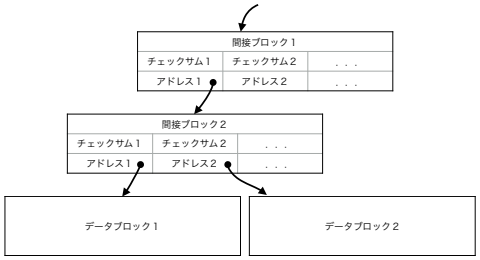
オペレーティングシステム
第１７章 ZFS

https://github.com/tctsigemura/OSTextBook

ZFS の特徴

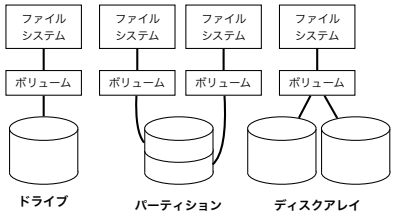
- 2005 年にサン・マイクロが OpenSolaris に実装して公開
- オープンソースで開発が続いている（FreeBSD 等でも使用可）
- 大きな主記憶，高速マルチプロセッサが前提（8GiB，64bit CPU）
- *COW*（*Copy On Write*）でデバイスに書き込む．
  （デバイスのブロックを上書きしないので前状態を維持）
- Uberblock を書き込みと同時に新しい状態になる．
  （Uberblock はファイルシステム管理データの根）
- チェックサムにより高い信頼性を確保（次ページ）
- 一瞬で**スナップショット**や**クローン**を作成
  （COW の手法を使用し違いの出たブロックだけコピーし変更）
- ボリュームの代わりにストレージプールを使用（後のページ）
- ファイルサイズ等の制約が事実上無い（*Zetta* $= 2^{70}$ ）
  （ファイルサイズ $2^{64}B$，ストレージプールサイズ $2^{70}B$）
- ミラーや RAID-Z 等が装備され信頼性・可用性が向上
- データ圧縮，重複除去機能があり（ディスク I/O を少なくする）

全ブロックにわたるチェックサム



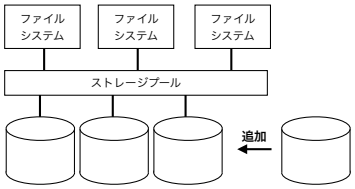
- ブロックポインタがチェックサムを持つ．
- メタデータだけでなく普通のデータにもチェックサムあり．
- チェックサムの不整合が見つかった場合，データの２重化（ミラー）がされていれば，自動的にミラーからデータを修復する．

従来の方式のボリューム



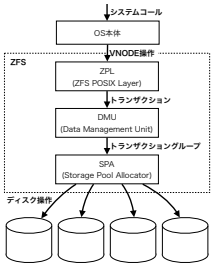
- ファイルシステムの初期化以前にボリュームを決定し，
- 後でサイズの変更などはできない．

ZFS のストレージプール



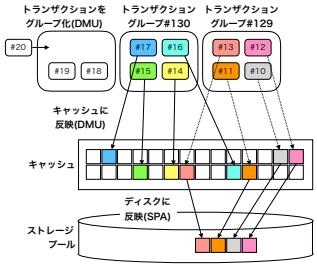
- ストレージプールは沢山のデバイスを収容する．
- ファイルシステムからの要求に応じてブロックを割り付ける．
- C 言語プログラムの `malloc()` や `free()` に似ている．
- ストレージプールに後でデバイスを追加することも可能．

ZFS のソフトウェア構成（１）



1. システムコールは，OS カーネル本体が VNODE 操作に変換する．
2. ZPL は VNODE 操作を ZFS のトランザクションに変換する．
3. DMU は複数トランザクションを**トランザショングループ**にする．
4. SPA は，DMU がトランザクショングループをキャッシュに書込み終わると，キャッシュの内容をデバイスに反映させる．

