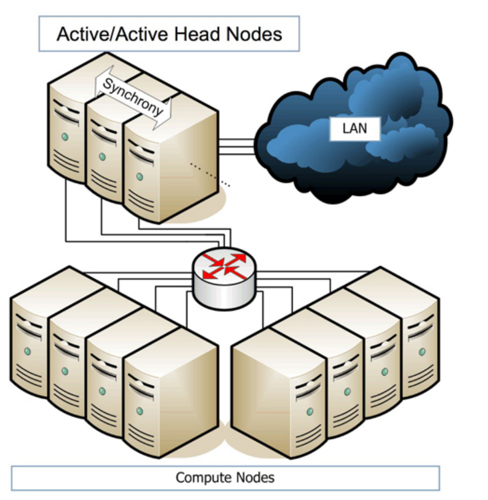
Resiliency for High-Performance Computing Systems

Stephen L. Scott, Christian Engelmann, Hong H. Ong, Geoffroy R. Vallée, Thomas Naughton, Anand Tikotekar, George Ostrouchov (Oak Ridge National Laboratory)
Chokchai Leangsuksun, Nichamon Naksinehaboon, Raja Nassar, Mihaela Paun (Louisiana Tech University)
Frank Mueller, Chao Wang, Arun Nagarajan, Jyothish Varma (North Carolina State University)
Xubin He, Li Ou, Xin Chen (Tennessee Technological University)

Research and development goals

- Efficient redundancy strategies for head and service nodes in HPC systems to provide high availability as well as high performance of critical infrastructure services
- Reactive fault tolerance for HPC compute nodes utilizing the job pause approach as well as checkpoint interval and placement adaptation to actual and predicted system health threats
- Proactive fault tolerance using system-level virtualization in HPC environments for preemptive migration of computation away from compute nodes that are about to fail
- Reliability analysis for identifying pre-fault indicators, predicting failures, and modeling and monitoring of individual component and overall HPC system reliability
- Holistic fault tolerance technology through combination of adaptive proactive and reactive fault tolerance mechanisms in conjunction with system health monitoring and reliability analysis

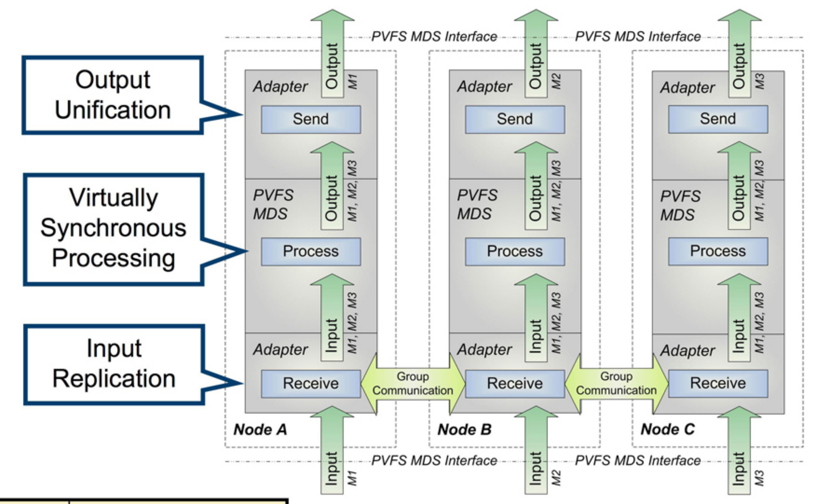
Symmetric active/active redundancy for head and service nodes



- Many active head nodes
- Work load distribution
- Symmetric replication between head nodes
- Continuous service
- Always up to date
- No fail-over necessary
- No restore-over necessary
- Virtual synchrony model
- Complex algorithms

