

ORACLE®



ORACLE®

Lustre Development

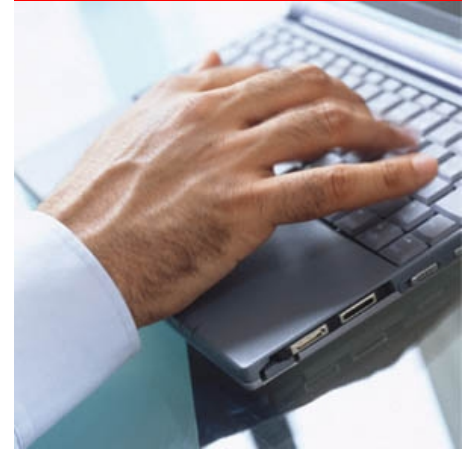
Eric Barton
Lead Engineer, Lustre Group



Lustre Development

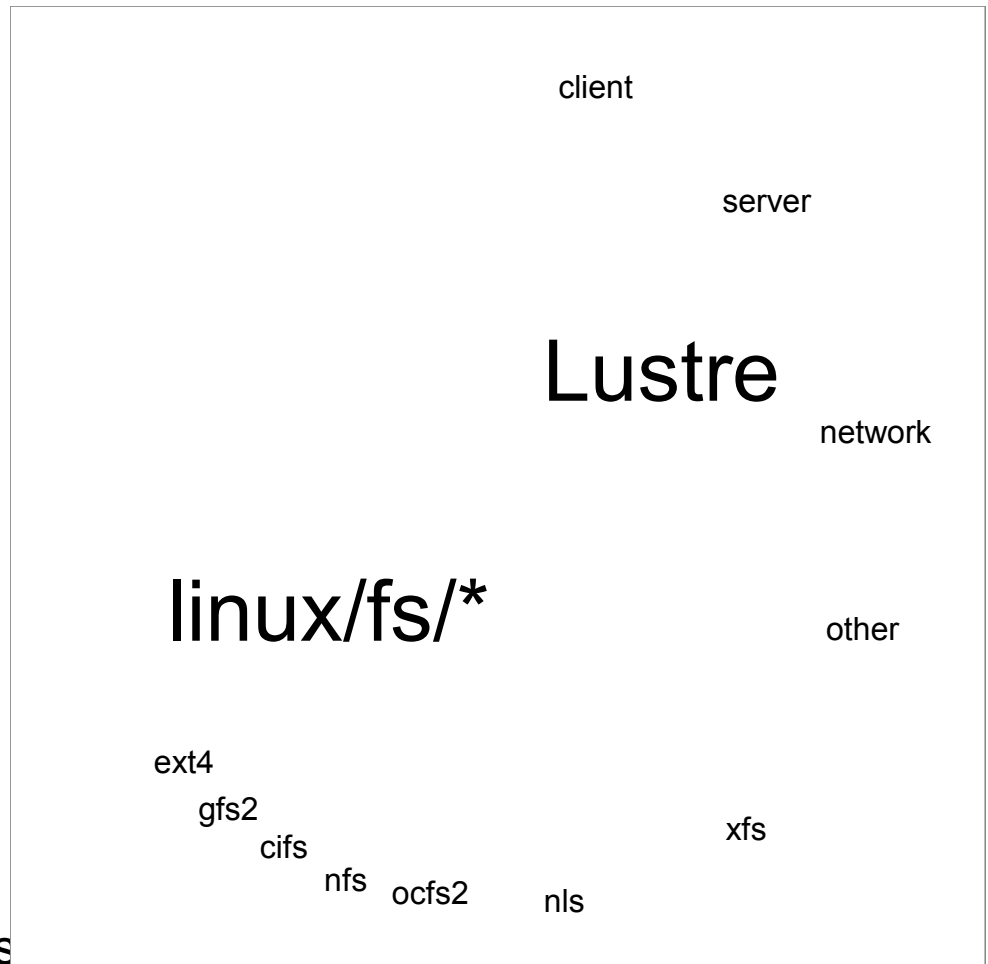
Agenda

- Engineering
 - Improving stability
 - Sustaining innovation
- Development
 - Scaling and performance
 - Ldiskfs and DMU
- Research
 - Scaling
 - Performance
 - Resilience