ZFS のソフトウェア構成（2）

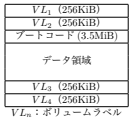


3. DMU は複数トランザクションを**トランザショングループ**にする．
4. SPA は，DMU がトランザクショングループをキャッシュに書込み終わると，キャッシュの内容をデバイスに反映させる．（バースト）

ストレージプールの構造（概要）

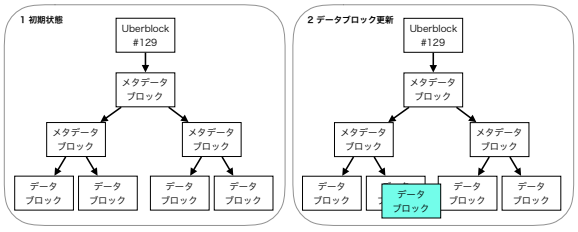
ボリュームラベル

デバイス内部の構造



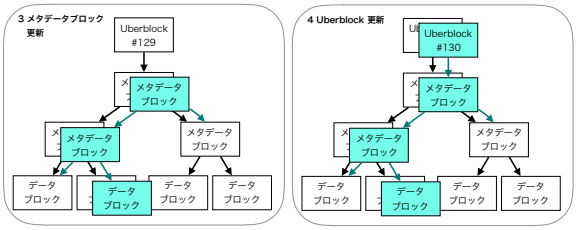
- デバイス（ディスク）の 4 箇所に同じボリュームラベルを書く．
- ボリュームラベルには 128 個の Uberblock を格納できる．
- Uberblock はトランザクショングループ番号を含んでいる．
- Uberblock はトランザクショングループ番号を 128 で割った余りの位置に書く．

ストレージプールの更新（1）



1. Uberblock を起点とする木構造でブロックが記録されている．
2. 変更するには，新しいブロックを確保し内容を書き込む（COW）．

ストレージプールの更新（2）



3. メタデータブロックも COW で更新する．
4. Uberblock を新しい領域に書き込む．
   （トランザクショングループ番号が最新の Uberblock が有効）
   （古い世代のブロックは解放され，再利用される．）

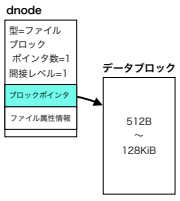
ブロックポインタ

図中でブロックを指していた**矢印**を表現するデータ構造を**ブロックポインタ**と呼ぶ．ブロックポインタはデータ多重化のために最大 3 組のアドレスを記録できる．ブロックポインタの内容は以下の通り．

- **サイズ**：ブロックの大きさに関する情報
- **チェックサム**（64 ビット）：ブロックのチェックサム（最大 3 個）
- **ブロックのアドレス**：ブロックのストレージプール内での格納位置に関する情報（最大 3 個）（デバイス，デバイス内アドレス）
- **タイムスタンプ**：ブロックを作成したトランザクショングループの番号（ブロックが削除される時にスナップショットと比較）
- **その他**：チェックサム計算に使用するアルゴリズムの種類，データ圧縮に使用するアルゴリズムの種類，圧縮後のサイズなど...

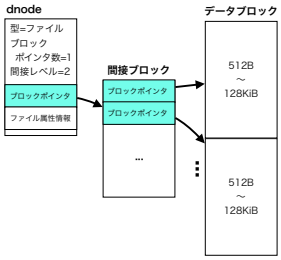
Dnode（基本）

あらゆるオブジェクトを表現する 512 バイトのデータ構造である．USF の `i-node` に似ているが，ファイル，ディレクトリだけでなく，ファイルシステム，スナップショット，クローンなども表現する．



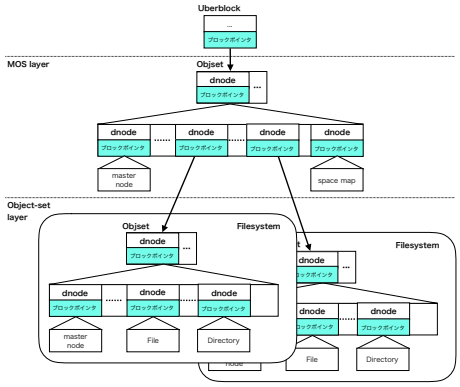
- dnode は三つ以内のブロックポインタを格納することができる．
- dnode は表現するオブジェクトに応じたデータを格納する領域を持っている．（この領域はブロックポインタと共用になっている）

Dnode（データが大きい場合）

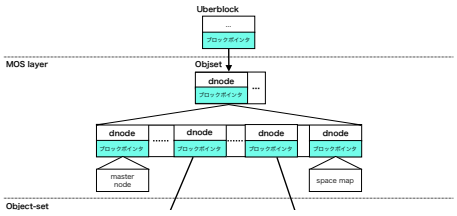


- データの大きさが 128KiB 以内の場合は直接参照（前ページの図）
- 大きさが 128KiB を超える場合は間接ブロック（上の図))
- 128KiB の間接ブロックはブロックポインタを最大 1Ki 個格納できる．
- $128KiB \times 1Ki = 128MiB$ より大きなデーを表現する時は，多重の間接ブロック（最大 6 レベル）を用いる．(2^{64} バイト以上)

