Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

JOÃO VITOR CAVALARI SPAVIER

AVALIAÇÃO DE SEGURANÇA DOS DADOS EM UM AMBIENTE DE SWAPPING

Assis/SP
2023

Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

JOÃO VITOR CAVALARI SPAVIER

AVALIAÇÃO DE SEGURANÇA DOS DADOS EM UM AMBIENTE DE SWAPPING

Trabalho de conclusão de curso apresentado ao curso de Ciência da Computação do Instituto Municipal de Ensino Superior de Assis – IMESA e a Fundação Educacional do Município de Assis – FEMA, como requisito parcial à obtenção do Certificado de Conclusão.

Orientando(a): João Vitor Cavalari Spavier
Orientador(a): Me. Fabio Eder Cardoso

Assis/SP
2023

FICHA CATALOGRÁFICA

SPAVIER, João Vitor Cavalari.
Avaliação de segurança dos dados em um ambiente de Swapping / João Vitor Cavalari Spavier. Fundação Educacional do Município de Assis –FEMA – Assis, 2023.
Número de páginas.

1. Swapping. 2. Segurança.

CDD:
Biblioteca da FEMA

JOÃO VITOR CAVALARI SPAVIER

AVALIAÇÃO DE SEGURANÇA DOS DADOS EM UM AMBIENTE DE SWAPPING

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Graduação, avaliado pela seguinte comissão examinadora:

Orientador: Me. Fabio Eder Cardoso

Examinador: Dr. Almir Rogerio Camolesi

Assis/SP
2023

DEDICATÓRIA

Dedico este capítulo da minha vida, dedicado ao estudo da Ciência da Computação, com profunda gratidão a todos que estiveram ao meu lado. Em especial, quero expressar minha imensa gratidão à minha família – meus pais e irmão –, cujo apoio inabalável e incentivo constante me guiaram por esta jornada desafiadora.

Suas palavras de encorajamento e confiança me deram força para superar obstáculos e persistir em busca do conhecimento. A dedicação que demonstraram ao meu crescimento é um tesouro que guardo com carinho.

Agradeço por acreditarem em mim e por serem a base sólida em que construí minha jornada acadêmica e profissional. Este momento de conclusão é também um tributo à dedicação e amor que me ofereceram.

Com gratidão eterna,

João Vitor Cavalari Spavier

AGRADECIMENTOS

Gostaria de expressar minha profunda gratidão por esses quatro anos de estudo em Ciência da Computação. Essa jornada não apenas enriqueceu meu entendimento da área, mas também me moldou como pessoa e profissional.

Quero agradecer aos colegas de classe por compartilharem essa trajetória comigo, aprendendo e crescendo juntos. Agradeço aos professores pelo conhecimento transmitido e especialmente ao meu orientador, Fabio, cuja orientação foi essencial para este trabalho.

Essa jornada na Ciência da Computação ampliou minha visão e me inspirou a continuar explorando novos desafios. A todos que contribuíram para minha jornada, meu sincero obrigado. Estou animado com o que o futuro reserva.

RESUMO

Em ambientes computacionais os processos que ocorrem na máquina ficam armazenados na memória em execução, também conhecidas como memória RAM (*Random Access Memory*) porém, em muitos casos, existe a falta de espaço para esses processos, surgindo então a necessidade da utilização da técnica de swapping. O presente trabalho apresenta técnicas para avaliar a segurança dos dados gerados e armazenados na memória secundária.Descrever aqui a importância do projeto e qual metodologia que será utilizada no trabalho e os resultados esperados.

Palavras-chave: *Swapping*, *Swap*, memória virtual, memória secundaria, memória principal, processo.

ABSTRACT

In computational environments, the processes that occur on the machine are stored in the memory while running. However, in many cases, there is a lack of space for these processes, which creates the need for the use of the swapping technique. With the use of this method, it is necessary to deepen the security evaluation of the data generated and stored in secondary memory.

Keywords: ***Swapping, Swap, virtual memory, secondary memory, main memory, process.***

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

O gerenciamento eficiente da memória é essencial para um bom desempenho do sistema operacional. Mesmo com a evolução da tecnologia, a capacidade dos programas tem aumentado mais rapidamente do que as memórias, o que faz com que seja necessário um gerenciamento cuidadoso da memória, segundo Monteiro (2006, p. 168), para que um programa seja alocado para execução ele necessita que haja espaço na memória principal, porém, em muitos casos, existe a falta de espaço na memória durante a criação do processo. Durante toda a evolução da tecnologia computacional, diversos métodos foram criados e testados para solucionar este problema, tais como, hierarquia de memórias, espaço de endereçamento, realocação dinâmica, entre outras.

Dentre esses métodos surgiu aproximadamente em 1960, a técnica de ***swapping***, conhecida também como "*paging*" ou "memória virtual". O *swapping* é uma técnica de gestão de memória computacional que permite a troca de processos dentro do sistema operacional, ela garante que o sistema operacional possa alterar o estado de funcionamento do processo, além de realizar a troca dele, sendo assim, esta técnica é aplicada dentro do sistema operacional no momento em que a memória principal se encontra sem espaços para mais processos. Quando a memória está sem espaço de armazenamento para mais processos dentro do sistema, a memória virtual é acionada realizando a troca de processos, ou seja, quando um processo que está em estado de espera dentro do "**SO**" é reconhecido e simultaneamente outro estado novo necessita ir para o estado de execução, o sistema operacional aloca um espaço na memória secundaria chamado de espaço de swapping e nele será alocado o processo que estava em estado de espera, ele ficará alocado ali até que feita uma chamada neste processo e assim consecutivamente é realizado para os demais processos existentes e os que irão surgindo conforme a necessidade.

