

オペレーティングシステム

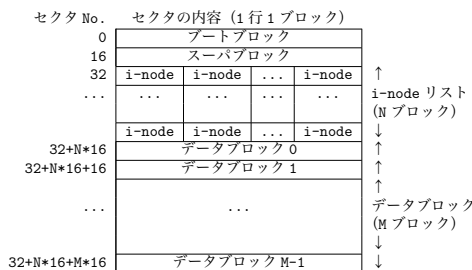
第16章 UNIX ファイルシステム

<https://github.com/tctsigemura/OSTextBook>

UNIX ファイルシステム

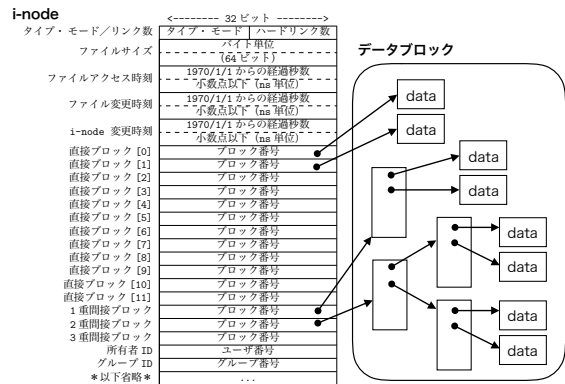
- UFS (UNIX File System)
- 1979年にリリースされた Version 7 UNIX のファイルシステムとそれを改良した多くのファイルシステムを UFS と呼ぶ。
- 第14章で「UNIX の場合」=「UFS の場合」だった。
 - 木構造のディレクトリシステム
 - ハードリンク、シンボリックリンク
 - ボリュームのマウント
 - ファイルの属性 (所有者: マルチユーザを意識)
 - ファイル操作のシステムコール, ...
- 多くのファイルシステムが UFS に準拠している。
 - Windows の NTFS
 - macOS の HFS+, APFS
 - Linux の ext3, ext4

ボリウム内部の配置



- セクタ 512B, ブロック 16セクタ
- i-nodeがファイルを表現する (i-node = ファイル?)
- i-nodeのサイズ= 128B

i-node (index node) (1)



i-node (index node) (2)

- タイプ・モード
 - `type/sst/rwxrwxrwx` の 16 ビット
 - `type` : ファイル型 (通常, ディレクトリ, シンボリックリンク等)
 - `ss` (`Set-uid`, `Set-gid`) : プログラムファイルを所有者の権限で実行
 - `t` : UNIX のバージョンによりさまざま
 - `rwxrwxrwx` : ファイルの保護モード
 - リンク数: ハードリンク数
 - ファイルサイズ: バイト単位
 - 3つの時刻
 - 最終アクセス時刻
 - 変更時刻
 - `i-node` 変更時刻
- 時刻は, 次の 2 つの 32 ビット整数で表現する.
- 1970 年 1 月 1 日午前 0 時 (UTC) からの経過秒数
 - 秒の小数点以下をナノ秒単位で表現

i-node (index node) (3)

- 直接ブロック
 - ファイル本体のデータ
 - $8KiB \times 12 = 96KiB$ まで表現（ブロックサイズ $8KiB$ 時）
 - ファイルの第 0 バイト～第 $96Ki - 1$ バイトまでを管理
- 1 重間接ブロック
 - 直接ブロックだけでは表現できない大きなファイルに使用
 - 1 重間接ブロックを用いてデータブロックの番号を格納
 - 2 重間接ブロックに $8KiB \div 4B = 2Ki$ 個のブロック番号を格納
 - $2Ki$ 個のデータブロックは $8KiB \times 2Ki = 16MiB$ のデータを格納
 - 第 $96Ki$ バイト～第 $96KiB + 16MiB - 1$ バイトの範囲を受け持つ
- 2 重間接ブロック
 - 1 重間接ブロックでも表現できない大きなファイルに使用
 - 2 重間接ブロックは 1 重間接ブロックの番号を格納
 - 2 重間接ブロックに $8KiB \div 4B = 2Ki$ 個のブロック番号を格納
 - $2Ki$ 個の 1 重間接ブロックは $16MiB \times 2Ki = 32GiB$ のデータを格納

i-node (index node) (4)

- 3重間接ブロック
 - 2重間接ブロックを2Ki個管理できる
 - 2Ki個の2重間接ブロックは $32\text{GiB} \times 2\text{Ki} = 64\text{T/B}$ のデータを格納
- 所有者 ID: ファイル所有者のユーザ番号 (マルチユーザ)
- グループ ID: ファイルのグループ番号