ストレージプールの全体像

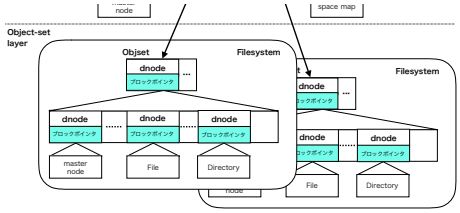


MOS（Meta Object Set）layer



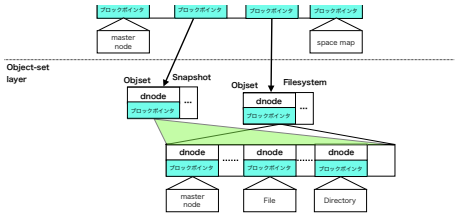
- Objset は dnode の配列を管理する．
- 配列には，ストレージプールプール全体の管理に関する情報や，ファイルシステムやスナップショット等の一覧が格納される．
- master node はストレージプールのコンフィグ，プロパティなど．
- space map はストレージプール内のブロックの割付を管理する．

ストレージプールの全体像（Object-set layer）



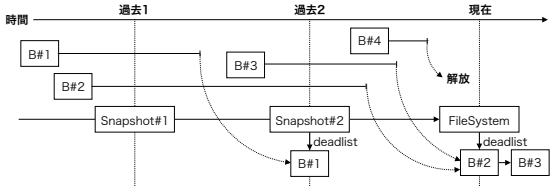
- ファイルシステムは Objset の dnode 配列で表現する．
- dnode リストが UFS の `i-node` リストに相当する．
- master node は root ディレクトリの dnode 番号などの情報．
- 図は，通常ファイルの例とディレクトリファイルの例

スナップショットの作成



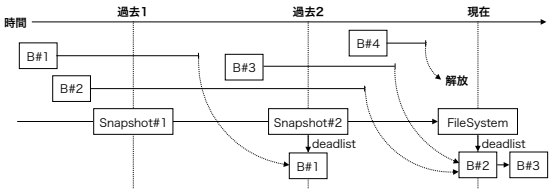
- ファイルシステムを表す Objset のコピーを作る．
- MOS layer の dnode 配列に登録する．
- スナップショットは一瞬で作成できる．
- 変更がないデータブロックは共用する．
- 内容が書き換わると COW で必要最小限のブロックのコピーを作る．

スナップショットと deadlist



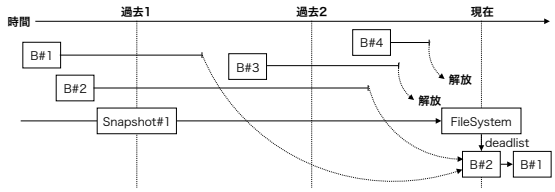
- ファイルシステム（`FileSystem`）は時間とともに変化する．
- ある瞬間の状態を保存したものがスナップショット．
- `B#N` と続く線はブロックとブロックの寿命を表す．
- `Snapshot#N` はスナップショットを表す．
- `deadlist` は使用されなくなったが解放できないブロックのリスト

ブロックの解放（1）



- B#1 は Snapshot#1 で使用されているので解放できない．
- B#2 と B#3 もスナップショットで使用されているので解放できない．
- B#4 はスナップショットで使用されていないので解放できる．
- 解放できないブロックは deadlist に格納される．

ブロックの解放（2）



- Snapshot#2 を削除した状態．
- B#1 と B#2 は Snapshot#1 で使用されているので解放できない．
- B#3 は Snapshot#2 が削除されたので解放できる．
- B#1 と B#2 は FileSystem の deadlist に格納される．

まとめ

ZFS は大きな主記憶と高速な CPU を前提に設計されている．
以下の特徴を持つ．

- COW（Copy On Write）を用い既存のブロックを上書きしない．
- COW のお蔭でシステムのクラッシュでも壊れない構造を実現．
- デバイス上の全てのデータについてチェックサムを持つ．
- スナップショットやクローンを一瞬で作ることができる．
- ボリュームの代わりにストレージプールを使用する．
- ストレージプールに後で新しいデバイスを追加可能．

練習問題（1）

次の言葉の意味を説明しなさい．分からない言葉は調べなさい．

- COW（Copy On Write）
- メタデータ
- チェックサム
- スナップショット
- クローン
- ボリューム
- ストレージプール
- ZPL（ZFS POSIX Layer）
- DMU（Data Management Unit）
- SPA（Storage Pool Allocator）
- VNODE

練習問題（2）

- Uberblock
- ブロックポインタ
- Dnode
- MOS（Meta Object Set）layer
- Object-set layer
- space map
- master node

練習問題（3）

- トランザクショングループ番号は 64 ビットです．毎秒 100 トランザクショングループを処理したとして，トランザクショングループ番号がオーバーフローするまでに約何年かかるか計算しなさい．
- ZFS で使用できるチェックサム計算アルゴリズムについて調べなさい．
- ファイルを表現する dnode が間接レベル 2 の時，最大のファイルサイズは何バイトになるか計算しなさい．
- ブロックにリンクカウントを設け，スナップショットからの参照数を管理することで，ブロックの解放を判断するアイデアの利点と問題点を挙げなさい．