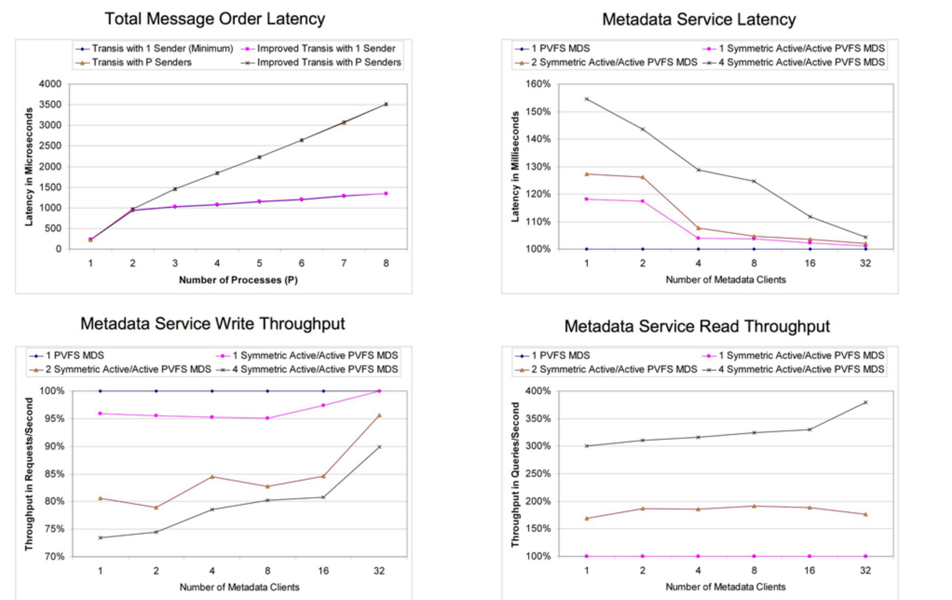
➡ Prototypes for PBS Torque and Parallel Virtual File System metadata server

Symmetric active/active replication



Nodes	Availability	Est. annual downtime
1	98.58%	5d 4h 21m
2	99.97%	1h 45m
3	99.9997%	1m 30s

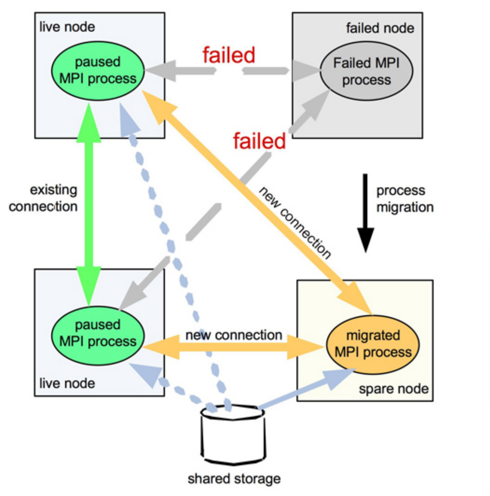
Symmetric active/active Parallel Virtual File System metadata service



Reactive vs. proactive fault tolerance for compute nodes

- Reactive fault tolerance:
  - State saving during failure-free operation
  - State recovery after failure
  - Assured quality of service, but limited scalability
- Proactive fault tolerance:
  - System health monitoring and online reliability modeling
  - Failure anticipation and prevention through prediction and reconfiguration before failure
  - Highly scalable, but not all failures can be anticipated
- Ideal solution: Matching combination of both

Enhanced reactive fault tolerance with LAM/MPI+BLCR job pause mechanism

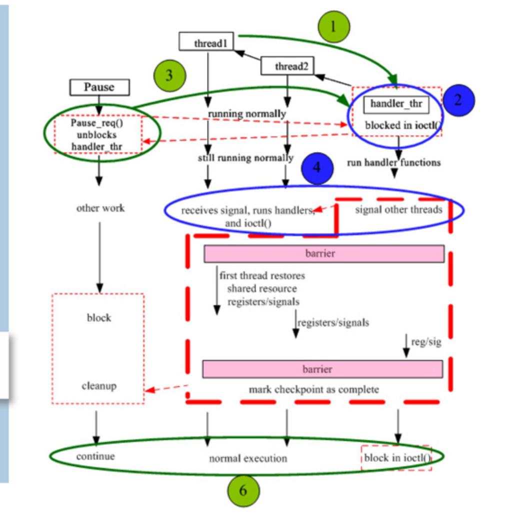


- Operational nodes: Pause
  - BLCR reuses existing processes
  - LAM/MPI reuses existing connections
  - Restore partial process state from checkpoint
- Failed nodes: Migrate
  - Restart process on new node from checkpoint
  - Reconnect with paused processes
- Scalable MPI membership management for low overhead

