Internals of ReiserFS

by

Lakshmipathi.G

Blocks :

- A Partition is divided into fixed size blocks.
- Default size of blocks is 4096 bytes or 4KB.
- Maximum possible blocks in a partition is 2^32.
- The first 64KB (or 65536 bytes) free space is allocated to boot loaders in a partition.

Super block :

- Starts after boot code (i.e) super block is at 64KB or at 65536 byte of a partition an it's size is 80 bytes. ext2/ext3 super block size is 1024 bytes.
- By default ,super block is located at 16 block.
- Unlike ext2/ext3 file system, ReiserFS has got only one copy of super block for entire partition.

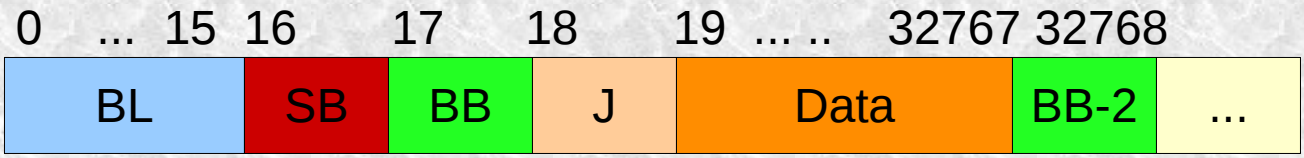
Block Bitmap :

- No.of Blocks mapped in a bitmap directly depends on block size.
- For example,say if a bitmap can map 'k' blocks ,then every 'k'-th block will be a new block bitmap.

Journal:

- Journal is located next to every block bitmap.

Structure of ReiserFS :



BL-BootLoader

SB-Super block

BB-Block bitmap

J-Journal Block

BB-2 – Block bitmap 2

Block Bitmap :

- No.of Blocks mapped in a bitmap directly depends on block size.
- For example,say if a bitmap can map 'k' blocks ,then every 'k'-th block will be a new block bitmap.

Block Bitmap :

Given a block number b , it's status (free or in-use) is determined by,

Step 1:Get block bitmap number

b / (8 * block size) :Block Bitmap No.

Step 2:Get the status of block.
Let s=b mod (8*block size) then,

s / 8 gives 'byte number within Block bitmap'

s%8 gives 'bit position within the byte'

Example : Check the status of block 32798.

Block Number (b) = 32798

Block size = 4096 bytes

Then by step 1:

32798 / (8 * 4096)
=> 32798 / 32768 =1

Block numbered 32798 status is available in Block Bitmap # 1

Step 2:
s=32798 % (8*4096)
s = 32798 % 32768 = 30

Then ,
s / 8 = 30 /8 => 3rd byte within block bitmap #1
s%8 = 30 %8 => 6th bit in byte.

So, the status of 32798 is present in block
bitmap #1 at 6th bit of the 3rd byte.

ReiserFS Tree:

11 points to note :

11) Each node is nothing but a disk block.
10)File System is made up of balanced (B+) tree.
9)Tree has internal nodes (denoted by level > = 2)
and leaf nodes (level = 1)
8)Each object is also called as item.
7)Each object(file,directory or stat item) is assigned to unique key. (This unique key is similar to inode number)
6)Internal nodes composed of keys (or inode number) and pointer to their child nodes ,that is ,pointer to direct/indirect blocks.
5)Always,there is one more pointer than the keys.

ReiserFS Tree:(contd)

4) p0 points to the items that have key smaller than k0 and p1 points to items >= k0 and <k1 .

3)Last pointer points to those items > last key in the node.

2)Each node has a level.

Level 1 denotes these are leaf nodes.

Level 2 and higher denotes internal nodes.

1)root node has the highest level.

ReiserFS Tree:

We have three types of nodes.

- Internal node.
- Leaf or formatted node.
  Contents of these two nodes are sorted of keys.
- Unformatted node.
  No keys

Leaf or Formatted node:

- Block Header
- Item Header
- Items. Item may be:
  - Direct Item
  - Indirect Item
  - Stat Item
  - Directory Item

