TOSS4's Development Environment Overview

John Gyllenhaal

LC User Meeting
Thursday June 29, 2023

LLNL-PRES-850792

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

Why couldn't we just keep using TOSS3 forever and avoid all this porting effort to TOSS4 and a new /usr/tce?

- Affordable support for RHEL7 is at an end (TOSS3 is based on RHEL7)
  - Imminent end to security patches from Red Hat driving current upgrade schedule
    - Incredibly expensive extended support doesn't make sense due to other upgrade drivers
  - Frozen "old" feature set of RHEL7 (especially in C++ and old bin utils) impacting many users
  - Layers of workarounds needed in TOSS3 for some complex codes to function (such as -stdlib=libc++)
- Newer machines (like CTS2 and El Cap) not supported by TOSS3/RHEL7
  - Big and powerful new machines this year and coming later must run TOSS4 (or later)
  - Having mix of machines at TOSS3 and TOSS4 creates large code maintainer burdens
    - Older machines that do not support TOSS4 are being retired instead of upgraded
- Many /usr/tce enhancement requests could not be done under TOSS3
  - We strive to limit 'breaking' changes to TOSS transitions and possibly one mid-cycle update
  - This was one of the largest changes to /usr/tce done with a TOSS update
  - In general, all code should and must be recompiled under TOSS4
    - There are very hacky ways to make TOSS3 executables run on TOSS4 for debugging and comparisons

Additional design goals for user environment on TOSS4

- Make a more standard user environment that is similar to environments at other sites
  - Make LC ”magic” optional and explicit (-magic) instead of implicitly forced on every user
  - Make design more compatible with Spack and modern build tools
- Make compatible and complementary with Cray’s user environment on El Cap/EAS3
  - On older IBM’s, we completely hid IBM’s environment behind LC ”magic” wrappers
    - Few other IBM sites limited impact of completely different user environment on our machines
  - Allow users to use ”pure” Cray environment on El Cap and EAS3, if they so desire
- Install /usr/tce with Spack instead of custom rpm packages
  - Make installed /usr/tce packages reproducible at other sites
  - Leverage Spack community’s efforts for installing complex software
- Enable a common environment deployment at the tri-labs
  - Work in progress: Possibly initially deployed as a supplement to our current environment

What's new in TOSS4 user environment

- Default gcc is RedHat's 8.5.0 ABI-compatible port of gcc 10.3.1
  - Robust support of up to C++17 standard
  - Can mix base gcc 8.5.0 with RedHat's ABI compatible gcc 10, 11, and 12 (not fully baked, C++20 experimental)
  - Note: Recent CUDA versions trigger latent gcc bug that is fixed in gcc 11 or later
- Much faster and dramatically more accurate gnu math library (-lm)
  - Very likely differences in end bits of calculations on TOSS4 compared to TOSS3
    - MPI shared-memory optimizations on TOSS4 also can contribute to differences
  - Problematic TOSS3-corner-cases math calls no longer run millions of times slower than expected
- Packages renamed to match current vendor standard names and/or Spack names
  - **intel-classic** for old non-llvm Intel compilers and **intel** for new llvm-based compilers
  - But some Spack package names overly verbose, so we shortened them (vtune instead of intel-oneapi-vtune)
- Newer versions of many RHEL8-provided system libraries, and some incompatible with TOSS3 versions
  - libreadline.so.6 and libgfortran.so.3 version change commonly encountered when using TOSS3 libraries
  - Need to recompile everything for TOSS4, including shared libraries to avoid "library not found" issues
- Generic 'python' name no longer available, must explicitly pick **python2** or **python3**
  - See https://hpc.llnl.gov/software/development-environment-software/python, includes info on Conda, etc.

