Tutorial 2: How to compile OpenFOAM® from source

by Xiaofeng Liu, Ph.D., P.E.
Associate Professor
Department of Civil and Environmental Engineering
Penn State University

1 Introduction and background

The website of OpenFOAM® is an excellent source for the information on installation, which is located at: https://develop.openfoam.com/Development/openfoam/-/blob/master/doc/Build.md.

So you should first refer to the installation page of their website. The purpose of this tutorial is to document/explain in more detail about how things are set up in OpenFOAM® . We will use the openfoam.com version. The `openfoam.org` version is slightly different (they were similar until OpenFOAM Foundation v11 released in 2022).

The installation of OpenFOAM® can be as simple as to install the pre-compiled binary package or to compile from source code on your computer. The choice depends on how you want to use it.

- If you are a user who only wants to run simulations using existing solvers and tools, then the installation of a pre-compiled binary package is enough. For example, the precompiled version for Linux: .
- If you want to do modifications to the source code, then you need to compile the code by yourself.
- On a given machine, there might be an OpenFOAM® installation at the system level. However, that installation might have issues with your code (it happens). To solve this problem, you can compile your own version in your own space. Multiple versions of OpenFOAM® can be installed.
- The pre-compiled package only contains the optimized ("`opt`") version of code. Thus, if you want a "`debug`" version, which gives more information for debugging and diagnostic purposes, then you need to compile from source and specify the compilation flag as debug.
- Just for fun! The sense of fulfillment after a first successful compilation is enormous. In addition, the compilation process helps understand how the code is organized.

Over time, OpenFOAM® has evolved from only Linux-Unix friendly to the support of multiple platforms (Mac OS and Windows). As of now, the support of multiple platforms is through either the Docker technology or Windows Subsystem for Linux (WSL). Previously, without Docker or WSL, one either has to use virtual machines on a host computer or cross-compile. Cross-compilation is very tedious and requires the patch of the original code.

This tutorial does not cover every possible ways of installation and compilation. Instead, we will introduce the old fashioned way of doing business, i.e., to compile on a Linux machine from source code.

We will use the ESI/OpenCFD v2412 version (https://develop.openfoam.com/Development/openfoam/-/blob/master/doc/Build.md).

If you prefer, you can also install the OpenFOAM Foundation versions https://openfoam.org/download/. The compilation process is similar. In fact, you can have multiple versions on one computer and you can specify which version to use (only one at a time though).

You will compile in your home directory, thus you do not need any superuser privilege. However, if you need to compile and install a system-wide version for others to use, you need to escalate your privilege level.

2 Downloading and unzip

If you have a GUI to access the Linux machine, then open a web browser and download the ESI OpenFOAM source code and Third-Party source code from https://dl.openfoam.com/source/latest/. Click on the two links to download the OpenFOAM® and "ThirdParty" tar balls:

- https://dl.openfoam.com/source/v2412/OpenFOAM-v2412.tgz
- https://dl.openfoam.com/source/v2412/ThirdParty-v2412.tgz

You need to decide where you want to install OpenFOAM. Typically, for your own use, `$HOME/OpenFOAM` (i.e. a directory OpenFOAM in the your home directory) is the place to be. Note that for Penn State ICDS Roar Collab, you need to install in your "work" directory, e.g., "/storage/home/abc123/work". If you have not done so, create an "OpenFOAM" directory:

```
$ cd   (note: this goes to your home directory)
$ mkdir OpenFOAM
$ cd OpenFOAM
```

After downloading, move the two downloaded files into your "OpenFOAM" directory (if they are not there yet) and unzip the *.gz files by

```
$ tar -xzvf OpenFOAM-v2412.tgz
$ tar -xzvf ThirdParty-v2412.tgz
$ ls
```

Now you will see there are two directories. The `OpenFOAM-v2412` directory contains all the source code and tutorial cases. The `ThirdParty-v2412` directory contains all the third party things, e.g., MPI, Gcc, and scotch, which are used by OpenFOAM® .

