**BTRFS e gerenciamento de arquivos**

O BTRFS (B-tree file system) é um sistema de arquivos projetado para atender aos requisitos de expansão de escalabilidade de grandes subsistemas de armazenamento desenvolvido em 2007 pela Oracle, com objetivo de fornecer recursos avançados de gerenciamento de dados, como gerenciamento de disco, snapshots, compressão, criptografia e suporte a RAID (Redundant Array of Indepedent Disks), baseado em um modelo de gravação COW (cópia na gravação), ou seja, não atualiza os dados no local, mas escreve uma nova cópia em um local diferente e, em seguida, altera atomicamente os ponteiros.

Esse sistema de gerenciamento utiliza a estrutura de arvore B, assim permitindo o acesso e atualização com eficiência de grandes blocos de dados, independentemente do tamanho da árvore B, pois organiza seus dados de tal maneira que permita a inserção ou remoção de dados com uma complexidade assintótica de e busca tendo a complexidade de sendo “m” a ordem da árvore, assim tornado ela bastante eficiente.

Figura X – Arvore B de Ordem 2

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

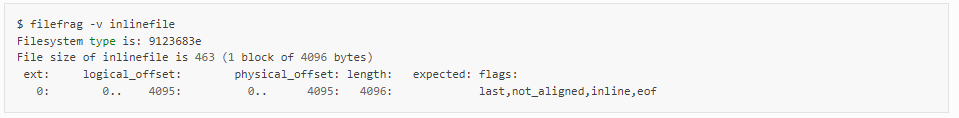
XFonte: Disponível em: <https://pt.wikipedia.org/>. Acesso em: 17 maio 2023.

<https://pt.wikipedia.org/wiki/%C3%81rvore_B>

Ele possui um tamanho máximo teórico de arquivos de 16 EiB (Exbibyte) ou bytes, sendo que o limite prático é de 8 EiB devido às limitações o VFS, pois ele utiliza armazenamento baseado em extensão, nesse método, em vez de alocar espaço de arquivos é reservado uma quantidade maior de espaço chamada “extensão” para armazenar os dados de um arquivo. Geralmente as extensões são alocadas no momento da criação do arquivo ou quando mais espaço é necessário para um arquivo existente. Uma extensão é um bloco contíguo de espaço em disco alocado para um arquivo específico, dessa forma trazendo algumas vantagens como redução de fragmentação, desempenho aprimorado e redução de operações de entrada e saída.

Permite embutir arquivos pequenos na seção de metadados nos nós da árvore B, ou seja, sem blocos separados para as extensões, sendo o limite padrão de 2.048 bytes ou 2 KiB (kibibyte) por arquivo, assim evitando fragmentação de disco, aumentando a eficiência de armazenamento e simplicidade de gerenciamento. Um arquivo embutido pode ser identificado enumerando as extensões, por exemplo, pela ferramenta filefrag:

Figura X – File Frag

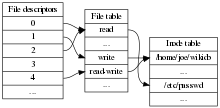


XFonte: Disponível em: <https://btrfs.readthedocs.io/>. Acesso em: 17 maio 2023.

<https://btrfs.readthedocs.io/en/latest/Inline-files.html>

O BTRFS utiliza alocação dinâmica de inodes, refere-se a prática de alocar inodes em um sistema de arquivos de forma flexível e sob demando. O inodes são estruturas de dados utilizados por sistemas de arquivos para armazenar informações sobre os arquivos e diretórios presentes no sistema. Cada arquivo e diretório no sistema de arquivos possui um inode associado a ele. Um inode contém metadados que descrevem algumas características dos arquivos, sendo eles o identificar único, permissões de acesso, proprietário, tamanho do arquivo, data e hora na qual o arquivo foi criado, data e hora da última modificação ou leitura, tipo de arquivo, contagem de links e ponteiros para dados.

Figura X – Tabela de Arquivos e Tabela de Inodes



XFonte: Disponível em: <https://www.wikiwand.com/>. Acesso em: 17 maio 2023.

<https://www.wikiwand.com/pt/N%C3%B3-i>

Também permite snapshots, assim permitido o capturar o estado do sistema em um determinado momento no tempo. Esses snapshots podem ser classificados em duas categorias principais: graváveis (writable) e somente leitua (read-only). Os graváveis permitem alteração e gravação de dados, quando ele é criado contém uma cópia do sistema no momento de sua criação. Já os somente leitura são apenas para leitura e não permitem alterações, são úteis para fins de backup, recuperação de dados e manutenção do estado do sistema.

Outro recurso que o BTRFS utiliza é o reparo automático ao ler, assim quando um aplicativo (software) tenta ler dados ou metadados que estão danificados, podem ser recuperados caso o sistema de arquivos tenha outra cópia válida ao usar perfil de grupo de blocos com redundância.

BTRFS suporta compactação de arquivos transparente. Existe três tipos de algoritmos disponíveis: ZLIB, LZO e ZSTD, com vários níveis. Sendo o ZLIB com taxa de compressão mais lenta e mais alta, possuindo 9 níveis sendo o padrão 3, LZO mais rápido que a ZLIB, entretanto sua taxa de compactação é pior e por fim a ZSTD, taxa de compressão comparável ao ZLIB com velocidade de compressão mais alta, possuindo 15 níveis diferentes.