インデクス方式 i-node のような構造を使う方式のこと。
高速なランダムアクセスができる。

スパースファイル 途中に穴の空いたファイルのこと。
インデクス方式はスパースファイルを表現できる。

UNIX ファイルシステム

7 / 13

ディレクトリファイル

- ファイルの一種である。→ i-node により表現
- ファイルの型 (type) がディレクトリを表す値のもの
- ディレクトリファイルはファイル名と i-node の対応表
- 対応表の1行がディレクトリエントリ

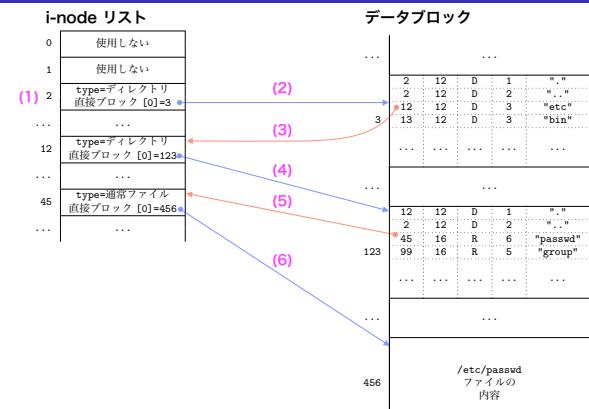
32bit	16bit	8bit	8bit	l_2 バイト	詰め物
i-node 番号	l_1	型	l_2	ファイル名	$\backslash 0 \dots \backslash 0$
←-----		l_1 バイト		----->	

- i-node 番号: ファイルの i-node 番号
- l_1 : ディレクトリエントリの大きさ (4の倍数バイト)
- 型: ファイルの型 (抹消されたディレクトリエントリの表現)
- l_2 : ファイル名のバイト数
- ファイル名: 255 文字以内
- 詰め物: エントリが4の倍数バイトになるように

UNIX ファイルシステム

8 / 13

パス名と i-node の対応付け (/etc/passwd を解析)



UNIX ファイルシステム

9 / 13

パス名と i-node の対応付け (/etc/passwd を解析)

- (1) 絶対パスなのでルートディレクトリから探索を開始する。
ルートディレクトリの i-node 番号は必ず2と決められている。
- (2) ルートディレクトリの i-node から、データブロック3にルートディレクトリの内容が格納されていることが分かる。
- (3) データブロック3の3番目のエントリから、etcが12番の i-node に対応すると分かる。
- (4) 12番目の i-node から etc はディレクトリファイルであること、内容がデータブロック123に格納されていることが分かる。
- (5) データブロック123の3番目のエントリから、passwdが45番の i-node に対応することが分かる。
- (6) 45番目の i-node から passwd は普通のファイルであること、ファイルの内容がデータブロック456に格納されていることが分かる。

UNIX ファイルシステム

10 / 13

練習問題 (1)

1. 次の言葉の意味を説明しなさい。

- ブートブロック
- スーパーブロック
- i-node
- i-node リスト
- インデクス方式
- スパースファイル
- ディレクトリファイル
- ディレクトリエントリ
- 直接ブロック
- 間接ブロック

UNIX ファイルシステム

11 / 13

練習問題 (2)

2. ブロックサイズが8セクタ (4KiB) の場合、直接ブロックだけ用いて表現できるファイルの最大サイズを答えなさい。
3. ブロックサイズが8セクタ (4KiB) の場合、1重間接ブロックを用いることによって、直接ブロックだけの場合と比較して、ファイルサイズを最大でどれだけ大きくできるか答えなさい。
4. ブロックサイズが8セクタ (4KiB) の場合、2重間接ブロックを用いることによって、直接ブロックと1重間接ブロックだけ使用する場合と比較して、ファイルサイズを最大でどれだけ大きくできるか答えなさい。

UNIX ファイルシステム

12 / 13

練習問題 (3)

5. 4 ページの例がスパースファイルを表現しているとする。また、ブロックサイズ等は「ボリューム内部の配置」で示したものと同一とする。次のアドレスはデータブロックが割り当てられているか答えなさい。
- (a) 第 0x00000000 バイト
 - (b) 第 0x00001000 バイト
 - (c) 第 0x00010000 バイト
 - (d) 第 0x00100000 バイト
 - (e) 第 0x01000000 バイト
 - (f) 第 0x10000000 バイト