newuserfs: a way to write Linux filesystems in userspace

Martin Pool 9405765 <mbp@samba.org>

October 2, 2001

1 Introduction

newuserfs adds to Linux one of the most interesting features of microkernel-based operating systems such as the GNU Hurd: the ability to mount directories provided by userspace servers. Interesting possibilities include prototyping disk filesystems, versioning filesystems, and transparent access to network or alien filesystems.

newuserfs builds on previous work including Jeremy Fitzhardinge's userfs for Linux 2.0, and podfuk. None of these systems provides a satisfactory solution for the current 2.4 kernel series, and so newuserfs is a genuine improvement in Linux functionality.

2 Approach

newuserfs consists of two interacting parts: a kernel module that implements a filesystem by passing operations out to a userspace server process and a userspace framework for writing servers.

The kernel communicates with the server over a local pipe or socket, passing binary request/response packets as a simple IPC mechanism. The protocol is not machine-independent for reasons of efficiency and simplicity, since servers are intended to always run on the local machine. Since the server only does binary IO and needs no special kernel hooks, filesystems may be implemented in almost any language, including Perl, Python or Java.

There is one userspace process per filesystem, although the process may fork or use threads.

3 Progress

The newuserfs code is available in anonymous CVS under the GNU GPL, and further documentation is available from `http://etc.samba.org/newuserfs/`. The current version can mount as a filesystem, perform basic IPC to the server, and unmount cleanly.

4 Testing

To date only ad-hoc testing has been done. Adhoc testing includes exercising basic operations from shell scripts, and will extend to running IO-intensive operations such as

The code is defensively designed with trace and assertion code to catch internal consistency bugs. Eventually an set of contumacious servers may be written to exercise the kernel module.

If time and resources permit, the module will be tested on multiprocessor and non-Intel systems.

5 Bugs

Ultimate success for newuserfs would be widespread adoption as a standard interface for plugging in filesystems. To reach this, the architecture must be seen as "tasteful": reliable, maintainable, extensible, and in keeping with the Linux architecture.

The greatest risk to successful completion is concurrency bugs: it has not yet been established that deadlocks will not be introduced by filesystem calls blocking on a userspace process that may itself call into the kernel.

6 Discussion

Development is proceeding well. It is anticipated that newuserfs will provide both an impressive demonstration in October, and a useful contribution to the Linux community.