Durante o processo de *swapping,* como mencionado anteriormente, os dados são salvos na memória secundaria em um espaço reservado, em sistemas operacionais como por exemplo o Linux, durante a instalação a partição de *swap* é criada automaticamente ou manualmente pelo usuário durante a instalação, esta partição retem parte da memória secundaria como espaço para atuação da memória virtual. Este espaço alocado no disco secundário pode ser acessado pelo usuário

durante o uso do sistema operacional, sendo assim um risco para segurança dos dados presentes nos processos que lá estão armazenados.

1.1. OBJETIVO

O objetivo primordial deste trabalho consiste em realizar uma avaliação abrangente do nível de segurança dos dados gerados na memória secundária, por meio da utilização da técnica de swapping, em ambientes computacionais. Nesse contexto, busca-se compreender em profundidade a eficácia das medidas de segurança implementadas, bem como identificar possíveis lacunas e vulnerabilidades existentes.

Além disso, o estudo visa medir a capacidade dos sistemas operacionais em lidar com a segurança e a integridade desses dados críticos armazenados na memória secundária. É fundamental analisar como os sistemas operacionais gerenciam o processo de swapping, considerando as implicações para a segurança e a confidencialidade das informações.

Um aspecto relevante a ser investigado é o acesso aos dados em memória secundária pelo usuário do computador. Serão exploradas as permissões e os controles de acesso implementados pelos sistemas operacionais, com o intuito de verificar se essas medidas são efetivas na proteção dos dados contra acesso não autorizado.

Além disso, serão conduzidos testes e experimentos com diferentes cenários, a fim de identificar possíveis falhas nos sistemas operacionais relacionadas ao gerenciamento do processo de swapping. Essa análise minuciosa permitirá uma compreensão mais abrangente dos riscos e desafios associados à segurança dos dados em memória secundária.

Como resultado deste estudo, espera-se fornecer insights valiosos para aprimorar a segurança dos dados em memória secundária, contribuindo para o desenvolvimento de práticas e diretrizes mais robustas no âmbito dos sistemas operacionais. Ademais, espera-se gerar recomendações específicas para os desenvolvedores de software e os responsáveis pela gestão de sistemas, visando fortalecer as medidas de segurança e garantir a integridade e a confidencialidade dos dados sensíveis armazenados na memória secundária.

1.2. JUSTIFICATIVA

O presente trabalho se justifica pela importância dos dados no ambiente computacional. Os dados são uma sequência de códigos binários que formam informações de valor e relevância para os usuários (Machado & Maia, 2013). Independentemente do contexto ou utilização, a segurança desses dados é essencial.

No contexto computacional, os dados têm um valor significativo tanto para as corporações quanto para os usuários. Muitos desses dados carregam informações críticas e confidenciais. Portanto, a falta de proteção adequada desses dados, especialmente em um ambiente de swapping, pode resultar em sérios transtornos para empresas, usuários comuns e órgãos governamentais.

A técnica de swapping é utilizada pelos sistemas operacionais para gravar temporariamente dados na memória secundária. No entanto, essa abordagem implica no potencial acesso dos usuários a esses dados sensíveis armazenados nessa área. Nesse sentido, é fundamental avaliar a segurança desses dados quando passam por esse processo de troca de memória.

A segurança dos dados em um ambiente de swapping é uma preocupação central deste estudo. É necessário investigar como os sistemas operacionais garantem a integridade e a confidencialidade dos dados durante o processo de troca, considerando as possíveis ameaças e os riscos de acesso não autorizado a essas informações.

Ao abordar essa situação, é crucial que o sistema operacional escolhido pelo usuário seja capaz de proteger efetivamente os dados. A integridade dos dados é um fator fundamental para a confiança dos usuários e a continuidade das operações nas organizações.

Portanto, a justificativa para este estudo baseia-se na necessidade de garantir a segurança dos dados, reconhecendo seu valor para o contexto computacional. A análise da segurança desses dados em relação ao processo de swapping, bem como a identificação de possíveis falhas nos sistemas operacionais, contribuirá para fortalecer as medidas de proteção e minimizar os riscos associados à exposição dos dados sensíveis.

1.3. MOTIVAÇÃO

O presente trabalho de conclusão de curso é motivado pela necessidade de avaliar se os sistemas operacionais fornecem a proteção adequada dos dados em um ambiente onde estes estão passando pelo processo de swapping. É de interesse de todos os usuários que seus dados estejam protegidos, preservados e com sua integridade assegurada.

Os dados gerados na memória secundária podem ser visualizados por usuários comuns, o que os expõe tornando-os acessíveis. Essa acessibilidade dos dados em memória secundária pode criar vulnerabilidades significativas, deixando em questão a segurança do sistema operacional.

A motivação para este estudo reside na necessidade de investigar se os sistemas operacionais garantem a confidencialidade e a integridade dos dados durante o processo de swapping. A exposição desses dados sensíveis representa um risco para a privacidade e a segurança das informações.

É fundamental compreender como os sistemas operacionais lidam com o gerenciamento e a proteção dos dados em memória secundária. Essa análise permitirá identificar possíveis deficiências nos sistemas operacionais que podem comprometer a segurança dos dados durante o processo de swapping.

