

Work in Progress Report

UFLFS: A Log-Structured File System Supporting Snapshots

A. Diot, D.R. Karrels, A.R. Katebi, P. Padala, J.N. Wilson
University of Florida

The technology trends that make log-structured file systems viable are continuing. Increases in memory size and speed have led to the ability to cache more data, whereas the huge increases in disk capacity have led to disks being used less and less like memory. Log structured file systems, having the ability to write large amount of data at once, can exploit these continuing trends. Yet the number of viable log structured file system implementations has remained small. Our project is aimed at developing an open-source log-structured file system that can serve as the basis for a network attached storage device. The uffs group consists of a handful of graduate students(Karrels, Katebi, Padala), an undergraduate(Diot), and one faculty member(Wilson).

One of the keys to the success of the WAFL file system and the network attached storage products built upon it is the efficient snapshotting mechanism that it supports. The fundamental properties of WAFL that support snapshotting are i) its representation of metadata (such as the inode map) in files, ii) its write-once policy, and iii) its use of a vectorized block-map. The first two of these mechanisms are shared by log-structured file systems, but the typical log-structured file system has no block-map whatsoever.

Providing an efficient snapshotting system, however, breaks one of the fundamental assumptions of log-structured file systems, namely, that any block is accessible through at most one inode accessible in the file system. This policy (that only one inode owns each block) is not necessary for log construction, but it is critical for file system cleaning. Our current activities are aimed at evaluating several different strategies for relaxing this policy and mechanisms for cleaning in the face of these relaxations.

At the time of the conference, we will report on the state of our implementation and provide results of simulation studies showing the relative number of read/write operations necessary to perform cleaning on a given trace both with and without snapshotting using several snapshot strategies.