BRFS Specification

Bruno Filesystem (formerly BOOT-ROOT)

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

October 29, 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
0.3		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
- unspecified: May be implementation dependent

1.3 Advantages and disadvantages

Advantages	**Disadvantages**
TODO when defined	

1.4 Volume layout

Superblock
Root directory
<other files>
Free space

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**	**Value**
4	Magic number	"BRFS" 0x42524653
1	Block size in power of 2	bytes = $2^{8+\text{this}}$
1	Pointer size in bytes	2, 4, or 8
Pointer	Total filesystem size in blocks	
Pointer	Free blocks count	
Pointer	First free block	
Directory entry	Root directory entry	
	Padding...	0x00

2.2 Theorical limits

Property	**Limit**
Block size	256
Pointer size	64
Attribute size	256
Addresseable blocks	2^{64}
Addresseable LBAs	$256 \cdot 2^{64}$
Absolute maxium capacity (512-byte LBA)	$512 \cdot 256 \cdot 2^{64} \approx 2$ YiB

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 ZiB; for embedded systems perhaps only a $p = 16$ $B = 1$ is needed, for 32 MiB.

2.3 Root directory entry

It is a standalone directory entry (see section 3.2.1) that refers to the root directory. Here is stored the size of the root directory, attributes, beginning. The fiename of the root directory is "/".

3 File

BRFS is a file based filesystem. Regular files and directories are both files.

3.1 Next block pointer

In the end of each file's block, lies a pointer to the next block of the file. This pointer is a linear offset of blocks.

Data
[Padding]
Next block pointer

Pointer number 0 is reserved to denote EOF, and pointer 1 refers to next block. Pointer space starts with 2, the block after the first block of the root directory, which coincides with the global block offset.

	Block
0	Superblock
1	Root directory
2	First file
3	...

3.1.1 Freed blocks

When deleting or shrinking files, the next block pointer must be marked with all 1s, or -1 in 2-complement, and then in the beggining of the block

3.2 Directory

A directory is a special file that holds file entries one after another. There is no limit on the number of entries. The directory ends with a null character 0x00, following the string terminator of the last entry's file name.

file0
...
filen
null terminator 0x00
[Padding]
Next block pointer

3.2.1 Directory entry

Describes file entry on directory. Its size is variable, and the filename null-terminator also serves as entry terminator.

Type (Size)	**Field**
8	File size
22	Attributes
Pointer	First block
unspecified	File name (null-terminated string)

3.2.2 Attributes

Following the POSIX.1-2017 standard, and inspired in some linux ext4 attributes. It takes 22 bytes. See linux kernel and man7.

Type (size)	**Name**	**Description**
uint16	mode	File type and mode
uint32	uid	User ID of owner
unit32	gid	Group ID of owner
uint32	crtime	Creation time
uint32	atime	Last access time
uint32	mtime	Last modification time