Além disso, a motivação para este estudo também se baseia na importância de estabelecer diretrizes e práticas mais robustas para garantir a proteção dos dados em memória secundária. Ao obter uma compreensão mais aprofundada dos riscos e desafios associados a esse ambiente, será possível desenvolver recomendações específicas para melhorar a segurança dos dados e fortalecer a confiabilidade dos sistemas operacionais.

Em resumo, a motivação deste trabalho reside na necessidade de avaliar a eficácia dos sistemas operacionais em proteger os dados durante o processo de swapping, garantindo sua confidencialidade, integridade e segurança. Ao identificar possíveis lacunas nos sistemas operacionais, busca-se contribuir para o desenvolvimento de medidas mais efetivas e robustas de proteção de dados em memória secundária.

1.4. PERSPECTIVAS DE CONTRIBUIÇÃO

A avaliação de segurança dos dados visa demonstrar aos usuários as possíveis vulnerabilidades do sistema operacional em relação a técnica de *swapping*, trazendo para o usuário a visibilidade do que ocorre dentro do próprio sistema operacional, mostrando e inspirando trabalhos de correção ou aprofundamento na avaliação da segurança neste contexto, apresentando uma visão melhorada deste assunto que tem uma grande importância para o cenário computacional.

1.5. METODOLOGIA DE PESQUISA EXPERIMENTAL

A metodologia de pesquisa adotada neste trabalho envolveu diferentes etapas para atingir os objetivos propostos. Inicialmente, foi realizada uma revisão bibliográfica abrangente, abarcando livros e artigos científicos relacionados à arquitetura de computadores e à arquitetura de sistemas operacionais. Essa revisão permitirá um aprofundamento no conhecimento sobre o funcionamento da máquina e o comportamento do sistema operacional instalado nela.

Em seguida, foi realizado um estudo detalhado dos sistemas operacionais Windows e Linux. Esse estudo proporcionará um conhecimento aprofundado das características e funcionalidades de cada sistema operacional, bem como das medidas de segurança implementadas. Serão exploradas as especificidades de cada sistema e as diferenças no gerenciamento do processo de swapping.

Um aspecto crucial da pesquisa foi o estudo aprofundado do método de swapping, que é utilizado pelos sistemas operacionais para trocar dados entre a memória principal e a memória secundária. Serão examinados os mecanismos de funcionamento do swapping, incluindo os algoritmos utilizados para seleção de páginas. Esse estudo permitirá compreender as implicações do processo de swapping na segurança e na integridade dos dados.

Ao longo de toda a pesquisa, foram seguidas as normas e os padrões éticos estabelecidos para a realização de estudos científicos. Todas as medidas necessárias serão adotadas para garantir a confidencialidade e a integridade dos dados utilizados nos testes.

1.6. ESTRUTURA DO PROJETO

O presente trabalho está organizado em 6 seções. A Seção 1, Introdução, apresenta os objetivos e as justificativas para a execução do trabalho. A seção 2 aborda o processo de criação e o ciclo que um programa percorre dentro do sistema operacional. A seção 3 aborda o conceito da técnica de *swapping*. A seção 4 relata possíveis vulnerabilidades quando se trata dos dados gerados em arquivos *swap*. A seção 5, aborda as duas principais maneiras de se prevenir contra os tipos de vulnerabilidades apresentadas.

2. ESTRUTURA DE UM PROCESSO DENTRO DO SISTEMA OPERACIONAL

A gerência de processo é uma função exclusiva do sistema operacional que deve controlar a execução dos diversos programas e o uso concorrente do processador e demais recursos, Monteiro (2006). Uma das principais funções do sistema operacional é a gerência de processos, que permite aos programas alocar recursos, compartilhar dados, trocar informações e sincronizar suas execuções.

O sistema operacional deve gerenciar a criação de ciclos de processos dentro da máquina, sendo assim, para que um processo seja gerado dentro do sistema, existe um ciclo de criação de processos, nesta etapa do trabalho vamos nos aprofundar neste ciclo de criação de um processo dentro do sistema operacional.

2.1. ETAPAS DO CICLO DE INSTRUÇÃO

O ponto inicial deste ciclo de criação de processos se encontra na solicitação de criação de um processo (instrução), ou seja,o usuário ou o próprio *S.O.* (sistema operacional) requisita a criação de um processo,

o processador busca a instrução a ser executada na memória principal, armazena estainstrução no registrador de instruções, decodifica seus bits e realiza a execução da instrução. Esta instrução inclui dentro de si três grupos de informações essenciais sobre este processo, tais como, identificação, quotas e privilégios.