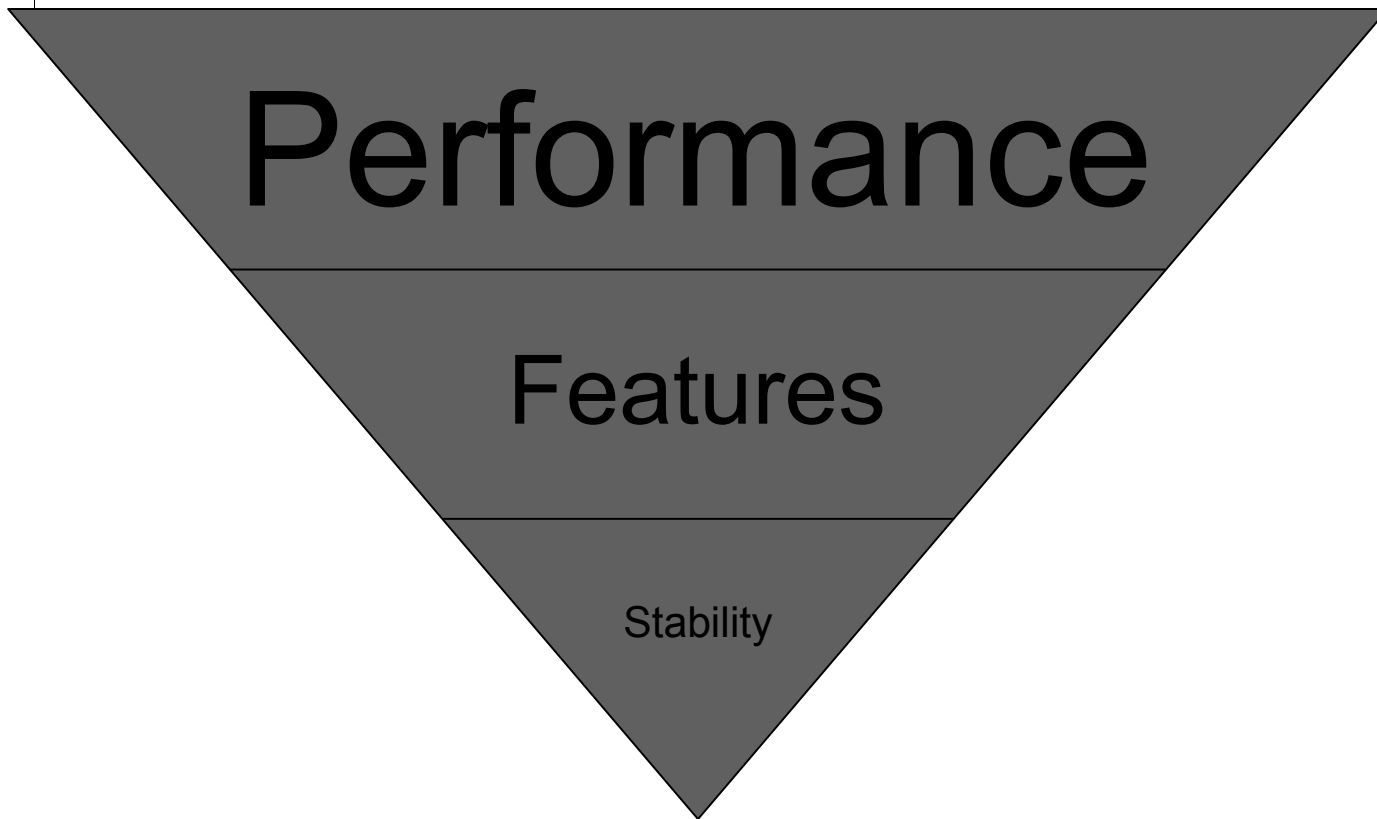


Engineering Lines of Code

- Lustre – 257 KLOC
- Total of all in-tree linux filesystems

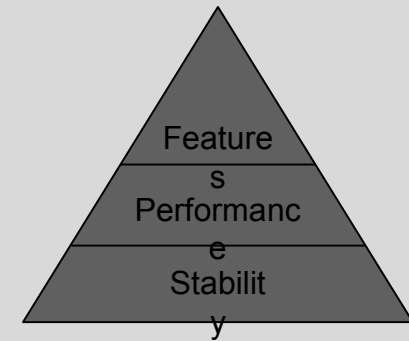


Engineering Historical Priorities



Engineering Priorities

- Stability
 - Reduce support incident rate
 - Reliable / predictable development
 - Address technical debt
- Performance & Scaling
 - Prevent performance regression
 - Exploit hardware improvements
- Features
 - Improve fault tolerance / recovery
 - Improve manageability



Engineering Knowledge

- ORNL
 - “Understanding Lustre Filesystem Internals”
- Lustre internals documentation project
 - Work in progress
 - Continuously maintained
- Subsystem map
- Narrative documentation
 - AsciiDoc
- Api documentation
 - Doxygen

	September	October	November	December	January	February	March (1-19)	Total Runs
ler	85%	100%	95%	95%	95%	100%	100%	308
simul	95%	72%	95%	95%	95%	100%	95%	298
racer	85%	48%	95%	87%	75%	95%	100%	297
sanity	85%	75%	95%	95%	71%	95%	100%	301
sanity_benchmark	85%	75%	95%	95%	82%	95%	100%	299
lustre-rsync_test	85%	85%	95%	95%	95%	85%	100%	248
sanityjn	95%	87%	57%	85%	87%	95%	100%	295
filebench	95%	95%	95%	95%	95%	95%	100%	305
replay_single	NT	95%	95%	95%	75%	95%	85%	299
metabench	NT	100%	NT	95%	100%	100%	100%	162
