Engineering a NAS box

One common use for Linux these days is as a ’Network Attached Storage’ server. In this talk I will discuss some of the challenges facing NAS server design and how these can be met within the Linux operating system design.

A multi-protocol file server

A NAS box is a multi-protocol file server. Typically the hardware contains some hot-swap disks, one or two gigabit cards and one or two CPUs.

The box might include the following protocols:

- CIFS/SMB
- NFS
- Appletalk
- HTTP
- possibly NCP and NDMP

The box may provide shared network storage for up to several hundred users.

Semantic Conversion

One of the major challenges facing NAS boxes is the problem of ’Semantic Conversion’.

Each of the protocols that the box supports has a quite different set of semantics, and these semantics are in turn quite different from the native semantics of the local operating system (often Linux). This means that for ’correct’ operation the server must map the expected protocol semantics onto the semantics of the local OS, and this mapping must be fast.

Semantic Conversion - part 2

The main semantic conversion problems are:

- locking
- ACLs
- case insensitivity
- short/long names
- delete/rename

There are two broad approaches to each of these - either map the required sematics onto the local OS semantics or add ’parallel semantics’ into the local OS.

Byte range locking

A good example of semantic conversion problems is byte range locking. POSIX byte range locking (in Linux) and CIFS byte range locking are completely different:

- POSIX locks are not stackable, CIFS are
- POSIX locks haved a fixed lock context
- signed/unsigned lock offsets
- POSIX locks are advisory, CIFS mandatory

Think of this sequence of locks

- lock(1,10)
- lock(5,8)
- unlock(1,10)

Is byte 6 now locked?

Byte range locking - part 2

To map from CIFS locking to POSIX locking Samba implements a locking database in user space. That database stores locking records set by CIFS clients. The Samba code then maps the CIFS locks to a ’best approximation’ set of POSIX locks so that other protocols can see the locks set by CIFS clients.

This is not ideal! It is slow and full of race conditions. Can you see why implementing mandatory locks in user space leads to race conditions?

It would be better to add CIFS byte range locking to the kernel.

Case Insensitivity

Applications running on CIFS clients expect filesystems to be case insensitive, whereas Unix systems are case sensitive. How do you provide case insensitive semantics on a case sensitive operating system?

Lets walk through the worst case - proving that a file doesn’t exist. How do you prove that the file /home/test/data/test.dat doesn’t exist?

First you need to search /, then you need to search /home then /home/test and so on. This is very expensive.

Case Insensitivity - part 2

The alternative is to add case insensitive support directly into the kernel. To do this on Linux you need to modify two main kernel subsystems, the low-level filesystem and the dcache.

In the simplest case we need to:

- change the filesystem to use strcasecmp() when looking up names in directories
- change the dcache hash function to be case insensitive
- change the dcache comparison functions to use strcasecmp()

Things soon get a bit more complex.

Case Insensitivity - part 3

The XFS filesystem is a common choice for a NAS box. The on-disk directory format in XFS is a hash system, this also means that we need to change the on-disk format when we change to a case insensitive hash.

For backwards compatibility and seamless conversion we need to mark each directory in XFS as being either case insensitive or case sensitive. The directory hash function is then chosen based on this flag.

Other major problems include:

- case insensitive in what character set?
- per-process case insensitivity

More semantic conversion

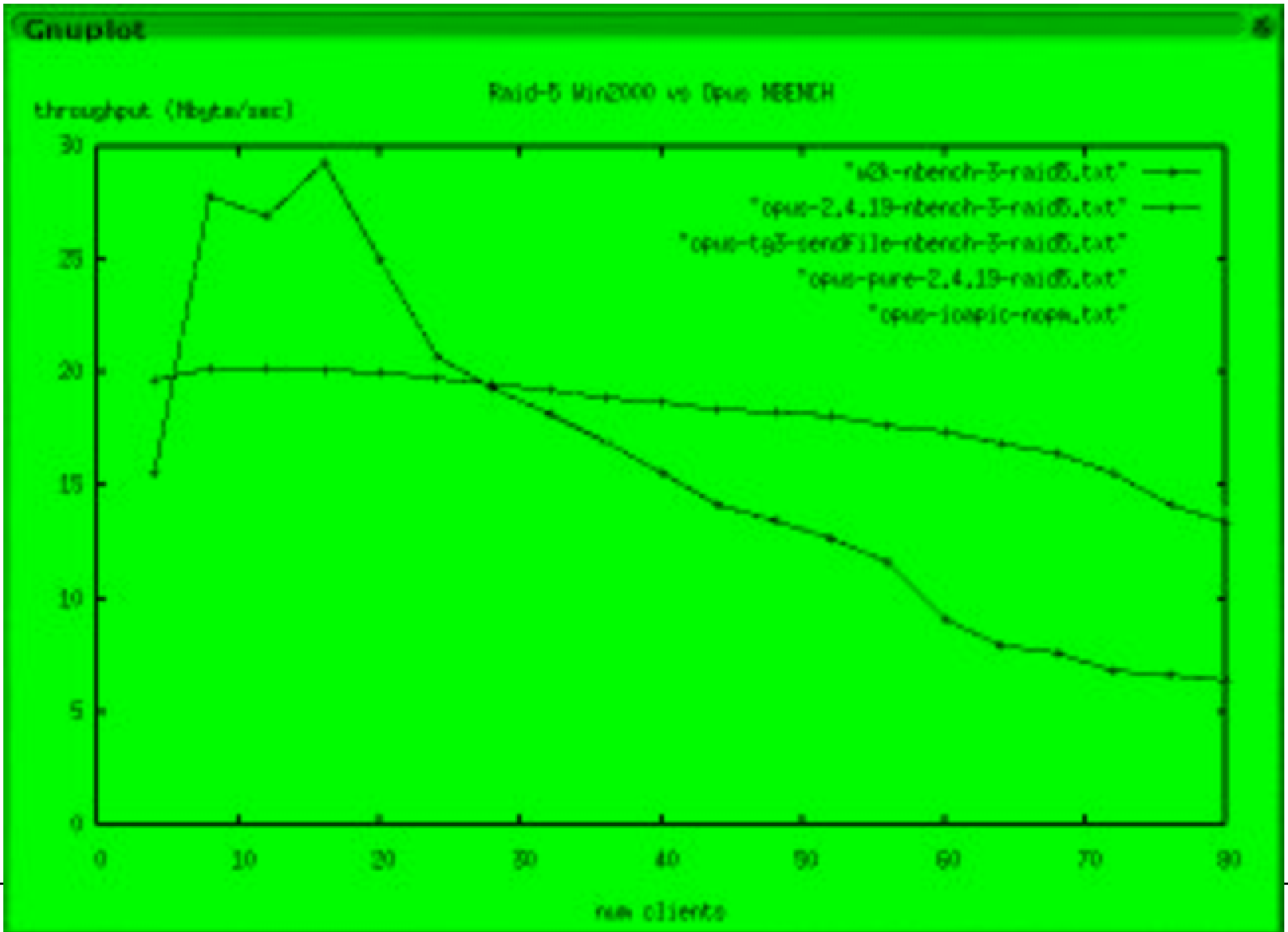
The other major points of semantic mismatch are

- File ACLs (access control lists)
- short/long names
- delete/rename semantics

With each of these we have the choice of semantic mapping or parallel access. Usually parallel access is preferable, but it is often much more complex to implement.

Performance tuning

Good performance is quite critical to a NAS box. Here is a performance comparison using the NBENCH netbench simulator.



Other issues

This talk has only scratched the surface of the issues facing NAS development. Other major issues include:

- hot-swap disk support
- scalable filesystem snapshots
- international character set support
- system management tools
- backup systems