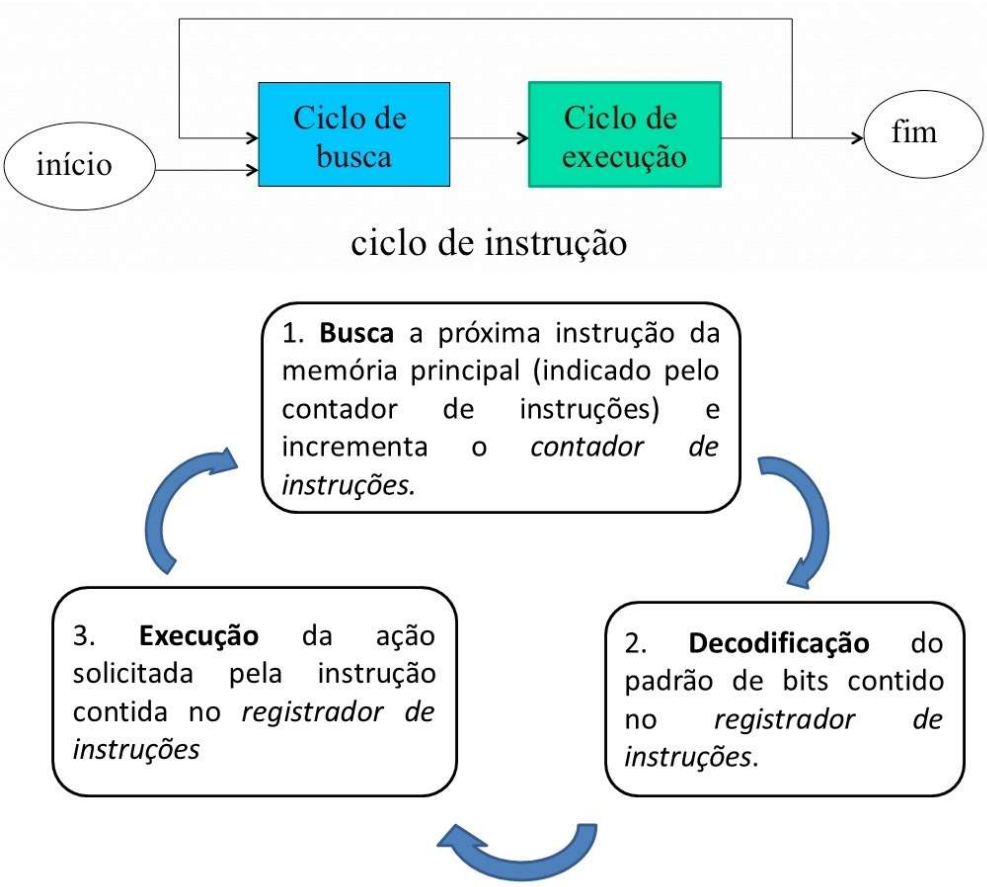

Figura 1 : Representação das etapas do ciclo de instrução (Fonte: Wiki IFSC, s.d., Fig085_MI1022806.png).

2.2. IDENTIFICAÇÃO DO PROCESSO

No ciclo de criação de um processo em sistemas operacionais, a identificação de um processo é uma das etapas essenciais para o correto funcionamento do sistema.

A identificação de um processo que consiste em atribuir um identificador único a cada processo criado no sistema. Esse identificador é usado para que o sistema possa gerenciar os processos, identificando-os e permitindo a realização de operações específicas em cada um deles.

O identificador de processo geralmente é composto por um número inteiro, que pode ser atribuído de forma sequencial ou por meio de um algoritmo específico. Esse número é único para cada processo e é utilizado pelo sistema para referenciar o processo em operações como escalonamento, comunicação entre processos, sincronização de processos, entre outras.

Além disso, a identificação de um processo também pode envolver a atribuição de outros recursos, como espaços de memória, dispositivos de E/S, prioridades de execução, entre outros.

Portanto, a identificação de um processo é uma etapa fundamental no ciclo de criação de processos, pois permite que o sistema possa gerenciar adequadamente os processos criados, garantindo o correto funcionamento do sistema operacional.

Cada processo criado pelo sistema recebe uma identificação única conhecida como (PID – *process identification*) representada por um número, Monteiro(2006). Por meio desse número de identificação, o sistema operacional e outros processos podem referenciar qualquer processo existente, consultando seu contextoou alterando uma de suas características. Alguns sistemas, utilizam além do **PID** também um nome dado ao processo para a identificação do mesmo. O processo também possui a identificação do usuário ou processo que o criou (processo pai). Todo usuário possui uma identificação única no sistema, chamada **UID** (*User Identification*), designada ao processo no momento de sua criação.

2.3. QUOTA

A quota de um processo é um dos aspectos importantes no ciclo de criação de um processo em sistemas operacionais. A quota se refere a um conjunto de limites ou restrições que são impostas pelo sistema em relação a um processo específico.

Essas limitações incluem o consumo máximo de recursos, como o uso de CPU, memória, espaço em disco, número de arquivos abertos, entre outros. Essas limitações podem ser definidas pelo sistema operacional para garantir que um processo não monopolize recursos do sistema ou para evitar que um processo mal-intencionado cause problemas no sistema.

As quotas de processos são geralmente definidas pelo administrador do sistema ou por um usuário com privilégios administrativos. Elas podem ser aplicadas em processos individuais ou em grupos de processos. Por exemplo, um administrador de sistema pode definir uma quota para um determinado processo para limitar o uso de CPU e memória para evitar que ele sobrecarregue o sistema.

As quotas também são úteis para garantir a segurança do sistema. Por exemplo, elas podem ser usadas para limitar o tamanho de um arquivo que um processo pode criar, para evitar que um usuário mal-intencionado sobrecarregue o sistema com arquivos grandes.

Em resumo, a quota de um processo é um conjunto de limitações impostas pelo sistema operacional que são aplicadas a um processo específico, a fim de garantir que o processo não consuma excessivamente os recursos do sistema ou cause problemas no sistema.

