BRFS Specification

Bruno Filesystem (formerly BOOT-ROOT)

Angel Ruiz Fernandez <arf20>
Bruno Castro García <bruneo32>

October 12, 2023

Abstract

This specification document describes the BRFS filesystem structure used to store data on storage devices. This provides a standard common description of the filesystem for developers to implement freely.

Revision History

Revision	Date	Author(s)	Description
0.1		bruneo32	Created
0.2		bruneo32	Unknown
0.2b		bruneo32, arf20	This document

Contents

1 Introduction

1.1 Scope

This document defines the Bruno Filesystem. As a filesystem it provides a way of structuring data in a block-based (i.e. LBA) storage device. It is meant for embedded systems where a ciomplex filesystem is not needed, this is not a replacement for any modern desktop filesystem such as ext4, because it lacks basic features of journaling. Although BRFS is able to address large volumes, it is not recommended.

1.2 Definitions

Key words will be refferred to with a `monospace font`.

- block: Minimum filesystem unit of data
- inode: File or directory entry

1.3 Advantages and disadvantages

Advantages	**Disadvantages**
TODO when defined	

1.4 Volume layout

- Superblock
- Root inode
- ¡other inodes¿

2 Superblock

The superblock records properties of the enclosed filesystem, such as the block size, pointer size and atribute size. It is 1 block in size. The remaining block will be padded with zeroes.

2.1 Superblock layout

Size (bytes)	**Field (Unsigned integer)**
4	Magic number "BRFS" 0x42524653
1	Block size in LBAs
1	Pointer size in bits, power of 2
1	Attribute size in bytes
b-6	Padding 0x00...

2.2 Theorical limits

Property	**Limit**
Block size	256
Pointer size	256
Attribute size	256
Addresseable blocks	2^{256}
Addresseable LBAs	$256 \cdot 2^{256}$
Absolute maxium capacity (512-byte LBA)	$512 \cdot 256 \cdot 2^{256}$ There is no prefix large enough

The maximum capacity of the filesystem is calculated as follows

$$C = L \cdot B \cdot 2^p \tag{1}$$

Where p is pointer size, B is block size and L is LBA size.

Some examples of reasonable configurations (assuming 512-byte LBA) are $p = 32$, $B = 8$, which gives 16 TiB capacity; or for more efficient storage, $p = 64$, $B = 1$: 8 YiB; for embedded systems perhaps only a $p = 16$ $B = 1$ is needed, for 32 MiB.