If you do not have a windows GUI (e.g., when you SSH to a remote Linux machine and thus only have a terminal), you can use an alternative way by using the `wget` command:

```
wget https://dl.openfoam.com/source/v2412/OpenFOAM-v2412.tgz
wget https://dl.openfoam.com/source/v2412/ThirdParty-v2412.tgz
```

3 Installation

3.1 Install Compilers

The source code in OpenFOAM is written in C++, and thus compiling the code requires a C++ compiler. If parallel computation is needed, a MPI library is required.

Commonly used compilers include GNU C++ Compiler and Intel C ++ Compiler. The GNU C++ compiler is free and installed on most Linux HPC systems. The Intel C++ compiler is also free.

In order to check if a compiler has been installed in your system, using either of the following commands:

```
$ gcc --version or which gcc
```

or

```
$ icc --version or which icc
```

If you SSH to the Penn State ICDS Collab Roar system, which uses "module" to manage and load its installed packages, you can use the following commands to load or check compilers:

```
$ module avail (note: show all available packages)
$ module load gcc/5.3.1 (note: load a specific package)
$ module list (note: show all software already loaded)
```

For MPI, use the following command to check if the MPI compiler is available:

```
$ which mpicc
```

The command `mpicc` comes with most MPI libraries and it will be used for the compilation of the parallel computation part of the code. If you are using the Penn State ICS ACI system, the MPI package can be directly loaded through the following commands:

```
$ module load openmpi/4.1.1-pmi2
```

if you use the Gcc; or

```
$ module load mpich/3.2
```

if you use Intel C++ compiler.

Generally, MPI is not installed in a desktop Linux system (e.g., your office computer or your personal laptop). So you will likely need to install by yourself. In this case, you have to download and compile a MPI library. We usually use the free OpenMPI by GNU. Different version of OpenMPI can be downloaded from its website. For example, for our purpose, you can go to:
https://www.open-mpi.org/software/ompi/v4.1/.

By default, OpenFOAM® v2412 uses the system default MPI library. In the "ThirdParty" directory, it also comes with an OpenMPI v4.1.2, which can be compiled for use if desired.

3.2 Modify default settings to adapt to your system

OpenFOAM® has some default settings for C++ compiler, MPI, and other things. You may need to change these settings to reflect what you have on a specific computer. In addition, we typically want all the environmental variables set properly whenever we open a new terminal. This can be done by adding content into the `.bashrc` file in your home directory, which will be automatically loaded when a new terminal is open. You can edit the `.bashrc` file with any text editor of your choice:

```
$ cd   (note: this will go to your home directory)
$ gedit .bashrc (or vim .bashrc or vi .bashrc)
```

For example, on Penn State ICDS Roar Collab, I add the following to the end of the `~/.bashrc` file:

```
# module load for of2412
module load paraview
module load cmake/3.26.3
module load boost/1.77.0
module load fftw/3.3.10
module load openmpi/4.1.1-pmi2
```

```
# define alias
alias of2412='source /storage/home/abc123/work/OpenFOAM/OpenFOAM-v2412/etc/bashrc'
```

This defines a Linux command alias "of2412". When you execute this alias, it runs its definition, which "source" the OpenFOAM "bashrc" file, the most important file for setting proper environmental variables for OpenFOAM® . The alias assumes that your OpenFOAM® v2412 is installed in a directory called `work/OpenFOAM` in your home directory. Otherwise, you need change accordingly. After the editing, save the " /.bashrc" file and exit. Now you can refresh the Linux environment by `sourcing` (loading) that file:

```
$ source ~/.bashrc
$ of2412
```

You can check the effect of this `sourcing` by looking at the environmental variables:

```
$ export
```

You should see lots of environmental variables with `FOAM` in their names. This is because you just ran the "of2412" alias command.