recovery_small	85%	85%	95%	95%	85%	95%	100%	285
replay_dedup	95%	79%	95%	95%	71%	95%	100%	302
sanity	95%	95%	95%	95%	95%	95%	100%	302
sanity_ops	95%	82%	95%	95%	85%	95%	95%	308
sanity_acc	95%	100%	100%	95%	95%	100%	100%	308
performance_sanity	85%	75%	87%	85%	75%	80%	85%	308
replay_vio	95%	71%	87%	95%	83%	95%	95%	305
replay_out_single	95%	91%	52%	95%	85%	100%	95%	307
conf_sanity	75%	84%	72%	87%	65%	95%	95%	302
pios	85%	85%	100%	82%	85%	100%	100%	54
parallel_scale	95%	95%	95%	95%	85%	95%	95%	58
lust_outlet	NT	95%	95%	95%	95%	85%	100%	48
recovery-end-scale	NT	75%	95%	95%	100%	85%	100%	48
recovery-double-scale	NT	85%	95%	95%	71%	95%	85%	48
recovery-random-scale	NT	75%	100%	100%	71%	75%	40%	48
out_pools	NT	25%	95%	85%	29%	25%	40%	42
parallel_scale_rfss4	NT	95%	95%	95%	NT	95%	95%	34
parallel_scale_rfss2	NT	95%	95%	95%	NT	15%	100%	36
metadata_updates	NT	95%	95%	95%	95%	95%	95%	40

Engineering

Branch management

- Prioritize major development branch stability
 - Solid foundation
 - Reliable / early regression detection
 - Predictable / sustainable development
- Gatekeeper
 - Control landing schedule
 - Enforce defective patch backout
 - Influence patch size for inspection / test
- Git
 - Retained all significant CVS history
 - Single repository covers everything
 - Much easier backouts