A **Quota** são os limites de cada recurso do sistema que um processo pode alocar, Monteiro (2006). Caso a quota do processoseja insuficiente, o processo poderá ser executado lentamente, interrompido durante seu processamento ou mesmo não ser executado. Alguns parâmetros utilizados pela quota dentro do processo são, número máximo de arquivos abertos simultaneamente,tamanho máximo de memória principal e secundaria que o processo pode alocar, numero máximo de operações de entrada e saída pendentes, dentre outros.

2.4. PRIVILEGIOS

O contexto de privilégios de um processo é importante no ciclo de criação de um processo porque determina as permissões e os recursos que um processo pode acessar e utilizar durante a sua execução.

Quando um processo é criado em um sistema operacional, ele herda o contexto de privilégios do processo pai que o criou. Esse contexto inclui informações como as permissões de acesso a arquivos, diretórios e dispositivos do sistema, as quotas de uso de memória e processamento, e os privilégios de execução de comandos de sistema.

Dependendo do contexto de privilégios, um processo pode ter mais ou menos permissões para realizar determinadas ações. Por exemplo, um processo com privilégios elevados pode ter acesso a recursos do sistema que outros processos não têm, como a capacidade de modificar arquivos do sistema ou realizar operações de rede de baixo nível. Já um processo com privilégios limitados pode ter acesso apenas a recursos específicos do usuário, como a pasta de documentos.

É importante que o contexto de privilégios seja gerenciado cuidadosamente durante o ciclo de vida do processo, para garantir que o sistema operacional esteja protegido contra possíveis vulnerabilidades e ataques maliciosos. Por isso, os sistemas operacionais geralmente implementam mecanismos de segurança e controle de acesso que permitem que os administradores de sistemas ajustem as permissões e privilégios de cada processo em tempo de execução.

Por fim, a terceira informação contida na criação do processo são os privilégios, ou seja, direitos que definem as ações que um processo pode ou não fazer em relação a ele mesmo, aos demais processo e ao sistema operacional. Os privilégios dizem muito sobre o processo, pois dependendo do privilegio obtido no processo ele poderá

fazer alterações no próprio processo ou em regras do sistema operacional.

Após esta solicitação ser efetivada, inicia a etapa de alocação de espaço de memória, esta etapa é de extrema importância para este trabalho, pois é nela que a utilização do swapping pode ser necessária.

O sistema operacional reserva um espaço na memória para o novo processo criado, esse espaço é usado para armazenar o código do programa, as variáveis e os dados necessários para a execução do processo.

Após a alocação do espaço de memória o programa a ser executado pelo novo processo é carregado na memória. Isso geralmente envolve a leitura do código do programa a partir do disco e sua transferência para a memória principal.

Quando o programa é carregado na memória e sua leitura é feita, o sistema operacional executa a rotina de inicialização do processo, que inclui a configuração dos registradores e outras estruturas de dados necessárias para a execução do programa. O processo é executado pelo processador, seguindo as instruções contidas no código do programa. Após sua conclusão, o sistema operacional libera o espaço de memória alocado pra o processo e removo o identificador do processo do sistema. Essas etapas são essenciais para o ciclo de vida de um processo dentro do **SO**.

O gerenciamento adequado do processo, desde sua criação até a sua conclusão,é fundamental para garantir a estabilidade do sistema e a segurança dos dados, a execução perfeita deste ciclo garante também a eficácia do sistema operacional e umaótima experiência ao usuário da máquina.

Figura 2 : Ilustração das características da estrutura de um processo conforme apresentado por Machado e Maia (2013) em Arquitetura de sistemas operacionais (5a ed.), Rio de Janeiro.

3. TÉCNICA DE SWAPPING

Swapping é uma técnica em sistemas operacionais que permite a transferência de dados entre a memória principal (RAM) e a memória secundária (disco rígido) quando a memória principal está cheia ou quase cheia impossibilitando o ciclo de criação de processos, mencionado anteriormente (Tanenbaum & Bos, 2021, p. 149).

Quando um programa é executado, ele é carregado na memória principal para que a CPU possa acessá-lo rapidamente. No entanto, a quantidade de memória disponível na memória principal é limitada, o que significa que se muitos programas estiverem sendo executados simultaneamente, a memória principal pode ficar cheia e não haverá espaço suficiente para carregar mais programas. Nesse caso, o sistema operacional pode usar a técnica de *swapping*para liberar espaço na memória principal, movendo partes do conteúdo armazenado na memória principal para a memória secundária e carregando novos programas na memória principal.

O processo de *swapping* envolve duas etapas principais: a primeira é a escolha dos dados que serão transferidos para a memória secundária e a segunda é a transferência dos dados selecionados.

Na primeira etapa, o sistema operacional escolhe quais blocos na memória principal serão transferidos para a memória secundária. Normalmente, os processos ociosos são armazenados em disco em sua maior parte, portanto não ocupam qualquer memória quando não estão sendo executados. Essa técnica de gerenciamento de memória, conhecida como "swapping", é amplamente utilizada em sistemas operacionais modernos (Saliba Júnior, 2017). Na segunda etapa, o sistema operacional transfere os dados selecionados da memória principal para a memória secundária, liberando espaço na memória principal. Quando o programa precisar dos dados que foram movidos para a memória secundária, o sistema operacional poderá acessá-los novamente, transferindo-os de volta para a memória principal.