Now we come back to the OpenFOAM® `bashrc` file where everything is set up. Among them, this file specifies things like what C++ compiler and what MPI you want to use. For most cases, you need to modify two lines:

```
export WM_COMPILER=?     (note: for C++ compiler)
export WM_MPIIB=?  (note: for MPI)
```

In order to determine the correct C++ compiler and MPI, you should open the *compliler* file in `~/OpenFOAM/OpenFOAM-v2412/etc/config.sh/compiler`. There, you will find the meaning of "Gcc", "Gcc45", "Gcc46", etc. Choose the correct compiler based on what you use on your computer. Or you can specify the following (default):

```
export WM_COMPILER_TYPE=system
export WM_COMPILER=Gcc
```

to use the system default Gcc compiler.

For the MPI, we usually use two options: *SYSTEMOPENMPI* or *OPENMPI*. The former option means that you want to use the system OpenMPI and the second means that you want to use the user-specific OpenMPI in your `ThirdParty-v2412` directory.

- For Penn State ICDS ACI computers, if you load the MPI through the `load module`, you should use the `SYSTEMOPENMPI` option. This is because *SYSTEMOPENMPI* is defined in the `~/OpenFOAM/OpenFOAM-v2412/etc/config.sh/mpi` file. In this file, you will find that the `SYSTEMOPENMPI` option points to the MPI library searched by the `mpicc` path.

- If there is no system MPI installed or you want to use your own OpenMPI, you should use the `OPENMPI` option. However, your computer still does not know where the "OpenMPI" is located. Thus you should specify this location which can be done again through the file `~/OpenFOAM/OpenFOAM-v2412/etc/config.sh/mpi`. In this file, find the line `export FOAM_MPI=openmpi-version`. Replace the "version" with the version number that you have on your system, e.g., `4.1.2`. OpenFOAM® will look into the `ThirdParty-v2412` directory and try to find the specified OpenMPI version there.

  Therefore, if you already downloaded and compiled OpenMPI in the "ThirdParty" directory, and set the correct version number in the `mpi` file, all the environmental variables should be set correctly such that OpenFOAM® can find the MPI library.

3.3 Other possible modifications

Generally, when you set the compiler and MPI correctly, you are ready to compile the whole code. However, in some situations, you may need to further modify the `bashrc` file.

- Sometimes, you may want to directly specify where to install OpenFOAM® . You can set this in the OpenFOAM® `etc/bashrc` file:

  ```
  export FOAM_INST_DIR=Your Install dir/$WM_PROJECT
  ```

3.4 Compiling

After you set up everything above (and don't forget to source your `.bashrc` file every time you make changes in the settings):

```
$ of2412
```

now you are ready to compile OpenFOAM® . Depending on your computer, the compilation will take some time. The compilation is through:

```
$ foam   (note: this ‘‘alias’’ command takes you to the \OpenFOAM directory)
$ ./Allwamke
```

Optionally, you can use parallel compilation, i.e., compile multiple files at the same time. For example, if you want to use all available cores, you can do:

```
$ ./Allwamke -j
```

If during the compilation errors occur, you should inspect the error message and make adjustment to the setting files accordingly, and then run the compilation command above again.

You can also remove the compiled version by

```
$ wclean all
```

which deletes the compiled files (but not the source code and cases).

3.5 Test the compilation result

After the successful compilation, the first thing to do is to test whether the compilation and installation are correct. The following is a small test to run a simple case.

Create a directory named `run` in your OpenFOAM® user space

```
$ mkdir -p $FOAM_RUN  (note: $FOAM_RUN is defined with your user name)
```

Copy a tutorial case to your own `run` directory. It is suggested that you don't run the tutorials in OpenFOAM® 's `tutorials` directory.

```
$ cd $FOAM_RUN
$ cp -r $FOAM_TUTORIALS/incompressible/simpleFoam/pitzDaily .
$ cd pitzDaily
$ blockMesh
$ simpleFoam
```

After running the solver on this tutorial case, you can use `ParaView` to visualize the result.

Appendix A A quick guide for ICDS Roar Collab

A.1 How to compile OpenFOAM on Roar Collab?