	September	October	November	December	January	February	March (1-19)	Total Runs
lcr	85%	100%	98%	98%	98%	100%	100%	308
simul	72%	72%	95%	95%	95%	100%	100%	298
rcsr	4%	42%	28%	57%	75%	100%	100%	207
sanity	4%	7%	6%	86%	71%	100%	100%	301
sanity_benchmark	65%	75%	88%	88%	82%	98%	100%	299
lustrv-sync_test	85%	85%	100%	9%	3%	85%	100%	248
sanityn	29%	87%	57%	86%	87%	100%	100%	295
libtest	1%	2%	8%	8%	8%	10%	10%	305
reply_single	1%	12%	8%	5%	7%	9%	8%	299
metabench	NT	100%	NT	9%	10%	100%	100%	162
recovery_small	2%	82%	100%	9%	83%	10%	100%	285
reply_dual	10%	79%	19%	3%	71%	10%	100%	302
sanity	100%	9%	1%	1%	1%	100%	100%	302
sanity_ops	35%	82%	10%	88%	85%	9%	9%	308
sanity_spc	1%	100%	100%	9%	3%	100%	100%	308
performance_sanity	4%	75%	67%	85%	75%	80%	86%	308
reply_vio	1%	71%	67%	1%	83%	9%	9%	305
reply_ext_single	10%	91%	52%	3%	8%	100%	100%	307
conf_sanity	73%	84%	72%	61%	65%	10%	10%	302
psis	40%	80%	100%	82%	86%	100%	100%	54
parallel_scale	1%	10%	5%	1%	6%	1%	1%	18
hot_updates	NT	10%	1%	10%	10%	8%	10%	48
recovery-ends-scale	NT	70%	9%	10%	10%	63%	10%	48
recovery-double-scale	NT	48%	9%	9%	71%	30%	8%	48
recovery-random-scale	NT	79%	100%	100%	71%	75%	40%	48
out_pools	NT	20%	10%	40%	29%	25%	10%	42
parallel_scale_rfss4	NT	0%	0%	0%	NT	10%	10%	34
parallel_scale_rfss2	NT	0%	0%	0%	NT	10%	100%	36
metadata_updates	NT	0%	0%	0%	0%	0%	0%	40

Engineering Test

- Hyperion
 - 100s of client nodes
 - Multimount – simulate 1000s of clients
 - Multiple test runs weekly
 - Leverage much earlier in development cycle
- Daily automated testing
 - Results vetting
- Improved defect observability
 - See trends
 - Discern regular v. intermittent issues
 - Early regression detection

	September	October	November	December	January	February	March (1-19)	Total Runs
lcr	80%	100%	98%	98%	98%	100%	100%	308
simul	91%	72%	91%	91%	99%	100%	91%	298
racer	93%	49%	93%	97%	75%	100%	100%	207
sanity	83%	7%	88%	86%	71%	97%	100%	311
sanity_benchmark	85%	75%	88%	89%	82%	98%	97%	299
lustre-rync_test	89%	85%	100%	93%	96%	85%	98%	248
sanityn	99%	87%	57%	86%	87%	98%	100%	295
dbcheck	91%	99%	80%	98%	88%	100%	100%	305
replay_single	91%	100%	91%	95%	76%	98%	87%	299
metabench	NT	100%	NT	98%	100%	100%	100%	162
recovery_small	92%	82%	98%	97%	83%	98%	100%	285
replay_dedup	98%	79%	100%	92%	71%	97%	98%	302
sanity	100%	98%	100%	98%	98%	100%	100%	302
sanity_ops	93%	82%	92%	88%	85%	91%	97%	308
sanity_acc	97%	100%	100%	98%	98%	100%	100%	308
performance_sanity	8%	75%	67%	85%	75%	80%	86%	308
replay_vio	9%	71%	67%	98%	83%	98%	97%	305
replay_out_single	99%	91%	92%	98%	88%	100%	98%	307
conf_sanity	73%	84%	72%	67%	65%	90%	98%	302
pios	82%	88%	100%	82%	86%	100%	100%	94
parallel_scale	9%	99%	97%	97%	87%	98%	81%	58
hot_walrus	NT	99%	97%	98%	100%	85%	100%	48
recovery-ends-scale	NT	70%	91%	100%	100%	83%	100%	48
recovery-double-scale	NT	88%	91%	97%	71%	90%	80%	48
recovery-random-scale	NT	79%	100%	100%	71%	75%	40%	48
out_pools	NT	29%	33%	40%	29%	29%	40%	42
parallel_scale_rfss4	NT	0%	0%	0%	NT	90%	7%	34
parallel_scale_rfss2	NT	0%	0%	0%	NT	13%	100%	36
metadata_updates	NT	0%	0%	0%	0%	0%	0%	40

