

# **MVAPICH2 3.0a Quick Start Guide**

MVAPICH TEAM

NETWORK-BASED COMPUTING LABORATORY  
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
THE OHIO STATE UNIVERSITY

<http://mvapich.cse.ohio-state.edu>

Copyright (c) 2001-2022  
Network-Based Computing Laboratory,  
headed by Dr. D. K. Panda.  
All rights reserved.

Last revised: August 12, 2022

# 1 Overview

This Quick Start contains the necessary information for MVAPICH2 users to download, install, and use MVAPICH2 3.0a. Please refer to our [User Guide](#) for the comprehensive list of all features and instructions about how to use them.

MVAPICH2 (pronounced as “em-vah-pich 2”) is an *open-source* MPI software to exploit the novel features and mechanisms of high-performance networking technologies (InfiniBand, iWARP, RDMA over Converged Enhanced Ethernet (RoCE v1 and v2), Slingshot 10, and Rockport Networks) and deliver best performance and scalability to MPI applications. This Alpha release of MVAPICH2 3.0 adds support for the Cray Slingshot 11, Cornelis OPX, and Intel PSM3 interconnects through the OFI libfabric library, and for the UCX communication library.

Please note that as this is an Alpha release, not all features have been implemented and not all performance will be optimal. For best performance on Mellanox InfiniBand, RoCE, iWARP, Slingshot 10 or lower, Rockport Networks, and Intel TrueScale or Omni-Path adapters with PSM2, please use MVAPICH2 2.3.7.

This software is developed in the [Network-Based Computing Laboratory \(NBCL\)](#), headed by [Prof. Dhabaleswar K. \(DK\) Panda](#) since 2001.

More details on MVAPICH2 software, users list, mailing lists, sample performance numbers on a wide range of platforms and interconnects, a set of OSU benchmarks and related publications can be obtained from our [website](#).

## 2 Build MVAPICH2 from Source

The MVAPICH2 3.0a source code package includes MPICH 3.4.3. All the required files are present in a single tarball.

### 2.1 Download & Unpack

Download the most recent distribution tarball from <http://mvapich.cse.ohio-state.edu/download/mvapich/mv2/mvapich2-3.0a.tar.gz>

```
$ wget http://mvapich.cse.ohio-state.edu/download/mvapich/mv2/mvapich2-3.0a.tar.gz
$ gzip -dc mvapich2-3.0a.tar.gz | tar -x
$ cd mvapich2-3.0a
```

### 2.2 Configure

If you have either UCX or OFI (libfabric) installed on your system on a default PATH you can use the default configuration to automatically detect the installed communication library. . .

```
$ ./configure
```

#### 2.2.1 The UCX Device - IB Networks Adapters

If you’re using a Mellanox InfiniBand, RoCE, iWARP, Slingshot 10, or Rockport Networks network adapter you can use the UCX library configuration. . .

```
$ ./configure --with-device=ch4:ucx
```

If a UCX installation is available on a system PATH that version will be used, otherwise the included version will be built. To configure with a particular version of UCX please use the following configuration. . .

```
$ ./configure --with-device=ch4:ucx --with-ucx=[PATH]
```

To force the included version to be built, you may use the following. . .

```
$ ./configure --with-device=ch4:ucx --with-ucx=embedded
```

### 2.2.2 The OFI Device - PSM/OPX Adapters

If you're using an Intel TrueScale, Intel Omni-Path, or Intel Columbiaville adapter you should use the OFI library configuration. . .

```
$ ./configure --with-device=ch4:ofi
```

If an OFI installation is available on a system PATH that version will be used, otherwise the included version will be built. The included version of OFI is v1.15.1 and will support the PSM, PSM2, PSM3, and OPX libraries. If you require a different libfabric version please use the following. . .

```
$ ./configure --with-device=ch4:ofi --with-libfabric=[PATH]
```

To force the included version to be built, you may use the following. . .

```
$ ./configure --with-device=ch4:ofi --with-libfabric=embedded
```

### 2.2.3 The OFI Device - Cray Slingshot 11 Adapters

If you're using a Cray Slingshot 11 Adapter you must use the OFI library configuration. . .

```
$ ./configure --with-device=ch4:ofi --with-libfabric=[PATH]
```

Where [PATH] points to the directory with the custom Cray libfabric library installed. This path should include both a lib and include directory. Using a non-Cray version of libfabric, or the embedded version, is not supported and will lead to poor performance.

### 2.2.4 Other Configure Options

—**prefix** This option tells the build system where to install mvapich2. If this option is not given mvapich2 will be installed in /usr/local.

—**disable-shared** This option tells the build system to create static libraries only. By default, both the shared and static libraries are built and installed.

—**enable-g=all** —**enable-error-messages=all** This option controls the amount of debugging information available in the MPI library. By default these are disabled since this will affect the size and speed of the MPI library.

## 2.2.5 More Options

MVAPICH2 supports many other configure and run time options which may be useful for advanced users. Please refer to our [User Guide](#) for more complete details.