The compilation of OpenFOAM on Penn State ICDS Roar Collab is similar to the procedure outlined above, with some subtle changes. This section provides a quick guide if you want to use ICDS Roar Collab for this course. To start, you should request a "Roar Collab RHEL8 Interactive Desktop" as I have demonstrated in class. Once the interactive desktop is running, you should have be able to use the allocated Linux computer resources (see Figure A.1).

Open a command terminal, go to the "work" directory, and then create a directory named "OpenFOAM":

```
$ cd work
$ mkdir OpenFOAM
$ cd OpenFOAM
$ pwd     (note: make sure you are in the OpenFOAM directory)
```

Download the OpenFOAM source code. Here, we use the "wget" command:

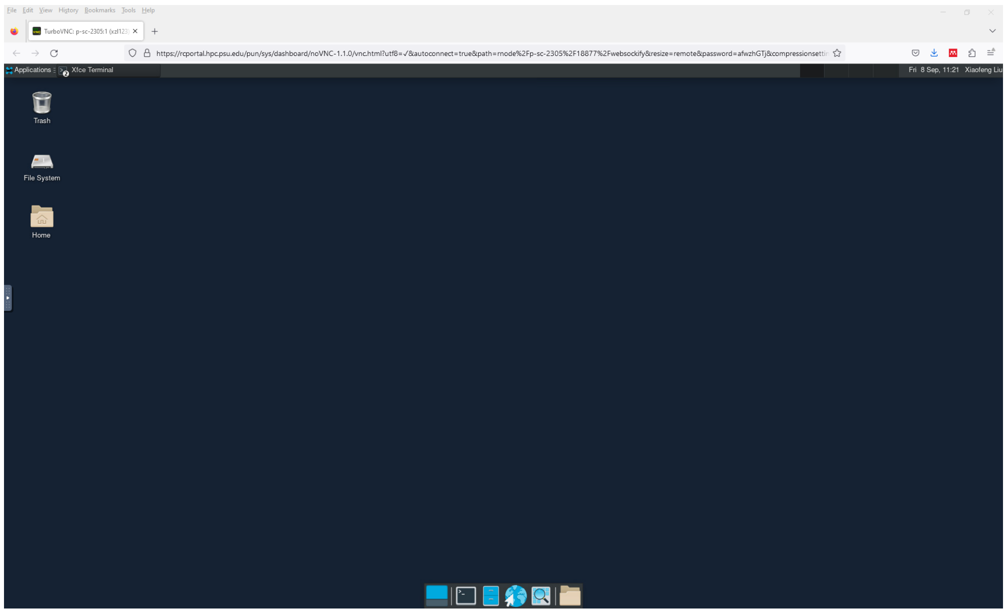

Figure A.1: Roar Collab Interactive Desktop

```
$ wget https://dl.openfoam.com/source/v2412/OpenFOAM-v2412.tgz
$ wget https://dl.openfoam.com/source/v2412/ThirdParty-v2412.tgz
$ ls  (note: make sure you see the two downloaded tgz files)
```

Unpack the two tgz files:

```
$ tar -xzvf OpenFOAM-v2412.tgz
$ tar -xzvf ThirdParty-v2412.tgz
$ ls
```

Modify OpenFOAM's "etc/bashrc" file to ensure the directories are specified correctly. By default, OpenFOAM assumes you install within the directory "$HOME/OpenFOAM". However, on ICDS Roar Collab, we need to install in "$HOME/work/OpenFOAM". Thus, you need to change the "etc/bashrc" file. You can search within this file for "HOME" and modify accordingly. For example, we need to change

```
projectDir="$HOME/OpenFOAM/OpenFOAM-$WM_PROJECT_VERSION"
```

to

```
projectDir="$HOME/work/OpenFOAM/OpenFOAM-$WM_PROJECT_VERSION"
```

To modify the "etc/bashrc" with a text editor of your choice:

```
$ cd OpenFOAM-v2412
$ vi etc/bashrc  (note: Here I use vi to edit this file. You can use whatever editor of
```

For convenience, define an alias “of2412” in your home directory's “.bashrc” file by adding the following to the end of it (note: replace “your_user_id” with your user ID):

```
alias of2412='source /storage/home/your_user_id/work/OpenFOAM/OpenFOAM-v2412/etc/bashrc'
```

