nnodes: an Abstraction Layer for NAS Servers

Sam Falkner
Sun Microsystems, Inc.

Agenda

- Background: original problem
- What are nnodes?
- How are nnodes used?
- What are the benefits?
- Futures?

Background

- Solaris pNFS data server is a different beast...
  - Doesn't use vnodes
- ...but it's still a v4.1 server
  - RPC, XDR, 4.1 protocol, sessions, ...

Background

- We want to maximize shared code
- But we want to allow vastly different implementations underneath

vnodes?

- Yes, vnodes could handle the i/o path
- But in our asymmetric cluster (smart MDS, dumb DS), state handling is also vastly different

Back Ends

	4.1	MDS	Data Server
i/o	VOP_READ	proxy read	dmu_read
namespace	VOP_LOOKUP	VOP_LOOKUP	n/a
state	our in-memory state tables	our in-memory state tables	ask the mds

nnodes: What Are They?

- Like vnodes, but custom made for our purposes
  - We don't "pollute" the vnode namespace with our quirky ops
  - We factor the code optimally for NAS

How Are nnodes Used?

rfs4_op_read()

nnop_check_state()

nnop_io_prep()

nnop_read()

nnop_io_release()

How Are nnodes Used?

- nnop_checkstate() for vnodes (non-DS):
  - check our in-core state tables
- nnop_io_prep() for vnodes:
  - check permission, labels, mandatory locks, attributes up-to-date, reading past EOF? ...
  - i.e. nearly everything before VOP_READ()

How Are nnodes Used?

- nnop_read() for vnodes
  - VOP_READ()
- nnop_io_release() for vnodes
  - drop all resources acquired in nnop_io_prep()

Benefits

- Expected: good code sharing between MDS and data server
- Expected: nnodes are a “filehandle cache”; great place for caching
- Unexpected: good code sharing between v3 and v4!

Futures

- Benchmarks!
- Refactoring exportinfo and the nnode cache
  - Leading to multiple server instances
- Use nnodes in the v4 pseudo namespace
- Leave hooks for other implementations, e.g. UFS and QFS