que ele seja capaz de recuperar os dados encriptados posteriormente [8].

Além disso, o Windows 10 implementa também o Windows Passport, que substitui senhas por uma autenticação dupla. Para isso, o usuário deve registrar um dispositivo por meio de uma conta de um dos serviços da Microsoft ou algum outro serviço que dê suporte para a autenticação FIDO (Fast IDentity Online). Após o registro, o próprio dispositivo se torna um dos fatores para realizar a autenticação. O segundo fator é o PIN. Desta forma, mesmo que um usuário com intenções maliciosas tenha caches de nomes de usuários e senhas, ele não será capaz de se autenticar sem o dispositivo físico e a habilidade de transmitir a credencial do usuário e o seu PIN ou informação biométrica. Para utilizar esse recurso, o dispositivo precisa ter um TPM, que armazena o certificado do dispositivo quando ele é registrado.

Além disso, a segurança do Windows se dá também por listas de controle de acesso e níveis de integridade. Cada processo tem um token de autenticação que especifica a identidade do usuário e quais são os seus privilégios e cada objeto tem um descritor de segurança associado que aponta para a lista de controle de acesso que dá ou nega acesso a grupos ou determinados indivíduos [5].

III. Conclusões

References

[1] Mark Russinovich, David Solomon, and Alex Ionescu. *Windows Internals Part 2*. Microsoft Press, 6 edition, 2012.

[2] Sushovon Sinha. Physical and virtual memory in windows 10. http://answers.microsoft.com/en-us/windows/forum/windows_10-performance/physical-and-virtual-memory-in-windows-10/e36fb5bc-9ac8-49af-951c-e7d39b979938. Acessado em 22 de novembro de 2016.

[3] Mark Russinovich, David Solomon, and Alex Ionescu. *Windows Internals Part 2*. Microsoft Press, 6 edition, 2012.

[4] Ethan Creeger. Windows 10: Memory compression.

[5] Andrew S. Tanenbaum and Herbert Bos. *Modern Operating Systems*. Prentice Hall, 4 edition, 2014.

[6] Default cluster size for ntfs, fat, and exfat. https://support.microsoft.com/en-us/kb/140365. Acessado em 26 de novembro de 2016.

[7] Windows kernel-mode i/o manager. https://msdn.microsoft.com/en-us/library/windows/hardware/ff565734. Acessado em 26 de novembro de 2016.

[8] Ed Bott. *Introducing Windows 10 for IT Professionals: Technical Overview*. Microsoft Press, 2 edition, 2016.