ULFM
USER LEVEL FAILURE MITIGATION

User Level Failure Mitigation is a set of MPI extensions to report errors, provide interfaces to stabilize the distributed state, and restore the communication capabilities in applications affected by process failures. Relevant communicators, RMA windows, and I/O files can be reconstructed online, without restarting the application, as required by the user recovery strategy.

ULFM's capability to restore communication after a fault is crucial infrastructure for supporting the design and deployment of production-grade recovery strategies. Multiple applications and programming frameworks are already taking advantage of ULFM constructs to deliver varied fault tolerance strategies—from run-through algorithms that continue without rejuvenating the lost processes, to methods that restore the lost processes and their dataset—either from checkpoints or from checkpoint-free forward recovery techniques.

**Open MPI ULFM 5.0**
Distributed as part of Open MPI 5.0
- All new features of Open MPI with resilience support
- Inherits the same build and runtime arguments and same modular software stack as Open MPI
- Resilience support with most networks and job schedulers:
  - Networks: UCX, uGNI, Open IB, TCP, CMA
  - Shared-memory
  - Launchers: Slurm, ALPS, PBS
- No measurable failure-free overhead on HPC networks
- Beta resilience support for Open Fabric transport, RMA, and FILE operations

**ULFM USER COMMUNITIES**
- Programming languages X10 over MPI with "DeadPlace" exception support
- CoArrays Fortran with "FailedImages" extension
- Checkpointing Frameworks Fenix, CRAFTS, LFLP, VELOC
- Applications PDE solvers, FTLA
- Non-HPC workloads SAP Databases, Hadoop over MPI

**FAILURE DETECTION**
New flexible group-centric failure detection modes let applications monitor more processes at no cost with regard to fault free performance.

**AGREEMENT**
Users can stabilize the global state after a failure with this consensus operation. ERA (early returning agreement) latency is only double Cray's optimized, non-resilient Allreduce.

**RELIABLE BROADCAST**
Revoke permits disseminating fault information. It’s latency is lower than a barrier. A reliable broadcast causes an insignificant difference from the failure free case, illustrating that it can complete locally before synchronization anymore and it can complete locally before.

**ULFM FEATURES**
**FLEXIBILITY**
- No predefined recovery model is imposed or favored.
- Instead, a set of versatile APIs is included to provide support for different recovery styles (e.g., checkpoint, ABFT, iterative, Master-Worker).
- Application directs the recovery, and it only pays for the level of protection it needs.
- Recovery can be restricted to a subgroup, thereby preserving scalability and easing the composition of libraries.

**PERFORMANCE**
- Protective actions are outside of critical MPI routines.
- MPI implementors can uphold communication, collective, one-sided, and I/O management algorithms unmodified.
- Encourages programs to be reactive to failures, and cost manifests only at recovery.

**PRODUCTIVITY**
- Backward compatible with legacy, fragile applications.
- Simple and familiar concepts to repair MPI.
- Provides key MPI concepts to enable FT support from library, runtime, and language extensions.

**STANDARDIZATION**
- Parts of ULFM, like the operational error model and the fact that errors should not "break" MPI, have already been standardized in MPI 4.0.
- Standardization effort continues to integrate advanced recovery features (like non-blocking recovery, session recovery) in MPI 5.0.

**ULFM AT SC23**
**TUTORIAL**
Sunday, November 12
Room 405 8:30am to 5:00pm MST
Fault-Tolerance for High Performance and Big Data Applications: Theory and Practice

**WORKSHOP**
Monday, November 13
Room 501-502 1:50pm to 4:00pm MST
Elastic Deep Learning through Resilient Collective Operations

**BOF**
Tuesday, November 14
Room 205-207 12:15pm to 1:15pm MST
Introducing MPI 4.1, the Newest Version of the Message Passing Interface Standard

**BOF**
Wednesday, November 15
Room 405-406-407 12:15pm to 11:15pm MST
Open MPI State of the Union
Resilience Extensions for MPI: ULFM

ULFM provides targeted interfaces to empower recovery strategies with adequate options to restore communication capabilities and global consistency, at the necessary levels only.

Continue Across Errors
In ULFM, failures do not alter the state of MPI communicators. Point-to-point operations can continue undisturbed between non-faulty processes. ULFM imposes no recovery cost on simple communication patterns that can proceed despite failures.

Exceptions in Contained Domains
A process can use MPI_Comm_win_file_revoke to propagate an error notification on the entire group, and could, for example, interrupt other ranks to join a coordinated recovery.

Full-Capability Recovery
Allowing collective operations to operate on damaged MPI objects (communicators, RMA windows, or files) would incur unacceptable overhead. The MPI_Comm_shrink routine builds a replacement communicator—excluding failed processes—that can be used to resume collective operations in malleable applications, spawn replacement processes in non-moldable applications, and rebuild RMA windows and files.

RECENT RESULTS: Evaluate the Cost and Expressivity of Asynchronous Recovery

Error Scoping
Adding per-communicator (window/file) control knobs for the application to control the scope of error reporting: set Info key mpix_error_range on a communicator to control which errors interrupt MPI calls.

* "local": current ULFM behavior: report an error only when communicating with a failed peer (e.g., recv from failed process, collective communication) default, current ULFM
* "group": report errors (i.e., REVOKE) for a failure at any process with a rank in the comm/win/file (e.g., in recv from an alive process in comm)
* "universe": report errors (i.e., REVOKE) for a failure anywhere in "universe"

Error Uniformity
All processes partake in a collective operation, should they return an error in unison? Use sets info key mpix_error_uniform on a communicator to control if error reports need to be uniform.

* "local": errors reported as needed to inform of invalid outputs (buffers/comm/ssh) at the reporting rank (i.e., other ranks may report success) default, current ULFM
* "create": if communicator/win/file creation operations (e.g., comm_split, file_open, win_create, comm_spawn) reports at a rank, it has reported the same ERR_PROC_FAILED/REVOKE at all ranks
* "coll": same as above, for all collective calls (including creates)

Asynchronous Error Recovery
Error recovery is difficult to overlap, because MPI currently misses asynchronous dynamic processes constructs.

* Adding MPI_COMM_ISHRINK to enable asynchronous failed processes exclusion
* Adding MPI_COMM_ISPAWN (and ICONNECT/ACCEP) to enable asynchronous spare respawn (as well as many other non-fl application use cases)