3.1. MOTIVIVO DA UTILIZAÇÃO DA TECNICA

Segundo Saliba Júnior (2017), mesmo com a constante evolução dos hardwares, o aumento da multiprogramação e da gerencia de memória, muitas vezes um programa não conseguia ser executado por falta de uma partição livre disponível, ou seja, por falta de espaço na memória principal, muitas vezes um programa não podia continuar ou iniciar a sua execução por conta deste motivo.

Sendo assim, após anos de pesquisas e tentativas para solucionar o problema de espaço de memória, foi desenvolvida a técnica de swapping. Esta técnica consiste em utilizar a capacidade total da memória principal e a memória secundária, criando uma chamada memória virtual e assim utiliza todo o espaço de memória para realizar a troca de dados entre uma unidade e outra, para manter o sistema executando os processos sem interrupções por falta de espaço na memória.

3.2. MEMÓRIA VIRTUAL

O conceito de memória virtual se baseia em criar uma memória utilizando a junção da memória principal e a memória secundaria, criando a ilusão para o usuário de um espaço de memória muito maior do que apenas a da memória principal. Quando o programa é executado o endereçamento feito pelo programa não se vincula em endereços físicos da memória principal, ou seja, programas e suas estruturas de dados deixam de estar limitados ao tamanho da memória principal física disponível, pois podem possuir endereços associados a memória secundaria. (Saliba Júnior, 2017).

A memória virtual é uma técnica que permitiu a execução de programas maiores do que a memória física principal, movendo rapidamente pedaços de processos entre a memória RAM e o disco. A memória virtual também permitiu que um programa se conectasse diretamente a uma biblioteca no momento da execução em vez de faze-lo no momento da compilação, fazendo com o processo levasse menos tempo durante sua execução. A ideia de memória virtual se propagou entre computadores de grande porte e em até mesmo microcomputadores, sendo usada em sistemas UNIX e Windows. A memória virtual usa também uma técnica chamada de paginação, que permite que os pedaços e partes de muitos programas estejam na memória simultaneamente.

3.3. SISTEMAS DE ARQUIVOS

Dentro de um sistema operacional é de grande importância o gerenciamento de arquivos, pois ele garante a organização e movimentação destes dados, o armazenamento e a recuperação de informações é uma atividade essencial para qualquer tipo de aplicação, ou seja, um processo deve ser capaz de ler e gravar de forma permanente grande volume de dados em diferentes dispositivos, Saliba Júnior (2017).

De acordo com Tanenbaum & Bos (2014), os sistemas de arquivos são uma parte fundamental dos sistemas operacionais, responsáveis por gerenciar a organização, armazenamento e acesso a arquivos e diretórios. Eles fornecem uma interface para que os usuários possam criar, modificar, mover e excluir arquivos, além de proteger esses arquivos contra acesso não autorizado. Os sistemas de arquivos também são responsáveis por gerenciar o espaço em disco disponível e garantir que os arquivos sejam armazenados de forma eficiente. Eles podem ser implementados de várias maneiras, incluindo listas encadeadas, mapas de bits e paginação.

Segundo os autores, os sistemas de arquivos na nuvem e a procedência dos dados têm ganhado atenção recentemente. Isso ocorre devido à crescente utilização de serviços e armazenamento remoto, o que oferece vantagens em termos de escalabilidade, acessibilidade de qualquer lugar e backups automáticos. Entretanto, essa abordagem também apresenta novos desafios em termos de segurança e controle de dados.

Portanto, conforme destacado no livro "Sistemas Operacionais Modernos," os sistemas de arquivos desempenham um papel essencial ao permitir que os usuários gerenciem seus arquivos de forma eficiente e segura, tanto localmente quanto na nuvem. A evolução contínua desses sistemas e o avanço tecnológico reforçam ainda mais sua importância no uso cotidiano e na proteção dos dados.

3.4. ARQUIVO

Os arquivos são uma parte fundamental dos sistemas de computação, desempenhando um papel essencial ao permitir que os usuários armazenem e acessem informações de forma eficiente e segura. Esses arquivos podem abranger diversos tipos de dados, incluindo texto, imagens, áudio e vídeo, (Saliba Júnior, 2017).

Para organizar e gerenciar esses dados, os arquivos são organizados em diretórios, proporcionando uma estrutura hierárquica que facilita a localização e a administração de arquivos relacionados.

Através dessa interface, os usuários têm a flexibilidade de criar, modificar, mover e excluir arquivos, garantindo também a devida proteção contra acessos não autorizados. Com o intuito de otimizar o uso do espaço em disco e preservar informações confidenciais, os arquivos podem ser compactados e criptografados.

Além disso, a possibilidade de compartilhar arquivos entre usuários e sistemas torna-se uma funcionalidade valiosa, permitindo colaboração e a transferência de dados de forma eficiente e segura.

Em suma, a importância dos arquivos como uma parte essencial dos sistemas de computação é inegável, conforme enfatizado na obra "Sistemas Operacionais Modernos." Eles capacitam os usuários a armazenar e acessar uma diversidade de informações, enquanto contribuem para a organização e segurança dos dados em um ambiente computacional.

3.5. DIRETORIO

Os diretórios, também conhecidos como pastas, desempenham um papel fundamental nos sistemas operacionais ao organizar e armazenar arquivos em dispositivos de armazenamento, como discos rígidos. Essas estruturas de dados possibilitam uma organização lógica dos arquivos presentes no disco, proporcionando aos usuários um acesso e gerenciamento mais eficiente de seus dados.