Run this alias to execute OpenFOAM's “bashrc” file:

```
$ source ~/.bashrc   (note: this will execute the ``.bashrc'' file in your
                      home directory where you defined the of2412 alias)
$ of2412    (note: run the of2412 alias command)
$ export    (note: check all the OpenFOAM environmental variables)
$ foam      (note: ``foam'' is an OpenFOAM alias, which takes you to OpenFOAM directory)
```

Load dependency modules. OpenFOAM depends on some external libraries and tools, such as MPI for parallel computing, cmake, boost, fftw, and ParaView (for visualization of simulation results). These libraries are taken care of within “ThirdParty-v2412”. However, since ICDS Roar Collab already have these installed as modules, we can simply just load them:

```
$ module load paraview
$ module load openmpi/4.1.1-pmi2
$ module load cmake/3.26.3
$ module load boost/1.81.1
$ module load fftw/3.3.10
```

Compile OpenFOAM:

```
$ pwd   (note: make sure you are within the OpenFOAM-v2412 directory)
$ ls
$ ./Allwmake -j -s -q -l
```

Post-compilation steps: Please refer to OpenFOAM's build instruction:

https://develop.openfoam.com/Development/openfoam/-/blob/master/doc/Build.md#post-compilation-steps

A.2 How to submit parallel jobs on Roar Collab?

ICDS Roar Collab uses the “Slurm” workload manager system to manage job scheduling on Linux clusters (https://slurm.schedmd.com/sbatch.html). To submit parallel jobs with “Slurm”, you should use its “sbatch” command to submit a job script to the scheduler. An example job script, “`of_slurm.sh`”,has been provided and you can modify according to your need.

Within a “Slurm” script, we use the “Slurm directives” to specify the job. The following is an example list with explanations (you should modify to suit your needs):

- **#SBATCH –nodes=2**: Allocates 2 nodes.
- **#SBATCH –ntasks-per-node=20**: Sets 20 tasks per node.
- **#SBATCH –mem-per-cpu=12G**: Requests 12GB memory per CPU core.
- **#SBATCH –time=02:00:00**: Sets a wall time limit of 2 hours.
- **#SBATCH –job-name=icoFoam_cavity**: Assigns a name to the job.
- **#SBATCH –partition=open**: Use the open queue.
- **#SBATCH –output** and **#SBATCH –error**: Define the paths for the output and error files respectively.

"Slurm" sets various environment variables for jobs, a few of which are used in the script:

- **SLURM_NNODES**: The number of nodes allocated to the job.
- **SLURM_SUBMIT_HOST**: The host (node name) from which the job was submitted.
- **SLURM_JOB_PARTITION**: The partition on which the job runs (open in this case).
- **SLURM_JOB_NODELIST**: List of nodes assigned to the job.
- **SLURM_SUBMIT_DIR**: The directory from which the job was submitted.
- **SLURM_JOB_ID**: The job id.
- **SLURM_JOB_NAME**: The job name.

Within the job script, we also import the OpenFOAM dependency modules and execute OpenFOAM's "bashrc" file to properly set up the computing environment in the allocated computers:

```
module load openmpi/4.1.1-pmi2
module load cmake/3.26.3
module load boost/1.81.1
module load fftw/3.3.10
```

Documentation on ICDS's website:

https://www.icds.psu.edu/running-batch-jobs-on-roar-collab/

Please also note that if you use ICDS's "open" queue, there are some limitations in terms of how many cores and how many hours you can request. See the details about the "open" queue here:

https://www.icds.psu.edu/computing-services/roar-user-guide/free-roar-access-versus-a-pa

To submit a job, put the provided script in your OpenFOAM case directory and execute:

```
$ sbatch of_slrum.sh
```

To check the status of your submitted jobs, execute the "`squeue`" command (https://slurm.schedmd.com/squeue.html):

```
$ squeue -u user_id  (Note: replace user_id with your user ID)
```