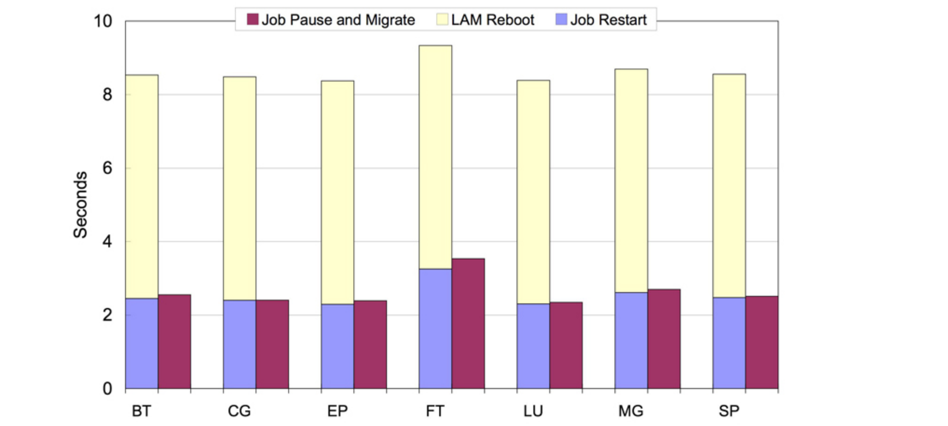
➡ Efficient, transparent, and automatic failure recovery

New job pause mechanism in BLCR

1. Application registers threaded callback→spawns callback thread
2. Thread blocks in kernel
3. Pause utility calls ioctl(), unblocks callback thread
4. All threads complete callbacks and enter kernel
5. New: All threads restore part of their states
6. Run regular application code from restored state

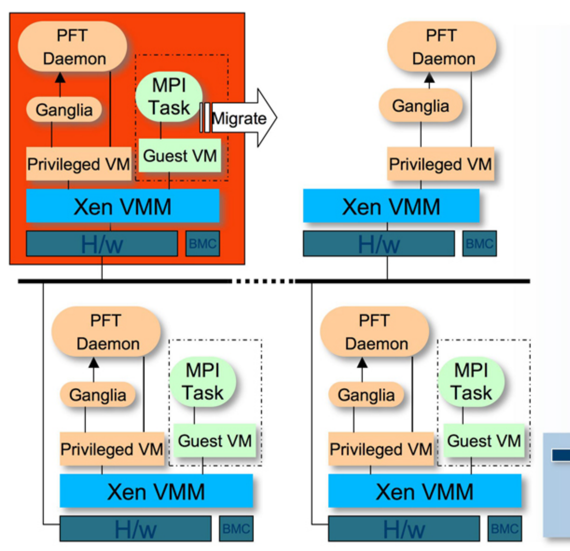


LAM/MPI+BLCR job pause performance



- 3.4% overhead over job restart, but
  - No LAM reboot overhead
  - Transparent continuation of execution
- No requeue penalty
- Less staging overhead

Proactive fault tolerance using Xen virtualization

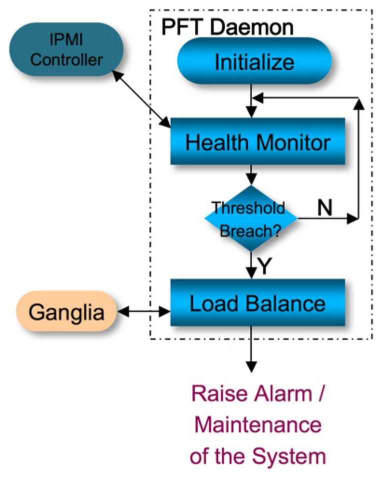


- Stand-by Xen host (spare node without guest VM)
- Deteriorating health:
  - Migrate guest VM to spare node
- New host generates unsolicited ARP reply
  - Indicates that guest VM has moved
  - ARP tells peers to resend to new host