**Recuperação de Sistema (SnapShot)**

A ferramenta de recuperação de sistema utiliza o sistema de snapshot do BTRFS para poder capturar o estado do sistema em um determinado momento no tempo. Esses snapshots podem ser classificados em duas categorias principais: graváveis (writable) e somente leitua (read-only). Os graváveis permitem alteração e gravação de dados, quando ele é criado contém uma cópia do sistema no momento de sua criação. Já os somente leitura são apenas para leitura e não permitem alterações, são úteis para fins de backup, recuperação de dados e manutenção do estado do sistema.

Figura X – SnapShot BTRFS

Diagrama

Descrição gerada automaticamente

XFonte: Disponível em: <https://gdevillele.github.io>. Acesso em: 17 maio 2023.

<https://gdevillele.github.io/engine/userguide/storagedriver/btrfs-driver/>

O processo para criar um snapshot no BTRFS envolve várias etapas que ocorrem em um nível mais baixo do sistema de arquivos. Primeiro, é preparada uma estrutura de metadados chamada Metadata Root (raiz de metadados) do snapshot. Essa raiz de metadados armazenará as informações específicas do snapshot. Normalmente são referentes a estrutura da arvore de metadados, ponteiros para inodes, arvore de alocação, relacionamentos entre snapshots e informações sobre compressão e criptografia.

Uma vez que os metadados são copiados, ocorre a atualização dos ponteiros de blocos nos inodes do snapshot. Esses ponteiros de blocos são atualizados para referenciar os mesmos blocos de dados dos snapshots anteriores. Essa abordagem de compartilhamento de blocos permite economizar espaço, pois os blocos de dados não são duplicados para cada snapshot.

Quando há modificações em arquivos após a criação do snapshot, o BTRFS utiliza a técnica de Copy-on-Write (COW). Isso significa que os dados modificados são copiados para local diferente e atualiza os inodes do snapshot, enquanto os snapshots anteriores continuam a apontar para os blocos de dados originais. Isso garante que cada snapshot seja uma representação consistente do sistema de arquivos em um determinado momento.

Além disso, a árvore de relacionamentos de snapshots é atualizada para incluir o novo snapshot como um filho do snapshot original. Essa estrutura de árvore permite a organização hierárquica dos snapshots, assim facilitando na navegação entre eles e na criação de uma linha do tempo do sistema de arquivos.

Durante todo o processo de criação do snapshot, são tomadas precauções para garantir a consistência e a integridade dos dados. Utiliza-se de transações para garantir que as operações sejam atômicas, ou seja, todas as alterações necessárias para criar o snapshot ocorrem de forma completa e coesa, assim garantindo um estado consistente do sistema.

**Interface (GNOME)**

O Ubuntu utiliza em sua interface o GNOME, ele é uma interface de usuário amplamente utilizada em várias distribuições Linux. Ele oferece uma experiencia moderna, intuitiva e personalizada para os usuários. Foi desenvolvido utilizando a biblioteca GTK+ e utiliza principalmente a linguagem de programação C em seu desenvolvimento.

Figura X – GNOME no Ubutu

Interface gráfica do usuário

Descrição gerada automaticamente

XFonte: Disponível em: <https://ubuntu.com>. Acesso em: 18 maio 2023.

<https://ubuntu.com/blog/whats-new-in-ubuntu-desktop-22-10-kinetic-kudu>

O GNOME também é composto por outros componentes fundamentais, tais como Mutter para realizar o gerenciamento de janelas do GNOME, composição da janela, efeitos visuais e manipulação de eventos.

Algumas das estruturas de dados mais utilizadas do GNOME é estrutura de arvore, pois, a estrutura de diretórios e arquivos que é utilizada pelo Nautilus é organizada em uma árvore geral.

Figura X – Estrutura de Dados Utilizada pelo Nautilus

Desenho em preto e branco

Descrição gerada automaticamente

XFonte: Disponível em: <https://pt.wikibooks.org/>. Acesso em: 18 maio 2023.

<https://pt.wikibooks.org/wiki/Algoritmos_e_Estruturas_de_Dados/%C3%81rvore>

**Referencias**

Diolinux. **BTRFS, o sistema de arquivos “do futuro”**, 2020. Disponível em: <https://diolinux.com.br/editorial/btrfs-o-sistema-de-arquivos-do-futuro.html>. Acesso em: 17 maio 2023.

Oracle. **Managing Local File System**. Disponível em: <https://docs.oracle.com/en/operating-systems/oracle-linux/8/fsadmin/index.html>. Acesso em: 17 maio 2023.

Archive Kernel. **Managing Local File System**. Disponível em: <https://archive.kernel.org/oldwiki/btrfs.wiki.kernel.org/index.php/Main_Page.html>. Acesso em: 17 maio 2023.

UPS. **Árvores B (B-trees)**, 2019. Disponível em: <https://www.ime.usp.br/~pf/estruturas-de-dados/aulas/B-trees.html>. Acesso em 17 maio 2023.

Documentação BTRFS: <https://btrfs.readthedocs.io/en/latest/> Não sei como colocar

E toda pagina <https://www.gnome.org/>