SPECTRA
SINCE 1979

CAM Target Layer

Ken Merry

Spectra Logic Corporation

Changing The World of Storage

What is CTL?

- SCSI target emulation framework
- Can present a ramdisk, file, or block device as a SCSI target.
- LUNs visible through target-capable CAM SIMs. Only fully supported driver right now is isp(4).
- LUNs also visible through the internal CTL SIM.

CTL History

- Written for Copan Systems, starting in early 2003.
- Originally written for Linux, has some similarities to CAM. Linux version had thousands of lines of CAM code included.
- CTL originally meant “Copan Top Level”, and later “Copan Target Layer”.
- Started shipping in 2005.
- Ported to FreeBSD in 2008, when Copan ported their whole I/O stack to FreeBSD.
- Currently ships in SGI’s ArcFiniti and COPAN 400M/400T products.

How CTL was Open Sourced

- Copan's assets bought by SGI in 2010.
- Spectra Logic made a deal with SGI in 2010.
- Spectra got the source to CTL under a BSD-style license, with the understanding that they would work to get it into FreeBSD.
- CTL committed to FreeBSD/head and stable/9 in early 2012.

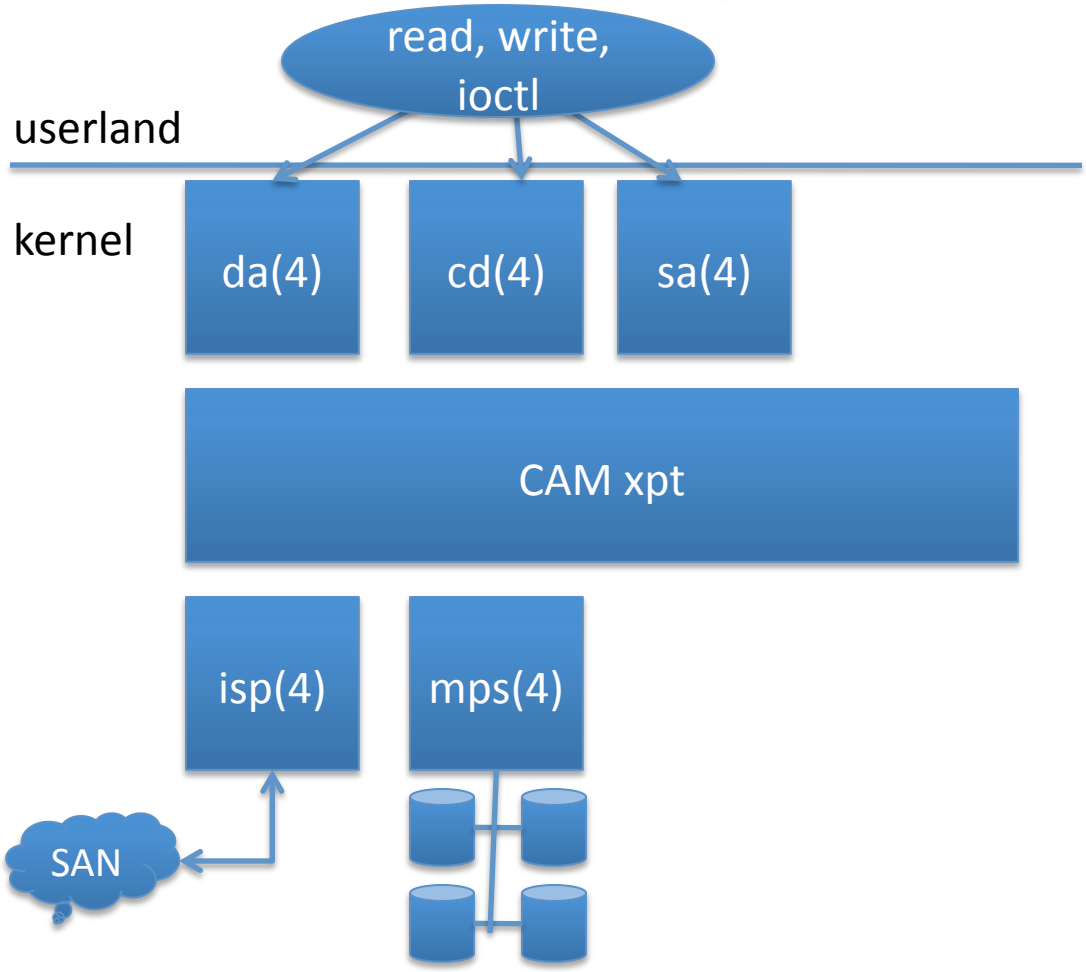
CTL Features

- Disk and processor device emulation
- Tagged queuing
- SCSI task attribute support (ordered, head of queue, simple tags)
- SCSI implicit command ordering
- Full task management support (abort, LUN reset, target reset, bus reset)
- Multiple ports
- Multiple initiators per port
- Multiple LUNs
- Multiple backend types