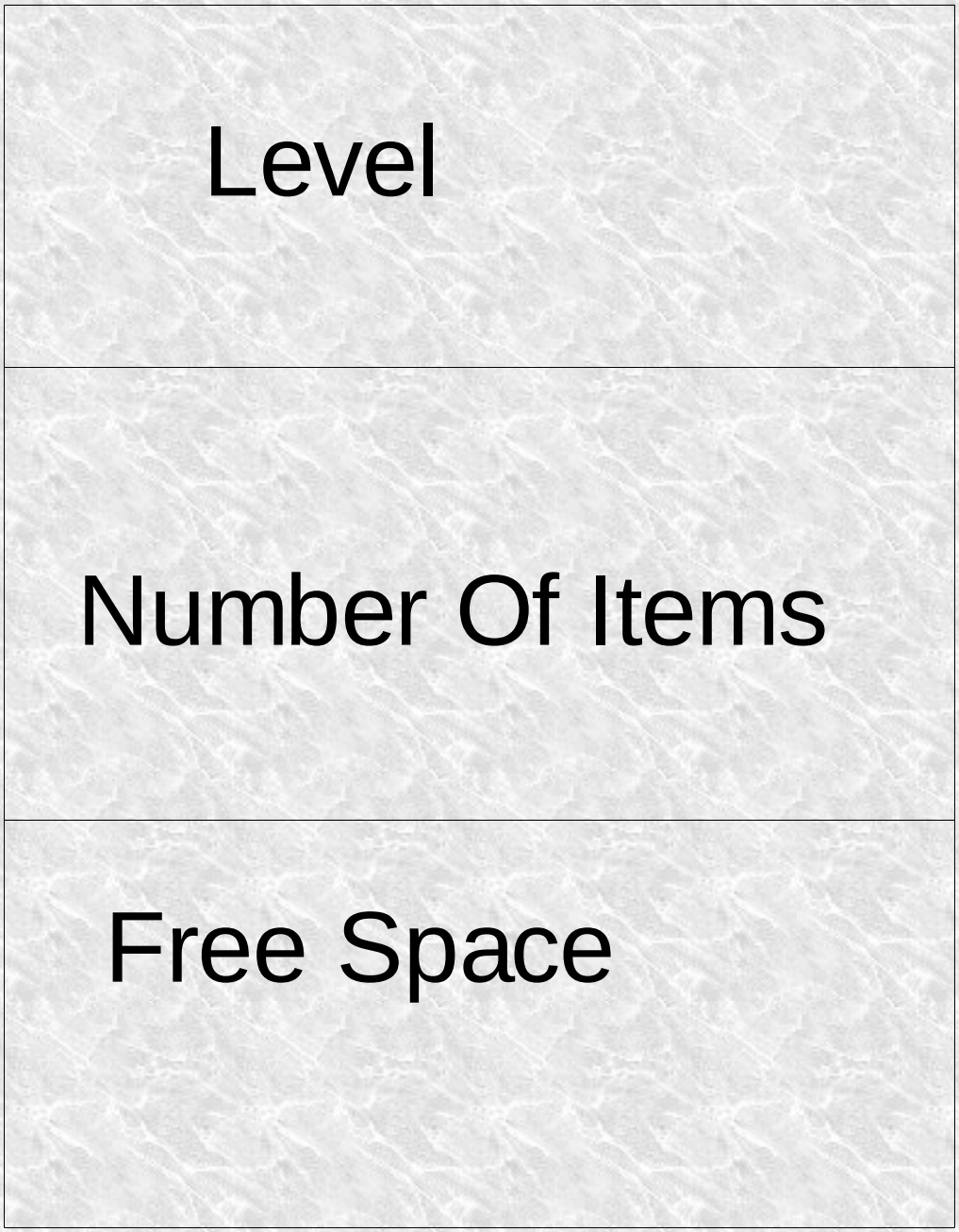
Block Header (BH)

Each disk block that belong to a internal or leaf node has a BH.
BH size is 24 bytes.
Remember unformatted block doesn't have BH

Block Header (BH)

Each disk block that belong to a internal or leaf node has a BH.
BH size is 24 bytes.
Remember unformatted block doesn't have BH

Block Header Format



Keys (or) Inode Number

1)Used to uniquely identity items & to locate items in the tree & achieve local grouping of items that belongs together.

2) Key contains four objects:

- Directory Id (to group files of same dir)
- Object Id (Item Id)
- Offset within the object.(Item)
- Type
-

3)Offset is used,because an indirect item can at most contain (block size – 48) / 4 pointers to unformatted blocks.
For block size = 4096 bytes,
Pointers to unformatted blocks=(4096-48)/4
=4048/4=1012
So we have 1012 pointers to unformatted blocks. Each pointer points to a disk block of 4KB that results in max.file size = 4048KB

Keys (or) Inode Number(contd)

4)To handle larger files , multiple keys are used.
5)In version 3.6 offset field was increased to bits and type field shrunk to 4 bits previously both has 4 bytes each. So now it allows max file size of 2^60 bytes.
6)Only stat item has offset=0
7)For files (direct /indirect items) & directories always start with an offset 1 ,so that they are sorted behind stat item in leaf nodes.
8)For directory item,offset contains “hash value” and generation number of left most directory header of directory item.

Key (Inode number)Format

Possible types are: 0 – stat , 1 – Indirect , 2 – Direct ,3 – Directory , 15 -Any