Engineering Process

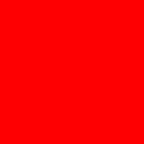
- Clear release objectives
 - Manage risk – stability / schedule uncertainty
 - Release blockers defined by bug priority
- Bi-weekly builds
 - Formal test plans
 - Prioritize test issues
- Daily review
 - Engineering progress
 - Testing results
 - Issue priorities

	September	October	November	December	January	February	March (1-19)	Total Runs
lcr	85%	100%	95%	95%	95%	100%	100%	308
simul	85%	72%	95%	95%	95%	100%	100%	298
racer	85%	40%	95%	87%	75%	100%	100%	207
sanity	85%	75%	95%	95%	82%	100%	100%	301
sanity_benchmark	85%	75%	95%	95%	82%	100%	100%	299
lustrv-sync_test	85%	85%	95%	95%	95%	85%	100%	248
sanityn	85%	87%	57%	85%	87%	95%	100%	295
dbhealth	85%	95%	85%	95%	85%	100%	100%	305
replay_single	85%	95%	95%	95%	75%	95%	85%	299
metabench	NT	100%	NT	95%	100%	100%	100%	162
recovery_email	85%	85%	95%	95%	85%	95%	100%	295
replay_dml	85%	79%	95%	95%	71%	95%	100%	302
sanity	85%	95%	95%	95%	95%	100%	100%	302
sanity_ops	85%	82%	95%	85%	85%	95%	95%	308
sanity_acc	85%	100%	100%	95%	95%	100%	100%	308
performance_sanity	85%	75%	87%	85%	75%	80%	85%	308
replay_vio	85%	71%	87%	95%	83%	95%	95%	305
recovery_double-scale	85%	91%	52%	95%	85%	100%	95%	307
recovery_random-scale	85%	91%	52%	95%	85%	100%	95%	307
conf_sanity	73%	84%	72%	87%	65%	95%	95%	302
psps	82%	88%	100%	82%	85%	100%	100%	54
parallel_scale	85%	95%	85%	95%	85%	95%	85%	58
hot_walrus	85%	95%	95%	95%	95%	95%	95%	48
recovery_end-scale	NT	75%	95%	95%	100%	83%	100%	48
recovery_double-scale	NT	88%	95%	95%	71%	95%	85%	48
recovery_random-scale	NT	75%	100%	95%	71%	75%	40%	48
out_posts	NT	25%	33%	45%	29%	25%	40%	42
parallel_scale_8ks4	NT	0%	0%	0%	NT	95%	7%	34
parallel_scale_8ks2	NT	0%	0%	0%	NT	15%	100%	36
metadata_updates	NT	0%	0%	0%	0%	0%	0%	40

Development Priorities

- Lustre 1
 - Maintenance
- Lustre 2
 - Stabilization
 - Performance
 - Eliminate regressions
 - Land improvements
 - Features

The screenshot shows the 'Lustre Internals Documentation' page with the Oracle logo. The main content area is titled 'Subsystem Map' and includes a 'Table of Contents' with links for '1. The Map', '2. Developer API Documentation', and '3. All documented subsystems'. Below this is a grid of blue boxes representing various subsystems, each with a list of sub-topics. The 'Client' subsystem includes CIO, CLOM, and ethto_client. The 'Protocol' subsystem includes pfsync, Cache, Recovery, and LQLM. The 'LNET' subsystem includes LND, MND, and Routing. The 'Infrastructure' subsystem includes libfs, Build/Packaging, and Configuration. Other subsystems shown include VFS, Disk, Server, Security, and Test.



The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.

