ZFS 分層架構設計

Table of Contents

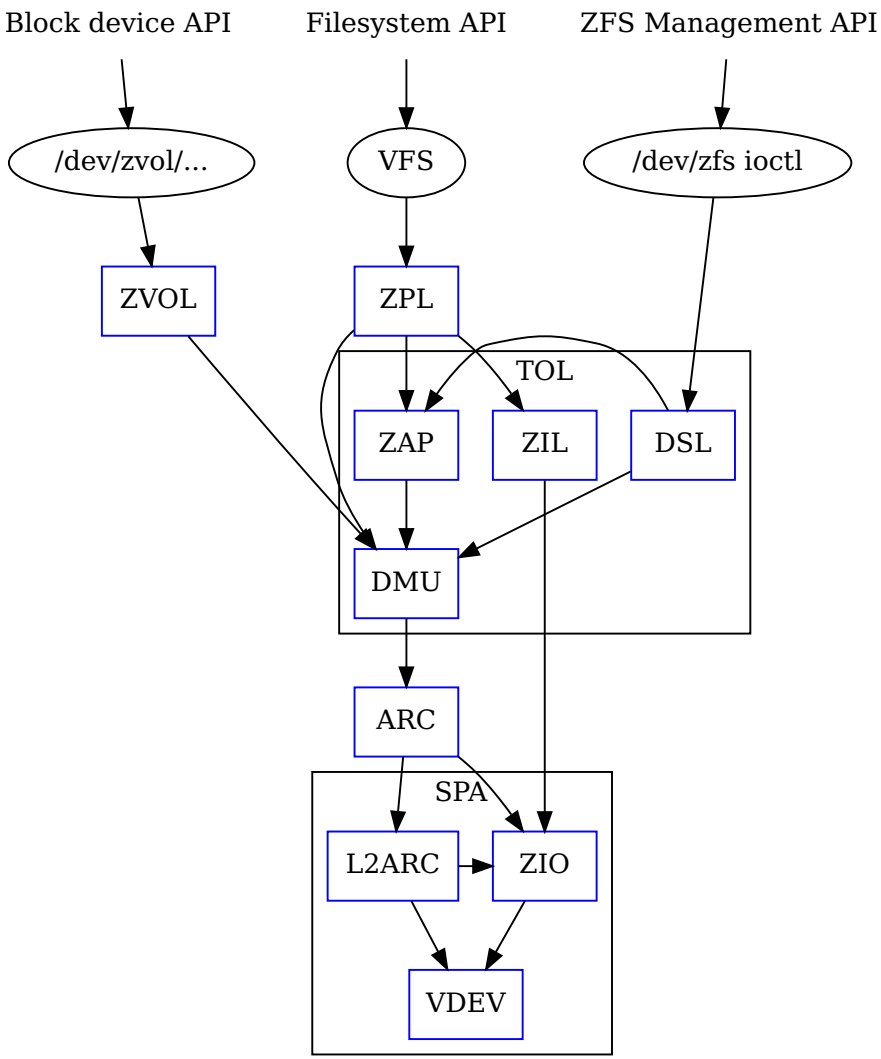
Contents

- ZAP
- ZIL
- DSL
- DMU
- ARC
- SPA

ZFS 在設計之初背負了重構 Solaris 諸多內核子系統的重任，從而不同於 Linux 的文件系統 只負責文件系統的功能而把其餘功能（比如內存髒頁管理，IO調度）交給內核更底層的子系統， ZFS 的整體設計更層次化並更獨立，很多部分可能和 Linux 內核已有的子系統有功能重疊。 而本文想講的只是 ZFS 中與快照相關的一些部分，於是先從 ZFS 的整體設計上說一下和快照相關的概念位於 ZFS 設計中的什麼位置。

子系統整體架構

首先 ZFS 整體架構如下圖，其中圓圈是 ZFS 給內核層的外部接口，方框是 ZFS 內部子系統：


Block device API
Filesystem API
ZFS Management API
/dev/zvol/...
VFS
/dev/zfs ioctl
ZVOL
ZPL
TOL
ZAP
ZIL
DSL
DMU
ARC
SPA
L2ARC
ZIO
VDEV

VDEV

Virtual DEVice

作用相當於 Linux Device Mapper 層或者 FreeBSD GEOM 層，提供 Stripe/Mirror/RAIDZ 之類的多設備存儲池管理和抽象。 ZFS 中的 vdev 形成一個樹狀結構，在樹的底層是從內核提供的物理設備， 其上是虛擬的塊設備。每個虛擬塊設備對上對下都是塊設備接口。

ZIO

ZFS I/O，作用相當於內核的 IO scheduler 。

ZPL

ZFS Posix Layer，提供符合 POSIX 文件系統的語義，也就是包括文件、目錄這些抽象以及 inode 屬性、權限那些，對一個普通 FS 而言用戶直接接觸的部分。

ZVOL

ZFS VOLume

有點像 loopback block device，暴露一個塊設備的接口，其上可以創建別的 FS。對 ZFS 而言實現 ZVOL 的意義在於它是比文件更簡單的接口所以一開始先實現的它，而且 早期 Solaris 沒有 sparse 文件的時候可以用它模擬很大的塊設備，測試 Solaris UFS 對 TB 級存儲的支持情況。

TOL

Transactional Object Layer

在數據塊的基礎上提供一個事務性的對象語義層。每個對象用多個數據塊存儲，每個數據塊大概是 4K~128K 這樣的數量級。

ZAP

ZFS Attribute Processor，在「對象」基礎上提供緊湊的 name/value 映射，從而文件夾內容、文件屬性之類的都是基於 ZAP 。

ZIL

ZFS Intent Log，記錄兩次完整事務語義提交之間的 log，用來加速實現 fsync 之類的保證。

DSL

Dataset and Snapshot Layer，數據集和快照層，這是本文的重點。

DMU

Data Management Unit，在塊的基礎上提供「對象」的抽象。每個「對象」可以是一個文件，或者是別的 ZFS 內部需要記錄的東西。

ARC

Adaptive Replacement Cache，作用相當於 pagecache 。

SPA

Storage Pool Allocator，從內核的多個塊設備中抽象出存儲池。