➡ Novel fault tolerance scheme that acts before a failure impacts a system

Proactive fault tolerance daemon

- Runs in privileged domain (host)
- Initialization
  - Read safe threshold from config file
  - Init connection with IPMI controller
  - Obtain/filter set of available sensors
- Health monitoring
  - Read sensors from IPMI controller
  - Periodically sample data
  - Trigger load balancing if exceeding sensor threshold
- VM migration
  - Select target based on load
  - Invoke Xen live migration for VM

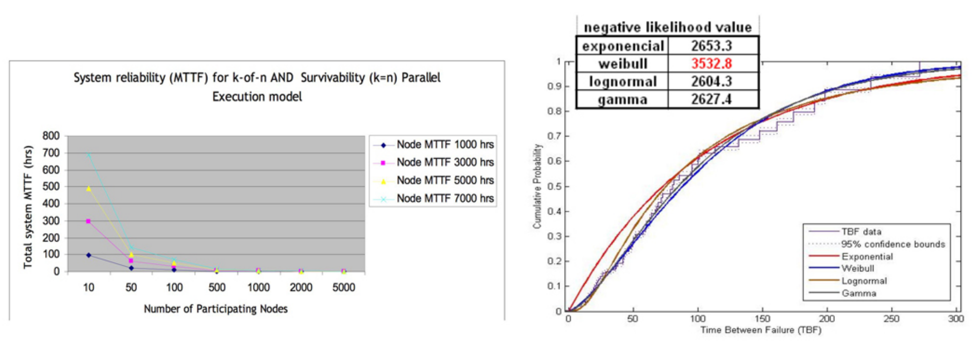


VM migration performance impact

- Single node failure: 0.5-5% additional cost over total wall clock time
- Double node failure: 2-8% additional cost over total wall clock time

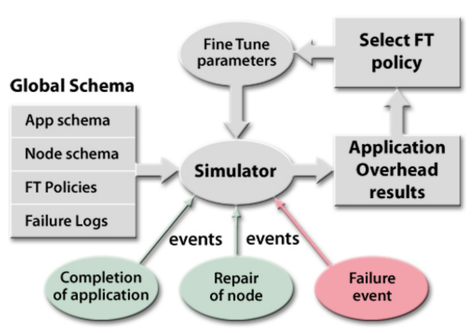
HPC reliability analysis and modeling for prediction and anticipation

- Programming paradigm and system scale impact reliability
- Reliability analysis:
  - Estimate mean time to failure (MTTF)
  - Obtain failure distribution: Exponential, Weibull, Gamma, ...
- Feedback into fault tolerance schemes for adaptation

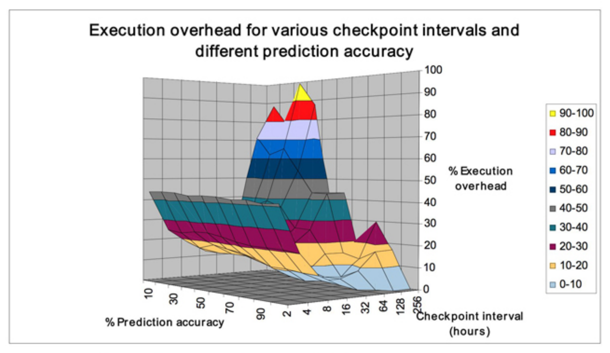


Simulation framework for HPC fault tolerance policies

- Evaluation of fault tolerance policies
  - Reactive only
  - Proactive only
  - Reactive/proactive combination
- Evaluation of fault tolerance parameters
  - Checkpoint interval
  - Prediction accuracy
- Event-based simulation framework using actual HPC system logs
- Customizable simulated environment
  - Number of active and spare nodes
  - Checkpoint and migration overheads