The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Development Projects

- SMP scaling
 - Exploit multicore servers
 - Improve metadata throughput
- Platform portability
 - Extend OS-specific / portable layering to metadata
 - Formalize porting primitives
- Ldiskfs / DMU(ZFS) OSD
 - Pluggable storage subsystem
- HSM
- Clean server shutdown / restart
 - Simplify version interoperation / rolling upgrade

Size on MDS

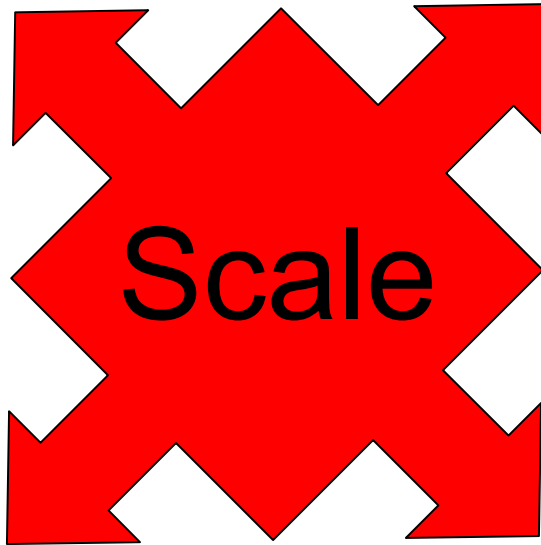
- $O(n) \rightarrow O(0)$ read-only metadata ops

The screenshot shows the LUSTRE INTERNALS DOCUMENTATION website. At the top, there is a navigation bar with the LUSTRE logo, "LUSTRE INTERNALS DOCUMENTATION", and the ORACLE logo. Below the navigation bar, there is a "Subsystem Map" section. On the left, there is a "Table of Contents" with links for "1. The Map", "2. Developer API Documentation", and "3. All documented subsystems". The main content area is titled "1. The Map" and contains a grid of blue boxes representing different subsystems. The boxes are organized into columns: VFS (fsck, windows, linkers, fsck), Client (CDD, CND, etch client), Protocol (pfsync, Cache, Recovery, LDM), LNET (LND, mds, mds, mds), Infrastructure (libfs, Build/Packaging, Configuration), Disk (Ldiskfs OSD, DMU OSD, LDM, OSD, OSD), Server (fsck, fsck, fsck, fsck, fsck, fsck, fsck, fsck), Security (fsck, fsck, fsck, fsck, fsck, fsck, fsck, fsck), and Test (libfs, Build/Packaging, Configuration, Test, Self Test).

Research Priorities

Metadata
Performance

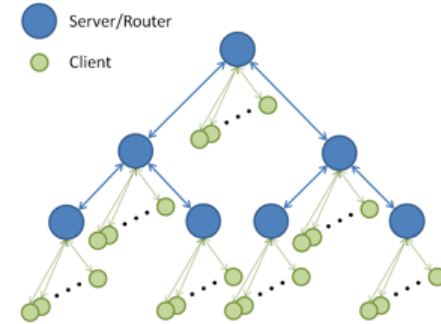
I/O
Performance



Numbers
of
clients

Resilience
and
Recovery

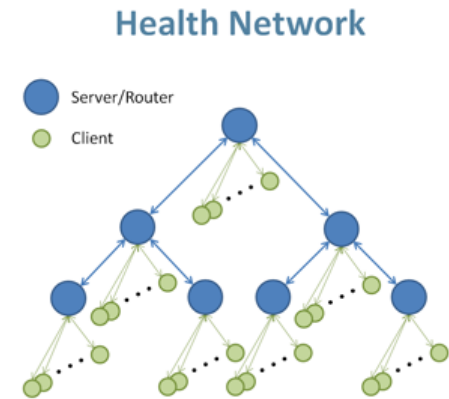
Health Network



Research

Numbers of Clients

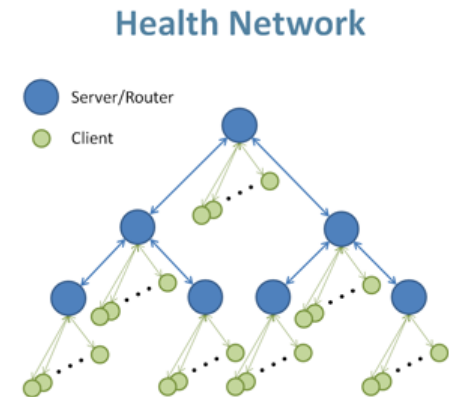
- Currently able to accommodate 10,000s
- Next steps
 - System call forwarders - 10-100x
- Further steps
 - Caching proxies
 - Subtree locking



