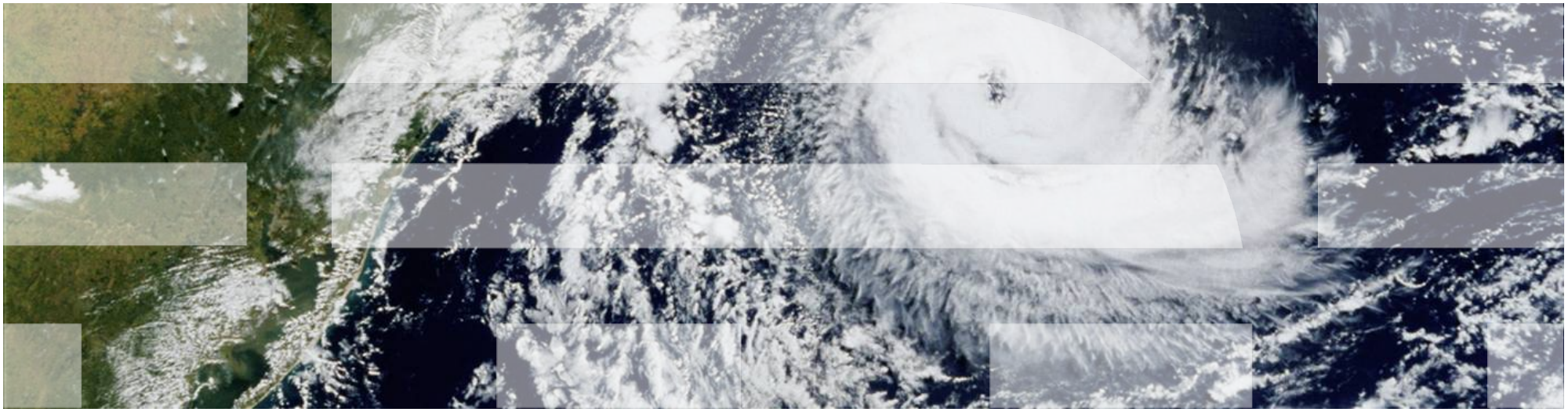
Programming Techniques for Supercomputers

(Heterogeneous and Distributed systems)

– Aditya Nitsure (IBM India)



Content

- OpenACC
- HPC Containers

OpenACC (Open Accelerators) (https://www.openacc.org)

Introduction

- Open GPU directive standard mainly designed for GPU programming
  - Developed by CAPS, Cray, Nvidia and PGI
  - Governed by OpenACC organization
  - Latest specification version 3.0
- Similar to OpenMP, the specification defines programming model for Fortran and C/C++ programming languages
  - compiler directives
  - library routines
  - environment variables
- Compatible with CUDA programming and libraries
- Supported by compiler vendors
  - Cray, PGI, GCC and few others

OpenACC

- Compiler directives
  - Specified with **pragma** preprocessing
  - Starts with **#pragma acc** (For C/C++)
  - Case sensitive
  - Any directive is followed by one or more clauses
  - Syntax #pragma acc directive-name [clause[[,] clause] ...] new-line
- Runtime library routines
  - **openacc.h** provides prototype definition of all library routines
- Environment variables
  - Set at the program start up
  - Always in upper case

Compilation of OpenACC program

PGI (pgcc)

- **GPU**

-acc

- -ta=tesla : <cuda version>
  e.g. -ta=tesla:cuda9.2
- -ta=multicore
  Generates code for CPU

More details can be found at
https://www.pgroup.com/resources/docs/19.1/x86/openacc-gs/index.htm

GNU (gcc)

- **GPU**

-fopenacc

Link with respective GPU library
(NVIDIA, AMD, Intel Xeon Phi)

More details can be found at
https://gcc.gnu.org/wiki/Offloading#Compilation_options

OpenACC Vs OpenMP

OpenACC

- Originally designed by keeping focus on GPU programming
- Supported on CPU
- OpenACC program more relies on compiler to exploit parallelism with efficient mapping to target

OpenMP

- Originally designed for CPU shared memory parallel programming
- Supported on GPU
- OpenMP puts all power and responsibility in the hands of programmer

OpenACC Vs OpenMP

OpenMP	OpenACC
#pragma omp parallel	#pragma acc kernels #pragma acc parallel
#pragma omp parallel for	#pragma acc parallel loop
Note : the kernels construct may be thought of as a hint to the compiler of where it should look for parallelism while the parallel directive is an assertion to the compiler of where there is parallelism.	
Implicit barrier at the end of parallel block / loop	Implicit barrier at the end of the parallel loop region
Avoid implicit barrier by using *nowait* clause	Avoid implicit barrier by using *async* clause
#pragma omp target data #pragma omp target enter data #pragma omp target exit data	#pragma acc data #pragma acc enter data #pragma acc exit data
map :: to, from, tofrom, alloc, delete	copyin, copyout, copy, create, delete
Same clauses	
shared, private, firstprivate, reduction, collapse	shared, private, firstprivate, reduction, collapse

Environment variables & Runtime library

	Environment Variable	Description
1	ACC_DEVICE_TYPE	Sets the *acc-current-device-type-var* ICV that specifies device type. Accepted values (case sensitive) • nvidia : NVIDIA GPUs • radeon : AMD GPUs • host : multi-core CPUs
2	ACC_DEVICE_NUM	sets the *acc-current-device-num-var* ICV that specifies device number
3	ACC_PROFLIB	Specifies profiling library

Runtime Library

- Device management routines – Get the number of devices, set the current device, and so on
- Asynchronous queue management – To synchronize all activities on anasync queue
- Device test routine – Test whether the statement is executing on the device or not
- Data and memory management – Manages memory allocation or copy data between memories.

