Block Group Descriptors

Each block group on the filesystem has one of these descriptors associated with it. As noted in the Layout section above, the group descriptors (if present) are the second item in the block group. The standard configuration is for each block group to contain a full copy of the block group descriptor table unless the sparse_super feature flag is set.

Notice how the group descriptor records the location of both bitmaps and the inode table (i.e. they can float). This means that within a block group, the only data structures with fixed locations are the superblock and the group descriptor table. The flex_bg mechanism uses this property to group several block groups into a flex group and lay out all of the groups' bitmaps and inode tables into one long run in the first group of the flex group.

If the meta_bg feature flag is set, then several block groups are grouped together into a meta group. Note that in the meta_bg case, however, the first and last two block groups within the larger meta group contain only group descriptors for the groups inside the meta group.

flex_bg and meta_bg do not appear to be mutually exclusive features.

In ext2, ext3, and ext4 (when the 64bit feature is not enabled), the block group descriptor was only 32 bytes long and therefore ends at bg_checksum. On an ext4 filesystem with the 64bit feature enabled, the block group descriptor expands to at least the 64 bytes described below; the size is stored in the superblock.

If gdt_csum is set and metadata_csum is not set, the block group checksum is the crc16 of the FS UUID, the group number, and the group descriptor structure. If metadata_csum is set, then the block group checksum is the lower 16 bits of the checksum of the FS UUID, the group number, and the group descriptor structure. Both block and inode bitmap checksums are calculated against the FS UUID, the group number, and the entire bitmap.

The block group descriptor is laid out in `struct ext4_group_desc`.

Offset	Size	Name	Description
0x0	__le32	bg_block_bitmap_lo	Lower 32-bits of location of block bitmap.
0x4	__le32	bg_inode_bitmap_lo	Lower 32-bits of location of inode bitmap.
0x8	__le32	bg_inode_table_lo	Lower 32-bits of location of inode table.
0xC	__le16	bg_free_blocks_count_lo	Lower 16-bits of free block count.
0xE	__le16	bg_free_inodes_count_lo	Lower 16-bits of free inode count.
0x10	__le16	bg_used_dirs_count_lo	Lower 16-bits of directory count.
0x12	__le16	bg_flags	Block group flags. See the bgflags table below.
0x14	__le32	bg_exclude_bitmap_lo	Lower 32-bits of location of snapshot exclusion bitmap.
0x18	__le16	bg_block_bitmap_csum_lo	Lower 16-bits of the block bitmap checksum.
0x1A	__le16	bg_inode_bitmap_csum_lo	Lower 16-bits of the inode bitmap checksum.
0x1C	__le16	bg_itable_unused_lo	Lower 16-bits of unused inode count. If set, we needn't scan past the `(sb.s_inodes_per_group - gdt.bg_itable_unused)` th entry in the inode table for this group.
0x1E	__le16	bg_checksum	Group descriptor checksum; crc16(sb_uuid+group_num+bg_desc) if the RO_COMPAT_GDT_CSUM feature is set, or crc32c(sb_uuid+group_num+bg_desc) & 0xFFFF if the RO_COMPAT_METADATA_CSUM feature is set. The bg_checksum field in bg_desc is skipped when calculating crc16 checksum, and set to zero if crc32c checksum is used.
			These fields only exist if the 64bit feature is enabled and s_desc_size > 32.
0x20	__le32	bg_block_bitmap_hi	Upper 32-bits of location of block bitmap.
0x24	__le32	bg_inode_bitmap_hi	Upper 32-bits of location of inodes bitmap.
0x28	__le32	bg_inode_table_hi	Upper 32-bits of location of inodes table.
0x2C	__le16	bg_free_blocks_count_hi	Upper 16-bits of free block count.
0x2E	__le16	bg_free_inodes_count_hi	Upper 16-bits of free inode count.
0x30	__le16	bg_used_dirs_count_hi	Upper 16-bits of directory count.
0x32	__le16	bg_itable_unused_hi	Upper 16-bits of unused inode count.
0x34	__le32	bg_exclude_bitmap_hi	Upper 32-bits of location of snapshot exclusion bitmap.
0x38	__le16	bg_block_bitmap_csum_hi	Upper 16-bits of the block bitmap checksum.
0x3A	__le16	bg_inode_bitmap_csum_hi	Upper 16-bits of the inode bitmap checksum.
0x3C	__u32	bg_reserved	Padding to 64 bytes.

Block group flags can be any combination of the following:

Value	Description

Value	Description
0x1	inode table and bitmap are not initialized (EXT4_BG_INODE_UNINIT).
0x2	block bitmap is not initialized (EXT4_BG_BLOCK_UNINIT).
0x4	inode table is zeroed (EXT4_BG_INODE_ZEROED).