Scalable HEC I/O Forwarding Layer

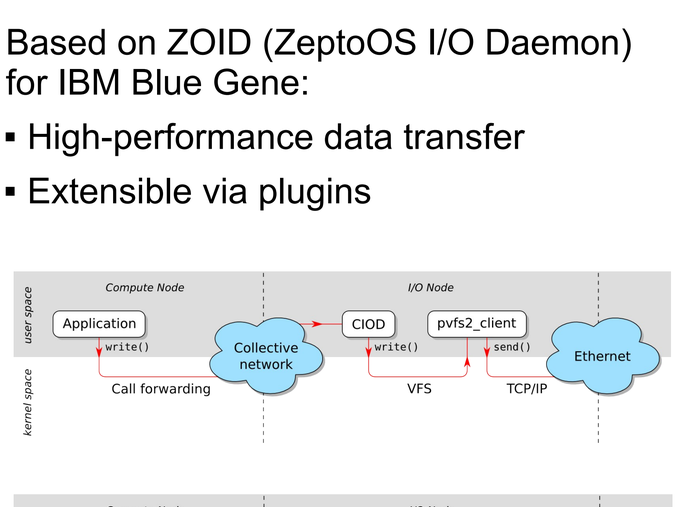
Rob Ross, Pete Beckman, Kamil Iskra **Argonne National Laboratory** *James Nunez, John Bent, Gary Grider* **Los Alamos National Laboratory**
Steve Poole **Oak Ridge National Laboratory** *Lee Ward* **Sandia National Laboratories** *Pete Wyckoff* **Ohio Supercomputer Center**

Mission

Design, build, and distribute a scalable, unified high-end computing I/O forwarding software layer that would be adopted and supported by DOE Office of Science and NNSA.

- Provide function shipping at the file system interface level (without requiring middleware) that enables asynchronous coalescing and I/O without jeopardizing determinism for computation
- Offload file system function from simple or full OS client processes to a variety of targets, from another core or hardware on the same system to an I/O node on a conventional cluster or a service node on a leadership class system
- Reduce the number of file system operations/clients that the parallel file system sees
- Support any/all parallel file system solutions
- Integrate with MPI-IO and any hardware features designed to support efficient parallel I/O

Software Stack



Focus Areas

- Portability across networks
- Support for multiple filesystems
- Hooking to applications
- Performance
- Security
- Testing

Platforms:

- IBM Blue Gene
- Cray XT Catamount
- Linux (clusters, Roadrunner, Cray XT)

I/O Forwarding Protocol

- ZOIDFS: stateless, NFSv3-like
- opaque, 16-byte file handles (no file descriptors)
- lookup, create (no open, close):

```
int zoidfs_lookup(const zoidfs_handle_t
                                *parent_handle,
                  const char *component_name,
                  const char *full_path,
                  zoidfs_handle_t *handle);
```

- Maximally flexible read, write:

```
int zoidfs_read(const zoidfs_handle_t
                                    *handle,
                int mem_count,
                void *mem_starts[],
                const size_t mem_sizes[],
                int file_count,
                const uint64_t file_starts[],
                uint64_t file_sizes[]);
```

Streamlined Forwarding

Based on ZOID (ZeptoOS I/O Daemon) for IBM Blue Gene:

- High-performance data transfer
- Extensible via plugins

Hooking to Applications

MPI-IO:

- ROMIO

POSIX:

- The SYSIO Library
  - VFS in userspace
  - File support for lightweight compute node kernels
  - Plugin architecture
- Modified GNU libc
- FUSE

Future

We will encourage the adoption of this framework both in production and for further research into I/O forwarding approaches.

- This framework will be available online under an open-source license
- We will test and support this solution on Leadership Class Systems/Parallel File System combinations at ANL, ORNL, Sandia, and LANL
- We will provide mailing lists for users to ask questions and obtain help