Research

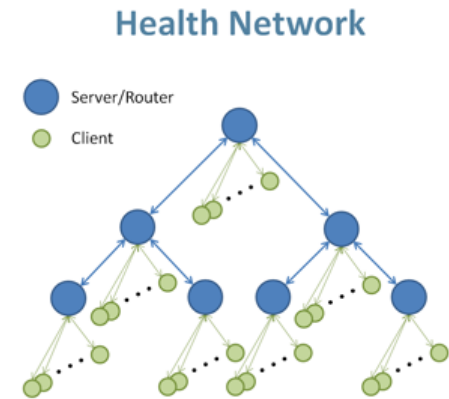
I/O

- Initial NRS experiments encouraging
 - 40% Read improvement
 - 60% Write improvement
- Next steps
 - Larger scale prototype benchmarking
 - Exploit synergy with SMP scaling work
- Further steps
 - Global NRS policies
 - Quality of service



Research Metadata

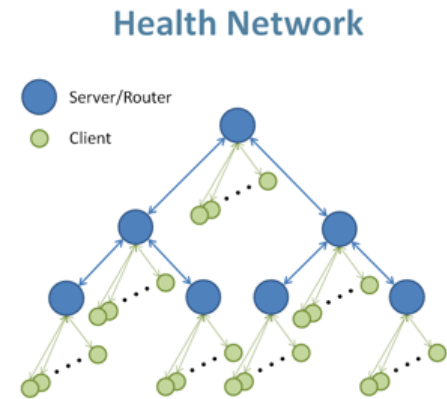
- SMP scaling
 - Deeper locking / CPU affinity issues
- CMD Preview
 - Sequenced / synched distributed updates
 - Characterise performance
- Next Steps
 - Productize CMD Preview
- Further Steps
 - CMD based on epochs



Research

Resilience & Recovery

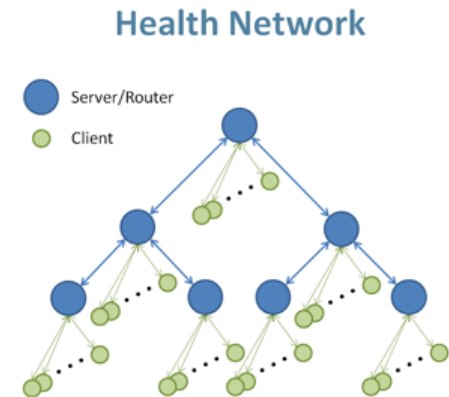
- $O(n)$ pinger overhead / detection latency
- Overreliance on client timeouts
 - $O(n)$ to distinguish server congestion from death
 - Include disk latency
 - Required to detect LNET router failure
- Over-eager server timeouts
 - Can't distinguish LNET router failure from client death
- Recovery affects everyone
 - Transparency not guaranteed after recovery window expires
 - COS/VBR only partial solution
 - MDT outage disconnects namespace
 - Epoch recovery requires global participation



Research

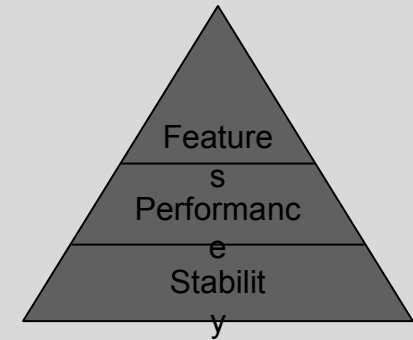
Resilience & Recovery

- Scalable health network design
 - Out-of-band communications
 - Low latency global notifications
 - Collectives: Census, LOVE reduction etc
 - Clear completion & network partition semantics
 - Self-healing
- Next steps
 - HN prototype
 - OST mirroring
- Further steps
 - Epoch based SNS



Lustre Development Summary

- Prioritize stability
 - Continued product quality improvements
 - Predictable release schedule
 - Sustainable development
- Continued innovation
 - Prioritized development schedule
 - Planned product evolution



ORACLE®