Em cada diretório, são mantidas listas de entradas associadas aos arquivos, contendo informações cruciais, como localização física, nome, organização e atributos. A flexibilidade dos diretórios é evidenciada pela capacidade de organizar-se em uma hierarquia de subdiretórios, permitindo que os usuários criem uma estrutura de pastas e subpastas, a fim de organizar seus arquivos de maneira mais lógica e intuitiva.

Tais diretórios são amplamente empregados nos sistemas operacionais modernos, abrangendo computadores pessoais, servidores e outros dispositivos de computação. Proporcionam aos usuários uma gestão de arquivos mais eficiente e organizada, tornando a localização e acesso aos arquivos uma tarefa mais simples e descomplicada quando necessário.

4. VUNERABILIDADES ENCONTRADAS EM UM AMBIENTE DE SWAPPING

No cenário global da computação, o processo de swapping se tornou uma ferramenta essencial, trazendo muitos benefícios que impulsionaram o avanço tecnológico. No entanto, ao aproveitar essas vantagens, também é necessário estar atento às possíveis vulnerabilidades que podem surgir. Neste projeto serão apresentadas algumas dessas falhas, que ocorrem em um ambiente swapping.

4.1. TABELA DE VUNERABILIDADES E RESPSCTIVAS DEFESAS

Com base no que foi mostrado ao longo de todo este projeto, agora será apresentada uma tabela que contém as falhas encontradas, os sistemas operacionais relacionados e, por fim, a técnica utilizada para a proteção dos dados. Esta tabela é fundamental para consolidar as informações e fornecer uma visão abrangente das vulnerabilidades identificadas, bem como das medidas de segurança implementadas.

Sistema Operacional	**Vulnerabilidade Swapping**	**Método de Segurança**
Windows	Troca de dados	• Criptografia de disco
	Dados remanescentes	• Criptografia de disco • *Loking* de memória
	Sincronização de memória	• Hardwares de mesma compatibilidade
Linux	Troca de dados	• Criptografia de disco
	Dados remanescentes	• Criptografia de disco
	Sincronização de memória	• Criptografia de disco • *Loking* de memória • Configuração de permissões do usuário

Tabela 1 - Tabela de vulnerabilidades com os respectivos sistemas operacionais e métodos de segurança

Em seguida será apresentada cada vulnerabilidade mostrada na tabela a cima, explicando como cada uma delas funciona, seguido da explicação sobre as técnicas de segurança para a proteção dos dados contra cada uma das falhas citadas.

4.2. TROCA DE DADOS

Em um ambiente de *swapping* pode-se encontrar alguns riscos, os dados que estão sendo transferidos da memória principal para a memória secundária podem ser armazenadostemporariamente em disco. Isso significa que, se um disco rígido ou outro dispositivo de armazenamento secundário for acessado por um usuário não autorizado, esses dados confidenciais poderão ser lidos ou copiados (Tanenbaum & Bos, 2021,p.149).

Isso representa um risco significativo de segurança para empresas ou organizações que lidam com informações confidenciais, como dados de clientes, informações financeiras ou dados depropriedade intelectual. Se esses dados caírem nas mãos erradas, eles podem ser usados para atividades maliciosas, como fraude, roubo de identidade ou espionagem.

Segundo St Denis (2007, p. 42) a exposição de dados confidenciais pode ocorrer devido ao fato de que os dados estão sendomovidos da memória principal para a memória secundaria, sendo assim ficando expostos para outros processos ou usuários maliciosos durante essa transição, podendo resultar na exposição e acesso a estes dados.

4.3. DADOS REMANECENTES

Outra vulnerabilidade durante este processo são os dados remanescentes. Quando os dados são movidos da memória principal para a memória secundária, algumas informações podem permanecer na memória principal mesmo após a transferência. Esses dados remanescentes podem ser acessados por usuários mal intencionados ou por outros processos que podem levar a exposição de dados confidenciais. St Denis (2007, p. 47)

4.4. SINCRONIZAÇÃO DE MEMÓRIA

Uma falha também observada é o problema de sincronização. O ***Swapping*** envolve a transferência de dados como já explicado anteriormente, sendo assim, o problema de sincronização entre essas duas memórias pode resultar em perda de dados durante este processo ou até mesmo em dados corrompidos dependendo do grau de falta de sincronia entre as memórias.

5. PROTEÇÃO

Os resultados obtidos revelam a existência de duas abordagens predominantes para proteger dados sensíveis durante operações de *swap memory*: o *locking* de memória e a criptografia de disco. O conceito do *locking* de memória reside na capacidade dos aplicativos restringirem o acesso à memória, mitigando a exposição de informações sensíveis durante o processo de swap *memory*. Por outro lado, a criptografia de disco emerge como uma técnica que assegura que, mesmo se as informações forem comprometidas durante a operação de swap, sua decodificação permanece inacessível sem a chave criptográfica apropriada.

5.1. LOKING DE MEMÓRIA

A implementação do *locking* de memória é realizada pelo próprio sistema operacional e se destina a aplicativos que requerem a salvaguarda de informações sensíveis. Em sistemas operacionais fundamentados no kernel NT, como o *Windows*, o *locking* de memória se apresenta de maneira voluntária, possibilitando que o sistema operacional decida transferir dados não-críticos para o disco rígido. Em contrapartida, em ambientes baseados no kernel POSIX, exemplificados pelo *Linux* e *BSD*, funções específicas, a exemplo de *mlock*(), *munlock*() e *mlockall*(), facultam aos programas o bloqueio da memória a ser protegida.