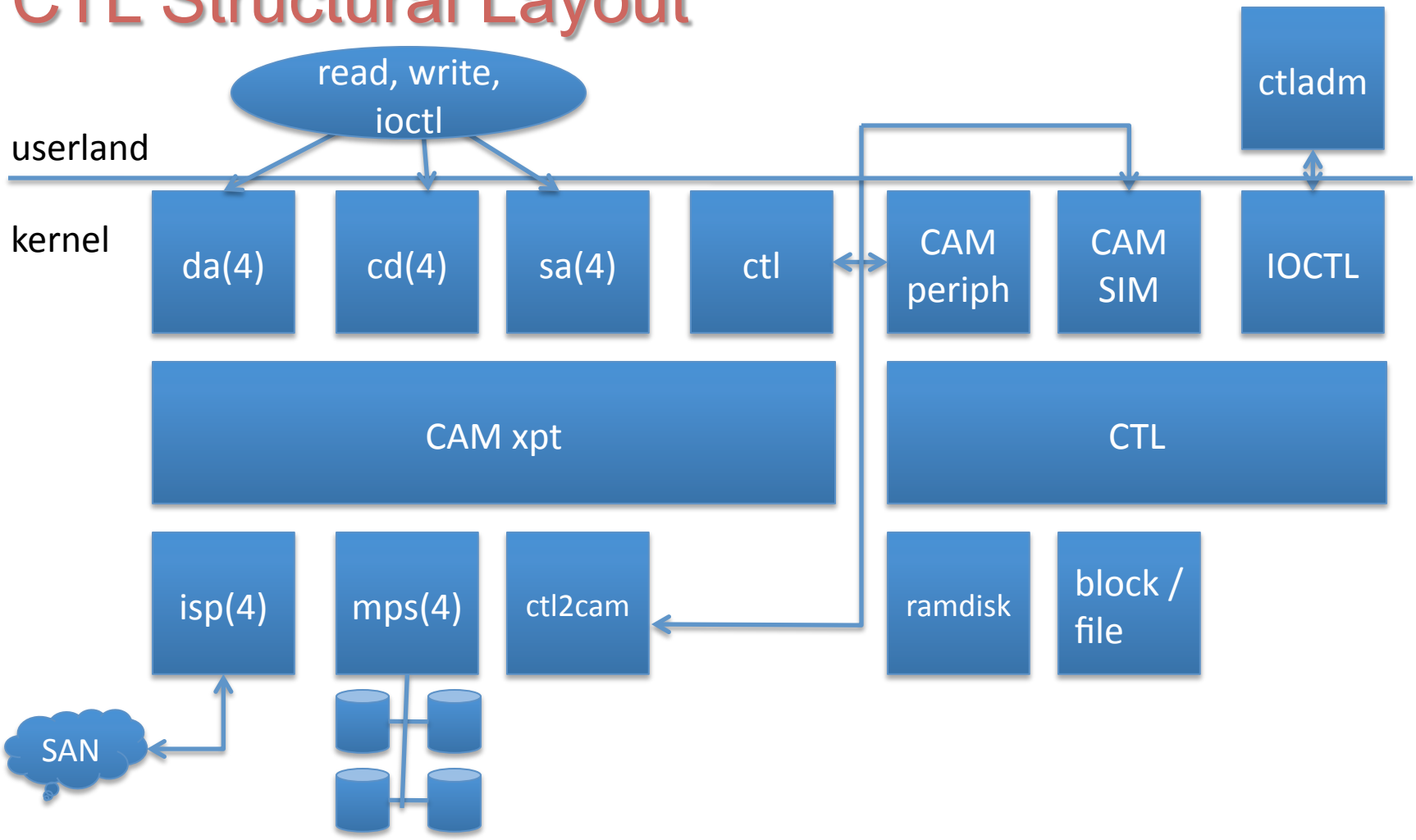
CTL Features (continued)

- Persistent reservations
- Mode sense and select
- All I/O is handled in-kernel, no userland context switches.
- Basic High Availability support stubs. (Needs to be fleshed out.)

