# MPI-IO Support

The Message Passing Interface (MPI) Standard, maintained by the [MPI Forum](https://www.mpi-forum.org/docs/), includes a chapter on MPI-IO.

[ROMIO](https://www.mcs.anl.gov/projects/romio/) is a well-known implementation of MPI-IO and is included in many MPI implementations. DAOS provides its own MPI-IO ROMIO ADIO driver. This driver has been merged in the upstream MPICH repository, see https://github.com/pmodels/mpich/tree/main/src/mpi/romio/adio/ad\_daos for details.

!!! note Starting with DAOS 1.1.3, the --svc parameter (number of service replicas) is no longer needed, and the DAOS API has been changed accordingly. Patches have been contributed to MPICH that detect the DAOS API version to gracefully handle this change, but those patches have not yet been picked up in the MPI releases below. For details check the latest commits [here](https://github.com/pmodels/mpich/commits/main?author=mchaarawi).

## MPI Implementations that support DAOS

### MPICH

The DAOS ROMIO ADIO driver has been accepted into [MPICH](https://www.mpich.org/). It is included in [mpich-3.4.1 (released Jan 2021)](https://www.mpich.org/downloads/).

### Building MPICH with DAOS Support

To build MPICH, including ROMIO with the DAOS ADIO driver:

export MPI\_LIB=""  
  
git clone https://github.com/daos-stack/mpich  
  
cd mpich  
  
./autogen.sh  
  
mkdir build; cd build  
  
../configure --prefix=dir --enable-fortran=all --enable-romio \  
 --enable-cxx --enable-g=all --enable-debuginfo --with-device=ch3:sock \  
 --with-file-system=ufs+daos --with-daos=/usr --with-cart=/usr  
  
make -j8; make install

This assumes that DAOS is installed into the /usr tree, which is the case for the DAOS RPM installation.

!!! note In DAOS 1.0, CART was packaged separately from DAOS, and the --with-cart option was needed to allow separate installation locations. With DAOS 1.1 or higher, CART is included in the main DAOS packages and the --with-cart option is no longer needed. An MPICH pull request is in process to remove the option from the MPICH configure step, but currently it is still required to build MPICH.

Set the PATH and LD\_LIBRARY\_PATH to where you want to build your client apps or libs that use MPI to the path of the installed MPICH.

### Intel MPI

The [Intel MPI Library](https://software.intel.com/content/www/us/en/develop/tools/mpi-library.html) includes DAOS support since the [2019.8 release](https://software.intel.com/content/www/us/en/develop/articles/intel-mpi-library-release-notes-linux.html).

Note that Intel MPI uses libfabric (both 2019.8 and 2019.9 use libfabric-1.10.1-impi).  
Care must be taken to ensure that the version of libfabric that is used is at a level that includes the patches that are critical for DAOS. DAOS 1.0.1 includes libfabric-1.9.0, and the DAOS 1.1.2 pre-release includes libfabric-1.11.1.

To use DAOS 1.1 with Intel MPI 2019.8 or 2019.9, the libfabric that is supplied by DAOS (and that is installed into /usr/lib64 by default) needs to be used by listing it first in the library search path:

export LD\_LIBRARY\_PATH="/usr/lib64/:$LD\_LIBRARY\_PATH"

The next Intel MPI release is expected to contain a version of libfabric that includes all patches to work with DAOS. It will then no longer be necessary to override the libfabric version that is shipped with Intel MPI by the version provided by DAOS.

### Open MPI

[Open MPI](https://www.open-mpi.org/) 4.0.5 does not yet provide DAOS support. Since one of its MPI-IO implementations is based on ROMIO, it will likely pick up DAOS support in an upcoming release.

### MVAPICH2

[MVAPICH2](https://mvapich.cse.ohio-state.edu/) 2.3.4 does not yet provide DAOS support. Since its MPI-IO implementation is based on ROMIO, it will likely pick up DAOS support in an upcoming release.

## Testing MPI-IO with DAOS Support

Build any client (HDF5, ior, mpi test suites) normally with the mpicc command and mpich library installed above (see child pages).

To run an example with MPI-IO:

1. Create a DAOS pool on the DAOS server(s). This will return a pool uuid “puuid” and service rank list “svcl”.
2. Create a POSIX type container: daos cont create --pool=puuid --type=POSIX This will return a container uuid “cuuid”.
3. At the client side, the following environment variables need to be set: export DAOS\_POOL=puuid; export DAOS\_SVCL=svcl; export DAOS\_CONT=cuuid. Alternatively, the unified namespace mode can be used instead.
4. Run the client application or test. MPI-IO applications should work seamlessly by just prepending daos: to the filename/path to use the DAOS ADIO driver.

## Known limitations

Limitations of the current implementation include:

* Reading Holes does not return 0, but leaves the buffer untouched.
* No support for MPI file atomicity, preallocate, or shared file pointers.