Incident Response - Understanding B Tree filesystems

Taz Wake | Halkyn Consulting Ltd | Originally published 2023-11-23

Introduction to B-Trees

B-Trees are a class of self-balancing m-way trees, where 'm' defines the order of the tree. They generalize the binary search tree, allowing a node to have more than two children based on the value of 'm'. The data within B-Trees is maintained in a sorted order, with lower values on the left subtree and higher values on the right. Notably, all leaf nodes in a B-Tree must exist at the same level. Each node in a B-Tree can have a maximum of (m-1) keys and a minimum determined by the ceiling function of (m/2 - 1), except for the root node, which has a minimum of one key. Notable for their balancing properties, B-Trees ensure that the node is always inserted at the leaf level.

B+ Trees

B+ Trees, considered an advanced self-balanced tree, ensure that every path from the root to the leaf has the same length, meaning all leaf nodes are at the same level. The primary structure of a B+ tree includes a root, internal nodes, and leaves, with the root being either a leaf or a node with two or more children. The node structure of a B+ tree contains n-1 key values and n pointers, with the keys kept in sorted order. The B+ tree's efficient storage of records in an indexed manner makes data access faster and easier. Each non-leaf node in a B+ tree can hold up to 'b' pointers (where 'b' is the order of the tree) and the leaf nodes are connected to facilitate sequential access.

Differences Between B-Trees and B+ Trees

B-Trees and B+ Trees, while similar, have distinct differences:

- **Storage of Keys and Records**: In B-Trees, both keys and records are stored in all nodes. In contrast, B+ Trees store keys in internal nodes and records in leaf nodes. This distinction allows for redundancy in key occurrence in B+ Trees, where keys can be present in internal nodes, but records are uniquely stored in leaf nodes.
- **Linking of Leaf Nodes**: B-Trees do not link leaf nodes, whereas B+ Trees link all leaf nodes to provide efficient sequential access.
- **Search Efficiency**: Searching in B-Trees is less efficient as records are stored in both internal and leaf nodes, while B+ Trees store all records in leaf nodes, making searches quicker.
- **Deletion Process**: Deleting internal nodes in B-Trees is slow and complex, whereas in B+ Trees, it's faster since all records are in leaf nodes.

- **Sequential Access**: Sequential access is not possible in B-Trees but is facilitated in B+ Trees through linked leaf nodes.
- **Tree Dimensions**: B-Trees perform more splitting operations, increasing their height. B+ Trees, on the other hand, are wider rather than taller due to their structure.

File Systems Utilising B-Trees and B+ Trees

Different file systems have adopted B-Tree and B+ Tree structures based on their specific requirements. For instance:

- NTFS: Uses B-Trees for directory and metadata indexing.
- XFS: Employs B+ Trees for metadata indexing.
- EXT4: Utilises extent trees, a modified B+ Tree data structure, for file extent indexing.
- APFS (Apple File System): Adopts B+ Trees for storing mappings from filesystem object IDs to disk locations and for storing filesystem records, although these trees' leaf nodes lack sibling pointers.

Rebalancing in B-Trees and B+ Trees

Rebalancing is crucial in both B-Trees and B+ Trees to maintain their properties during insertions and deletions. When a key is inserted into a full node in a B-Tree, the node is split into two, and the median key is moved up to the parent. In B+ Trees, the leaf nodes are also split when they become full, but since all records are in the leaf nodes, rebalancing can involve moving keys up or redistributing keys among sibling nodes. This rebalancing ensures the trees remain balanced, optimizing search and access times.

Conclusion

Understanding the intricacies of B-Trees and B+ Trees is essential for professionals dealing with data storage and retrieval. These structures are foundational in the design of efficient file systems and databases, with each variant offering unique advantages tailored to specific use cases. The ongoing evolution and adoption of these data structures in various file systems underscore their significance in the field of computer science.

About Halkyn Consulting

Halkyn Consulting Ltd is a UK-based security and risk management consultancy providing 24/7 incident response, digital forensics, threat hunting, virtual security leadership (vCISO) and security compliance support to organisations worldwide. We deliver clear, practical, vendor-independent advice based on real-world threats — no hype, no upsell.

More free security resources are available at www.halkynconsulting.co.uk/security-resources.

Web: www.halkynconsulting.co.uk | **Email:** info@halkynconsulting.co.uk | **Tel:** +44 1244 940858