## 2.3 Build & Install

```
$ make -j          # parallel build
$ make install
```

## 3 Run MPI Program

In this section we will demonstrate how to build and run a hello world program which uses mpi.

```

////////////////////////////////////
.Sample MPI code [mpihello.c]
-----
#include <mpi.h>
#include <stdio.h>
#include <unistd.h>

int main(int argc, char *argv[])
{
    int rank;
    char hostname[256];

    MPI_Init(&argc, &argv);
    gethostname(hostname, 256);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    printf("rank %d on %s says hello!\n", rank, hostname);
    MPI_Finalize();
    return 0;
}
-----
////////////////////////////////////

```

### 3.1 Build & Run

```
$ mpicc -o mpihello mpihello.c <1>
$ mpirun_rsh -hostfile hosts -n 2 ./mpihello <2>
```

1. mpicc is one of the basic commands used to compile MPI applications. This, along with mpicxx, mpif77, and mpif90, are wrapper scripts that invoke the compiler used to compile the MVAPICH2 library. Use of these scripts are recommended over invoking the compiler directly and adding the CFLAGS and LDFLAGS
2. mpirun\_rsh is used to launch MPI programs. This command tells mpirun\_rsh to launch 2 ./mpihello processes using the nodes specified in the hostfile *hosts*.

### 3.1.1 Using mpirun\_rsh

#### syntax

```
mpirun_rsh <options> <env variables> <command>
```

#### options

**-hostfile** specify the location of the hostfile

**Hostfile Format** The mpirun\_rsh hostfile format allows for users to specify hostnames, one per line, optionally with a multiplier, and HCA specification. The multiplier allows you to save typing by allowing you to specify blocked distribution of MPI ranks using one line per hostname. The HCA specification allows you to force an MPI rank to use a particular HCA.

The optional components are delimited by a ':'. Comments and empty lines are also allowed. Comments start with '#' and continue to the next newline.

The following demonstrates the distribution of MPI ranks when using different hostfiles:

#### Examples:

```
hosts1 node1  
node2
```

```
hosts2 node1  
node1  
node2  
node2
```

```
hosts3 node1:2  
node2:2
```

```
Output of mpihello with different hostfiles $ mpirun_rsh -hostfile hosts1 -n 4 ./mpihello  
rank 0 on node1 says hello!  
rank 1 on node2 says hello!  
rank 2 on node1 says hello!  
rank 3 on node2 says hello!
```

```
$ mpirun_rsh -hostfile hosts2 -n 4 ./mpihello  
rank 0 on node1 says hello!  
rank 1 on node1 says hello!  
rank 2 on node2 says hello!  
rank 3 on node2 says hello!
```

```
$ mpirun_rsh -hostfile hosts3 -n 4 ./mpihello  
rank 0 on node1 says hello!  
rank 1 on node1 says hello!  
rank 2 on node2 says hello!  
rank 3 on node2 says hello!
```

**-n** Number of mpi processes to launch.

**-ppn** Number of mpi processes to launch per node.

[IMPORTANT]

=====

The `-ppn` option will create a block of N processes on each node in the hostfile. This is analogous to using the `:#` syntax in the hostfile. Using both of these capabilities to create a block ordering will be multiplicative. Ie: setting `node1:2` in the hostfile and `-ppn 2` on the command line will result in 4 processes being allocated to node1.

If you are using the SLURM resource manager, omitting a hostfile will result in `mpirun_rsh` using the `SLURM_JOB_HOSTLIST` environment variable to determine the hosts. It will distribute processes across all active nodes in the job according to the value set by `-ppn`.

**env variables** Environment variables are specified using the `'NAME=VALUE'` syntax directly before the command name is specified. These are used to export `MPICH_CVAR` values to control underlying MPICH functionality as well as `MV2_CVARS` to control MVAPICH2 specific functionality.

Pass an environment variable named `FOO` with the value `BAR`

```
$ mpirun_rsh -hostfile hosts -n 2 FOO=BAR ./mpihello
```

### 3.1.2 Other Launchers

By default MVAPICH2 is built with `mpirun_rsh` and the MPICH Hydra process manager. Hydra can be invoked using the `mpiexec` binary in a standard install.

To configure with SLURM's `srun` launcher as your launcher, please use the following configuration:

```
$ ./configure --with-pm=slurm --with-pmi=pmi1
```

Or, if you are on a Cray cluster:

```
$ ./configure --with-pm=slurm --with-pmi=pmi2
```

If your SLURM or Cray `libpmi.so`, `pmi.h`, `libpmi2.so`, and `pmi2.h` files are on non-standard paths you may need to add the appropriate lib and include directories to `LD_LIBRARY_PATH`, `LIBRARY_PATH` and `CPATH` respectively.

Please look at our [User Guide](#) for more complete details.

## 4 More Information

Please see the following for more information.

- [User Guides](#)
- [OSU Micro-Benchmarks](#)
- [FAQ](#)