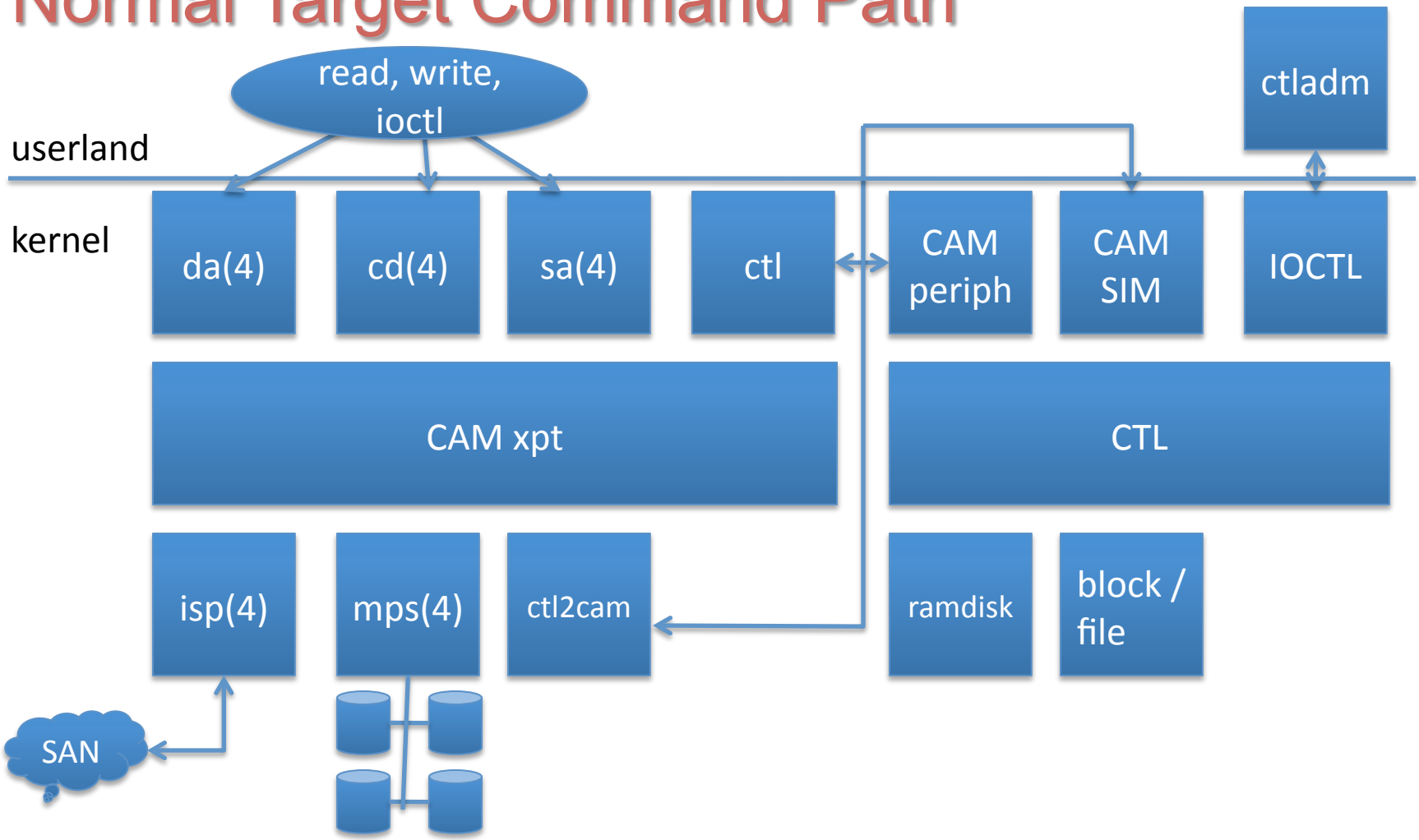
CAM Structural Layout



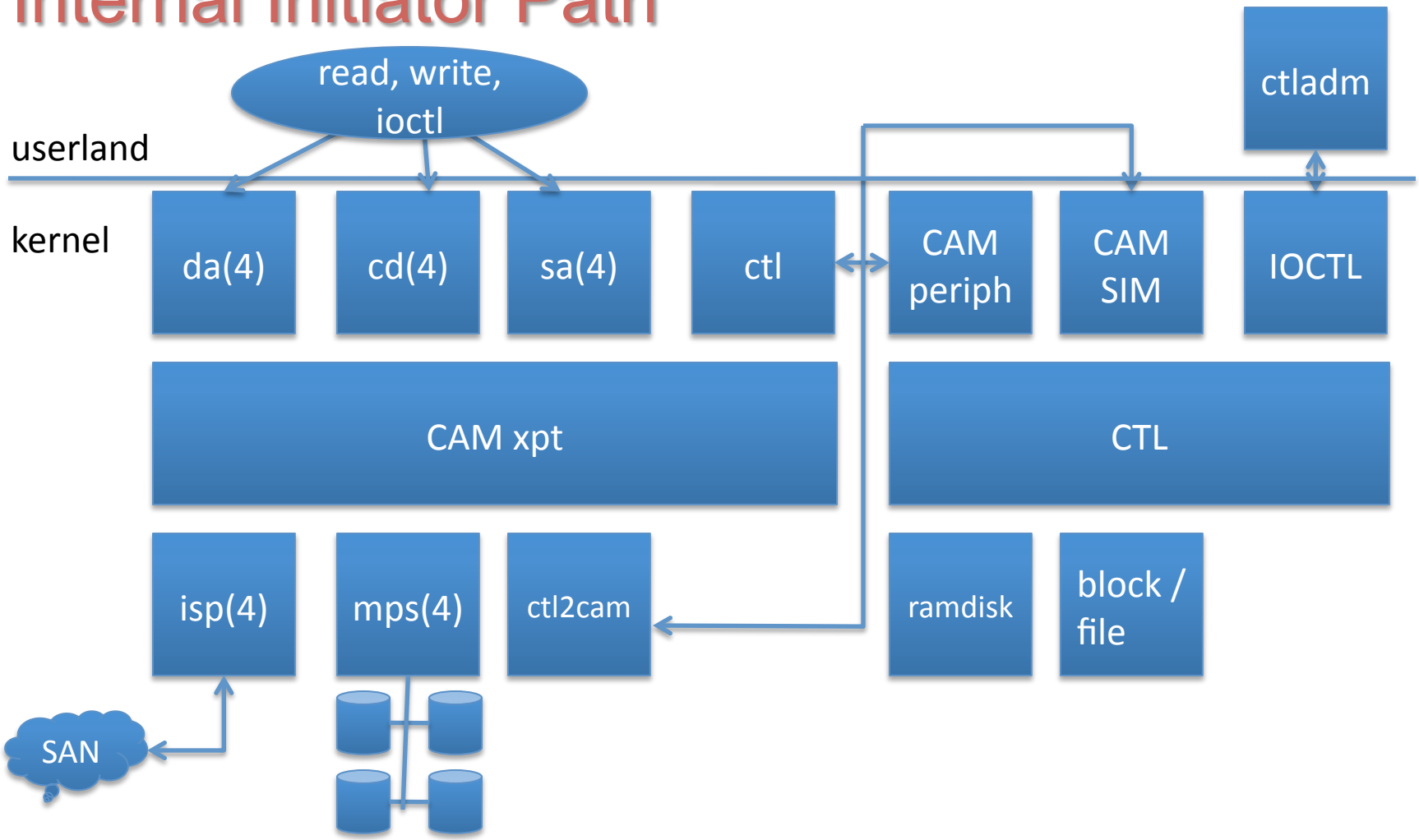
CTL Structural Layout



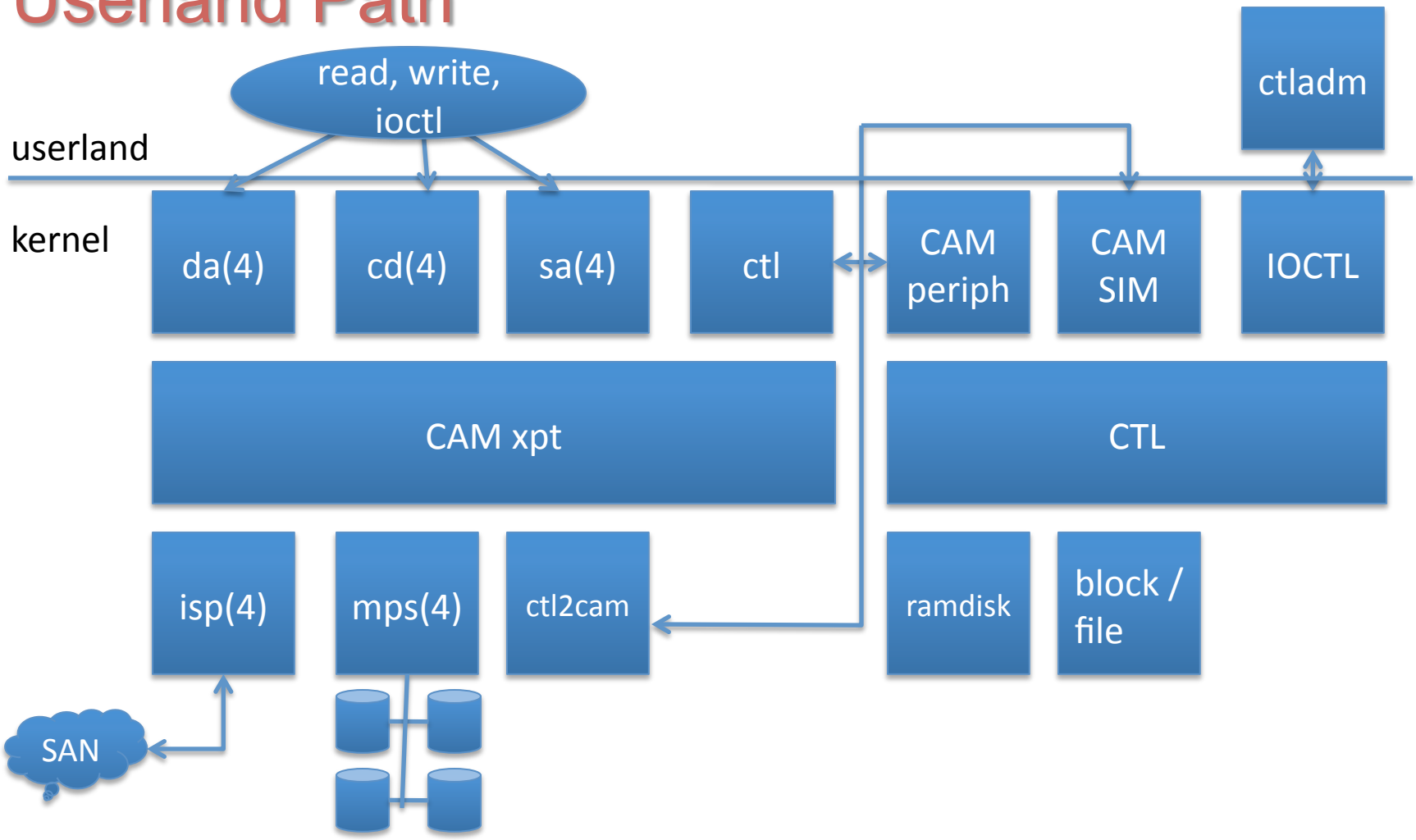
Normal Target Command Path



Internal Initiator Path



Userland Path



CTL Uses

- Turn a FreeBSD box into an external RAID box.
- CAM error injection testing: CTL used to test SCSI descriptor sense support for CAM.
- SCSI testing without SCSI hardware
- HBA bandwidth testing (using ramdisk backend)
- Prototyping new OS features (e.g. trasz@FreeBSD.org growfs work)

Demo

To-Do List

- Make CTL buildable as a module.
- Use devstat(9) for statistics collection.
- ZFS ARC backend for CTL.
- Use CAM CCBs instead of ctl_io structure.
- Full-featured High Availability support.
- Allow sending status back with final data on reads.
- Multi-thread CTL command processing.
- Use busdma S/G list format.
- Make data copy in the CTL SIM optional.
- More driver support.

Questions?