References

- OpenACC programming and best practices https://www.openacc.org/sites/default/files/inline-files/OpenACC_Programming_Guide_0.pdf
- OpenACC API 3.0 https://www.openacc.org/sites/default/files/inline-images/Specification/OpenACC.3.0.pdf
- OpenACC and OpenMP comparison https://openmpcon.org/wp-content/uploads/openmpcon2015-james-beyer-comparison.pdf
- OpenACC tutorial http://www.hpcadvisorycouncil.com/events/2014/swiss-workshop/presos/Day_1/7_CSCS.pdf
- OpenACC examples https://www.olcf.ornl.gov/tutorials/openacc-vector-addition/

HPC Containers

HPC Container

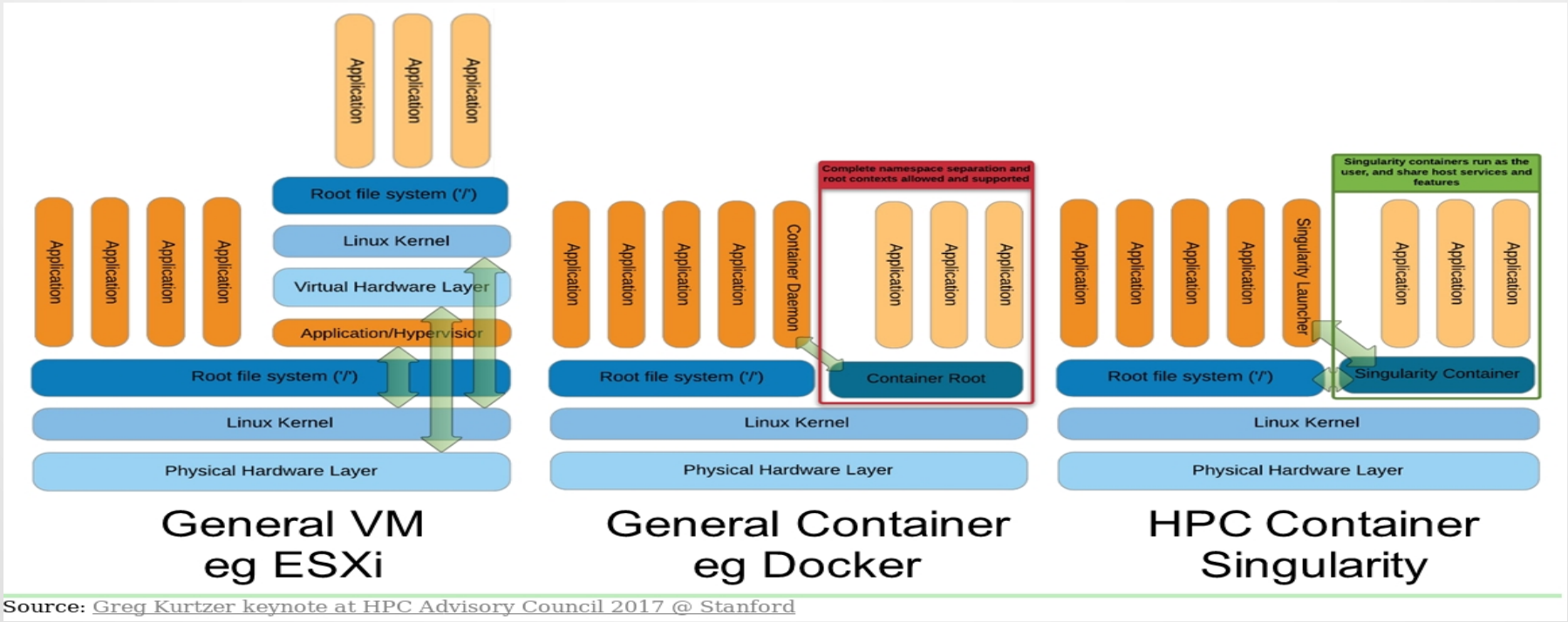
- Ease of development and deployment of HPC applications
  - HPC containers can package entire scientific workflows, software, libraries and even data
- Use cases
  - Customise environment
  - Commercially supported code requiring a particular environment
  - Static environments (software appliances)
  - Legacy code on old operating systems
  - Complicated software stacks and work-flows
    - Sometimes for some applications, moving to different version of same OS is challenging.
- Popular HPC Containers
  - Singularity
  - Shifter
  - CharlieCloud

Singularity

- Singularity enables users to have full control of their environment
- Build container
  - Singularity definition file or singularity hub
  - Docker file, dockerhub or docker container
- Supports MPI application (OpenMPI and MPICH)
- Presently supported only on Linux
- https://singularity.lbl.gov/
- https://sylabs.io/docs/

Docker Vs Singularity

- Docker container runs with different user id (root) than the process which launches it. Singularity maintains the user context
- Docker isolates filesystem i.e. by default anything running inside container don't have access to host filesystem



Source: Greg Kurtzer keynote at HPC Advisory Council 2017 @ Stanford

Reference

- Docker Vs Singularity Vs Shifter
  https://tin6150.github.io/psg/blogger_container_hpc.html
- Shifter container
  https://github.com/NERSC/shifter

Thank you !