TOSS4 focus on Intel/Clang/Gcc with Mvapich2/Openmpi

- Run ‘ml avail’ or ‘module avail’ to list available modules (‘man ml’ for details)
  - LC uses lmod module hierarchies, so you must have a compiler loaded to see available MPI modules
  - TOSS4 default similar to TOSS3 env but newer versions ‘**intel-classic/2021.6.0-magic mvapich2/2.3.7**’
  - New with TOSS4: we now provide both portable and ‘-magic’ versions of dev environment
  - Example of some modules available on TOSS4 on 6/29/23 with defaults loaded:

```
---------------------- /usr/tce/modulefiles/MPI/intel-classic/2021.6.0-magic/mvapich2/2.3.7 ----------------------
   hdf5-parallel/1.14.0       netcdf-c-parallel/4.9.0         netcdf-fortran-parallel/4.6.0
   hdfview-parallel/3.3.0     netcdf-cxx4-parallel/4.3.1      parallel-netcdf/1.12.3

--------------------------- /usr/tce/modulefiles/Compiler/intel-classic/2021.6.0-magic ---------------------------
   hdf5-serial/1.14.0         mvapich2/2.3.7          (L)    netcdf-fortran-serial/4.6.0
   hdfview-serial/3.3.0       netcdf-c-serial/4.9.0          openmpi/4.1.2
   mkl-interfaces/2022.1.0    netcdf-cxx4-serial/4.3.1       print-openmp-mapping/1.0

------------------------------------- /usr/tce/modulefiles/toolchains/Core -------------------------------------
   clang/14.0.6-magic (D)  gcc/10.3.1-magic   (D)  intel-classic/2021.6.0-magic (L,D)  intel/2022.1.0-magic    (D)
   clang/14.0.6            gcc/10.3.1              intel-classic/2021.6.0              intel/2022.1.0
```

Effects of using -magic versions (intel-classic/2021.6.0-magic) versus portable versions (intel-classic/2021.6.0) of compilers

- The -magic versions of compilers/MPI ignore environment and modules minimizes env changes
  - Modules do not modify LD_LIBRARY_PATH, instead compiler wrappers adds extra RPATHs to link line
    - Goal is to get same shared libraries app was built with no matter what modules loaded
  - Modules do not set CC, CXX, F77, FC, MPICH_CC, MPICH_CXX, MPICH_F77, MPICH_FC env variables
    - Setting CXX in default modules tends to break build system subshells, since may redefine what build system specified
  - MPI wrappers (mpicc, mpicxx, mpif90, etc) ignore MPICH_CC, MPICH_CXX, MPICH_F77, MPICH_FC
    - Full path to MPI wrapper determines exactly which compiler is used, good for reproducible builds but non-standard!
    - 'magic' MPI behavior determined solely by if compiler version has -magic or not
- Portable versions of compilers/MPI uses environment and modules may change environment
  - Modules set LD_LIBRARY_PATH and may rely on same modules being loaded at run time as compile time
    - Can make it complicated to get right if running different apps compiled with different modules
  - MPI wrappers (mpicc, mpicxx, mpif90, etc) use MPICH_CC, MPICH_CXX, MPICH_F77, MPICH_FC settings
    - Build system/Spack can specify exactly which compiler to use even if not in path. Many build systems expect this!
  - Users may need to manually add additional RPATHs to link line for consistent behavior
    - Many users end up using shortcut of hardcoding compiler and mpi modules into their dot files to workaround this issue

Mapping compilers and MPI from TOSS3 to TOSS4

- TOSS3's default gcc is 4.9.3 and TOSS4 uses gcc/10.3.1-magic for robust C++17 support
- TOSS3's 'intel/19.1.2' module maps to 'intel-classic/19.1.2-magic' on TOSS4
  - intel-classic is Intel's name for the older compilers that were the workhorse on TOSS3
  - Avoid intel-classic/19.0.4 since it does not support gcc/10.3.1 base (use gcc/8.5.0 instead if need 19.0.4)
- TOSS3's 'intel/oneapi.2022.1' module maps to 'intel/2022.1.0-magic' on TOSS4
  - This is Intel's new clang-base 'intel' compiler, but clang-derived Fortran compiler not yet stable
  - We recommend using very stable intel-classic instead, especially if you plan to compile any Fortran code
- TOSS3's 'mvapich2/2.3.6' module mostly maps to 'mvapich2/2.3.7' on TOSS4
  - Patched mvapich2 but with -magic compiler, currently ignores MPICH_CC, MPICH_CXX, etc
- TOSS4 Systems currently defaults to intel-classic/2021.6.0-magic mvapich2/2.3.7
  - Testing found "pure" mvapich2/2.3.6 buggy and mvapich2/2.3.7 more closely matches patched 2.3.6 on TOSS3

Burning Questions?