Diplomarbeitspräsentationen der Fakultät für Informatik

FAKULTÄT FÜR !NFORMATIK

Diplomstudium:
Informatik

LLFS, A Copy-On-Write File System For Linux

Rastislav Levrinc

Technische Universität Wien
Institut für Computersprachen
Arbeitsbereich: Programmiersprachen und Übersetzer
Betreuer: Ao. Univ.-Prof. Anton Ertl

Why Yet Another Linux File System?

There are many Linux file systems to choose from: Ext2, Ext3, Ext4, Minix, ReiserFS, XFS, JFS...
They have been improved, rethought, rewritten and rewritten again. What can they be possibly missing? This thesis gives an answer.

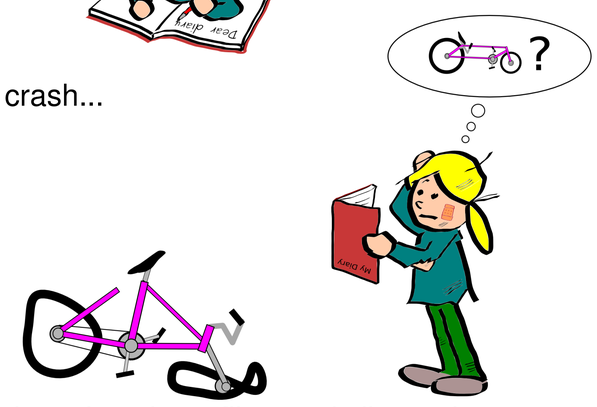
File System Requirements

There are many requirements for file systems: good performance, scalability, portability, reliability, little fragmentation, **fast crash recovery, data consistency**...
The most popular approaches to meet all the requirements are journaling file systems and log-structured file systems. Let's have a look how do they deal with fast crash recovery and data consistency.

Journaling File Systems

Journaling file systems keep journal to record changes that user made to files and directories and will be written soon to the disk.

Now imagine a system crash...



The journal is read and partly-written files and directories are recovered. Unfortunately it does not work well all the time. Journaling file systems only log meta-data, which means, you may find your directory structure recovered, but you may loose the whole content of your files and hours of work.

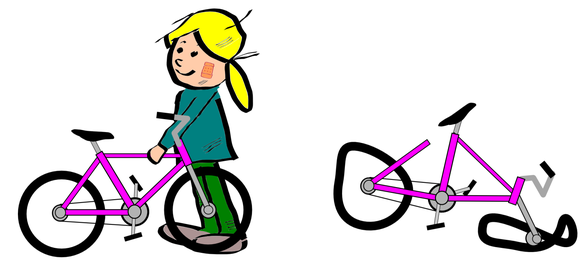
Some journaling file systems can log data too. This solves the problem - or does it?

The journal gets much bigger if data are logged and it makes the file system **slow**. That is why nobody uses it.

Log-Structured File Systems

Log-structured file systems do it differently. They do not overwrite data in place. They treat the hard disk like an endless log and write everything to the end of it. The last consistent state is kept on the disk.

In a case of system crash...



The file system rolls back in time when everything was still all right. Only the data that were written just before the crash are lost.

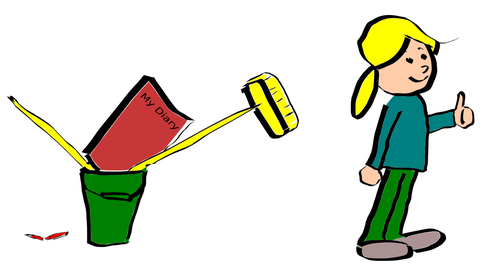
But they do not make endless hard disks. Space must be reclaimed from the beginning of the log. There is a program that takes care of it.

It is called cleaner...

But the cleaner makes log-structured file systems **slow**.

LLFS

LLFS does not write journal nor does it require cleaner...



In spite of that LLFS offers **fast** crash recovery and data consistency.

LLFS does not overwrite data in place, but unlike log-structured file systems it uses bitmaps to keep track of free space and that is **fast** if done right.

The copy-on-write policy opens a door for new features that everybody likes:

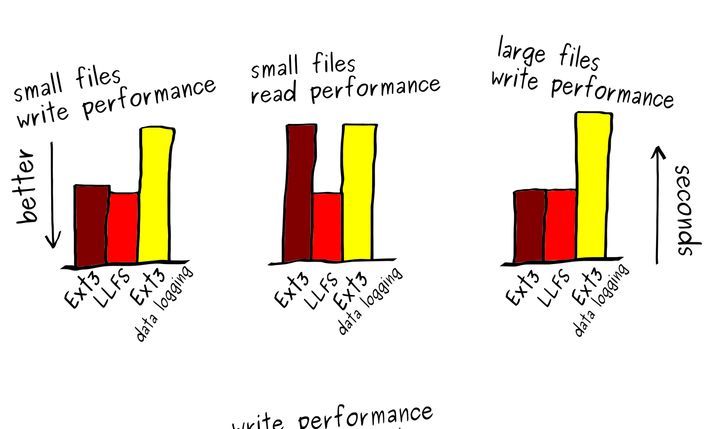
snapshots - a snapshot of all the data as they were in one point in time
clones - writable snapshots and clones of clones

Snapshots and clones are good for consistent backups, trying out different system configurations, multiple undo of mistakenly removed files and so on.

And this is **fast** too.

Results

I have tested the performance, here are some results. See the thesis for more.



My first implementation created for this thesis showed that it is indeed possible to implement a copy-on-write file system that supports clones, snapshots and offers data consistency after system crash. Its performance is on par and in some cases better than journaling file systems that offer less consistency guarantees.

Further Work

However, there is still much to do in LLFS to become an accepted stable Linux file system. The code should be reviewed, improved and optimized. All the temporary solutions should be ironed out. The user tools should be programmed beyond the rudimentary level they are now and so on.

http://stud3.tuwien.ac.at/~e9526925/

LLFS is free software under GPL license and you are invited to help.

Kontakt:
rasto@linbit.com
anton@mips.complang.tuwien.ac.at