Combination of proactive and reactive fault tolerance: Simulation example 1

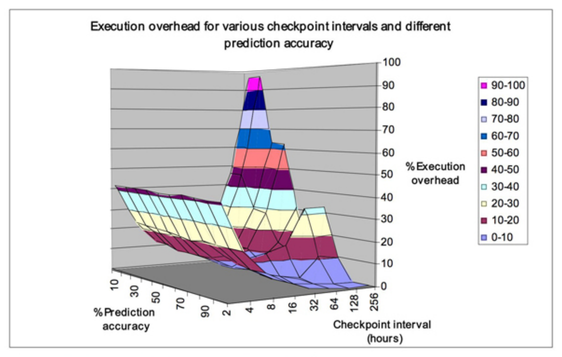


- Best: Prediction accuracy >60% and checkpoint interval 16-32 hours
- Better than only proactive or only reactive
- Results for higher prediction accuracies and very low checkpoint intervals are worse than only proactive or only reactive

Number of processes	125
Active nodes / Spare nodes	125 / 12
Checkpoint overhead	50 min/checkpoint
Migration overhead	1 min/migration

Simulation based on ASCI White system logs (nodes 1 – 125 and 500-512)

Combination of proactive and reactive fault tolerance: Simulation example 2

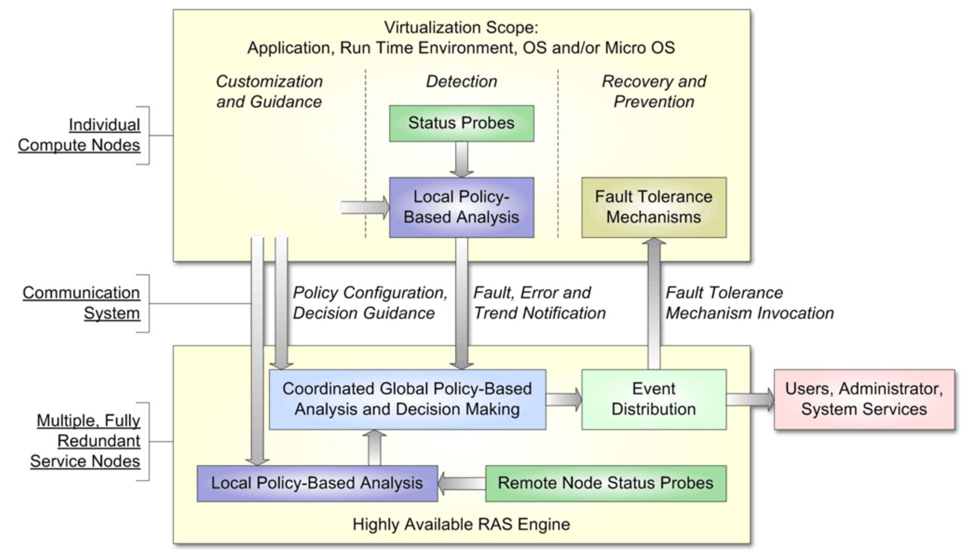


- Best: Accuracy >60%, interval 16-64h
- 70% and 32 hours:
  - 8% gain over reactive only
  - 24% gain in over proactive only
- 80% and 32 hours:
  - 10% gain over reactive only
  - 3% loss over proactive only

Number of processes	125
Active nodes / Spare nodes	125 / 12
Checkpoint overhead	50 min/checkpoint
Migration overhead	1 min/migration

Simulation based on ASCI White system logs (nodes 126 – 250 and 500-512)

A holistic resiliency framework for high-performance computing



Office of Science, U.S. Department of Energy
Point of Contact: Christian Engelmann (engelmannc@ornl.gov)
www.fastos.org/ras
Oak Ridge National Laboratory
Louisiana Tech University
NC State University
Tennessee Tech University
The University of Reading