Abstract: Improving System Management With ZFS

Brooks Davis <brooks@aero.org>
The Aerospace Corporation

The Zetabyte File System (ZFS) is a modern file system which combines traditional file system features like a POSIX file system interface with RAID and volume management functionality. Features such as snapshot management and file share management are all managed within the ZFS interface. This management interface provides a number of opportunities to simplify system management. In the Technical Computing Services Sub-division of The Aerospace Corporation we are taking advantage these features in a number of different ways. This paper presents some of the more interesting ones.

ZFS Basics

This section will provide a brief overview of ZFS operations and in particular the two ZFS command line tools zpool and zfs. Features we plan to use later will be introduced including making snapshots, cloning snapshots, promoting snapshots, setting attributes including user defined ones on file systems, and using zfs send/receive to transfer snapshots.

Simple ZFS Use

This section will provide a few simple examples of how we use ZFS for home directories and mailing list archive storage. The goal is to provide a little background on ZFS and reinforce the idea that administrators tend to create a LOT of file systems in normal ZFS operations. Some of the issues this can cause will also be covered.

Fixing Mirror Problems With ZFS

One of the more vexing problems when running a mirror server is the issue of partial and thus non-functional mirror updates where available packages do not match the package database. In the past we adopted a strategy where we performed an integrity check after each rsync and restarted immediately if the mirror was inconsistent. This is fairly effective, but during new releases this can leave the repository out of sync for a significant period of time. We will demonstrate a solution to this problem using ZFS clone and promote operations.

Efficient Replication With ZFS Metadata

Replication of snapshots for disaster recover is a common practice. In this section we will present our method of using a combination of ZFS send/receive and ZFS meta data to let us store all configuration data including replica destinations and last snapshots in ZFS attributes. We use this system to replicate projects in Aerosource, our internal Source Forge like infrastructure.

More ZFS Metadata

On Aerosource we also use metadata to store project configuration data in place of storing it in configuration files. This keeps all the data in one place and directly ties project configuration to project storage.

Summary

This paper presents a few ways ZFS features can be used to provide enhanced integration with applications. By taking advantage of these features we have reduced the number of configuration files in our environment and improved over all robustness. We hope these ideas inspire our readers to try integrating advanced ZFS features in their environment and to help grow the set of ZFS patterns and tools available today.