Importante salientar que o emprego do *locking* de memória pode induzir impactos no desempenho do sistema, dado que ele obstaculiza a transferência de memória para o disco rígido, potencialmente resultando em um aumento da utilização da memória RAM. Nesse sentido, é prudente que os programas façam uso do *locking* de memória exclusivamente com o intuito de resguardar informações sensíveis, liberando a memória bloqueada assim que a necessidade se dissipe.

À medida que a busca por ambientes computacionais seguros ganha destaque, o *locking* de memória emerge como uma ferramenta vital no arsenal de estratégias de proteção. Ao habilitar aplicativos a defenderem informações sensíveis contra acessos indesejados, esta abordagem reforça a necessidade de um equilíbrio entre a segurança da informação e o desempenho do sistema.

5.2. CRIPTOGRAFIA DOS DADOS

Outra abordagem essencial para manter a integridade e a confidencialidade dos dados é a criptografia. A criptografia de disco representa uma técnica de segurança crucial, abarcando a codificação integral do conteúdo de um disco rígido ou de uma partição específica. Tal procedimento transforma os dados em algo ilegível para qualquer indivíduo sem acesso à chave de criptografia correspondente. Esse método se destina a preservar informações sensíveis, como registros pessoais, dados financeiros e informações empresariais, resguardando-as contra acessos não autorizados. St Denis (2007, p. 42).

No âmbito da criptografia de disco, duas abordagens primordiais se destacam: a criptografia de disco completo e a criptografia de disco parcial. Na primeira, ocorre a codificação de todo o conteúdo do disco, compreendendo o sistema operacional, arquivos do usuário e componentes do sistema. Já na segunda, apenas uma porção do disco é submetida à criptografia, coexistindo com setores não criptografados.

A implementação da criptografia de disco pode transcorrer por via de software ou hardware. No enfoque de software, o próprio sistema operacional ou um programa designado assume a responsabilidade de codificar e decodificar os dados armazenados. Por outro lado, na abordagem de hardware, um dispositivo externo, como um cartão inteligente ou um módulo de segurança dedicado, assume o papel de executar as operações criptográficas.

A aplicabilidade da criptografia de disco abrange diversas instâncias, abrangendo computadores pessoais, servidores, dispositivos móveis e dispositivos de armazenamento externo. Cumpre sublinhar que a criptografia de disco não constitui uma solução de segurança autônoma, mas sim um componente suplementar de salvaguarda. Medidas de segurança adicionais, tais como senhas robustas, autenticação de dois fatores e atualizações regulares de software, são igualmente essenciais para assegurar a inviolabilidade dos dados. Em conjunto, essas estratégias estabelecem uma postura resiliente e holística para a proteção de informações sensíveis em ambientes computacionais. St Denis (2007, p. 45)

6. CONCLUSÃO

É fundamental ressaltar que, apesar das vulnerabilidades potenciais inerentes aos processos de swap *memory*, um sistema operacional adequadamente protegido pode gerenciar e preservar os dados de maneira eficaz durante a aplicação dessa técnica. O desenvolvimento de sistemas operacionais com camadas de segurança sólidas tem a capacidade de mitigar os riscos associados à exposição de informações sensíveis no contexto do swapping.

Conclui-se, portanto, que a preservação de dados sensíveis durante os procedimentos de swap *memory* assume uma posição de destaque no cenário da segurança da informação. Aplicativos e sistemas devem incorporar de maneira proativa abordagens de proteção, abraçando tanto o *locking* de memória quanto a criptografia de disco, a fim de guardar a integridade das informações sensíveis. Além disso, conscientizar os usuários sobre os riscos subjacentes aos processos de swap *memory*, encorajando a adoção de práticas exemplares de segurança da informação para a preservação de seus dados confidenciais.

Nesse contexto, este estudo não apenas evidencia a crítica importância da proteção de informações sensíveis nos procedimentos de swap *memory*, mas também destaca a necessidade contínua de desenvolver e implementar estratégias de segurança sólidas para proteger dados vitais no cenário digital contemporâneo. A atenção a este contexto emerge como uma imperativa, ressaltando o valor intrínseco de um ecossistema seguro de informações. Ainda assim, é notável que a escassez de documentos disponíveis para a pesquisa nesta área chama a atenção para a necessidade de investigações aprofundadas e documentação abrangente, a fim de orientar futuras iniciativas de segurança e pesquisa.

7. REFERÊNCIAS

Machado, F. B., & Maia, L. P. (2002). Arquitetura de sistemas operacionais (5a ed.). LTC.

Machado, F. B., & Maia, L. P. (2013). Arquitetura de sistemas operacionais (5a ed.). Rio de Janeiro.

Maziero, C. A. (2020). Sistemas operacionais: conceitos e mecanismos (2a ed.). Curitiba: UTFPR.

Monteiro, M. A. (2006). Introdução à Organização de Computadores (5a ed.). Rio de Janeiro.

Saliba Júnior, E. (2017). Introdução a Arquitetura de Sistemas Operacionais - Parte 02. Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro. Recuperado de https://esj.eti.br/IFTM/Disciplinas/Grau02/SOL/SOL_Unidade_04.pdf.

St Denis, T. (2007). Cryptography for Developers (1a ed.). Syngress.

Tanenbaum, A. S., & Bos, H. (2014). Sistemas Operacionais Modernos (4ª ed.). Prentice Hall.