GEOS-Chem on cloud computing platforms

-- experience with porting models to the cloud and engaging users

Jiawei Zhuang

07/30/2018

These public cloud platforms allow atmospheric scientists to

- Request computational resources on-demand, instead of maintaining local compute infrastructure
- Immediately access large volumes of public data, instead of spending weeks on data transfer
- Share pre-configured software environment, instead of building libraries and models from scratch

Many studies have tested atmospheric models on cloud platforms – but most scientists still don’t know how to use cloud for actual research

Author & year	Model	Platform
Evangelinos and Hill, 2008	MITgcm	AWS
Molthan et al. 2015	WRF	AWS
Withana et al., 2011	WRF	Azure
Siuta et al., 2016	WRF	Google Cloud
Chen et al. 2017	CESM	AWS
Li et al. 2017	GISS ModelE	AWS
Jun et al. 2017	ROMS	AWS

Our goal: Go beyond technical testing and allow GEOS-Chem users worldwide to actually use the cloud for real research

Achievements over the past year:

- Removed all dependency on proprietary software to facilitate cloud migration
- Gained support from the cloud vendor for hosting data
- Mitigated vendor lock-in by HPC containers
- Engaged the user community to actually use the cloud

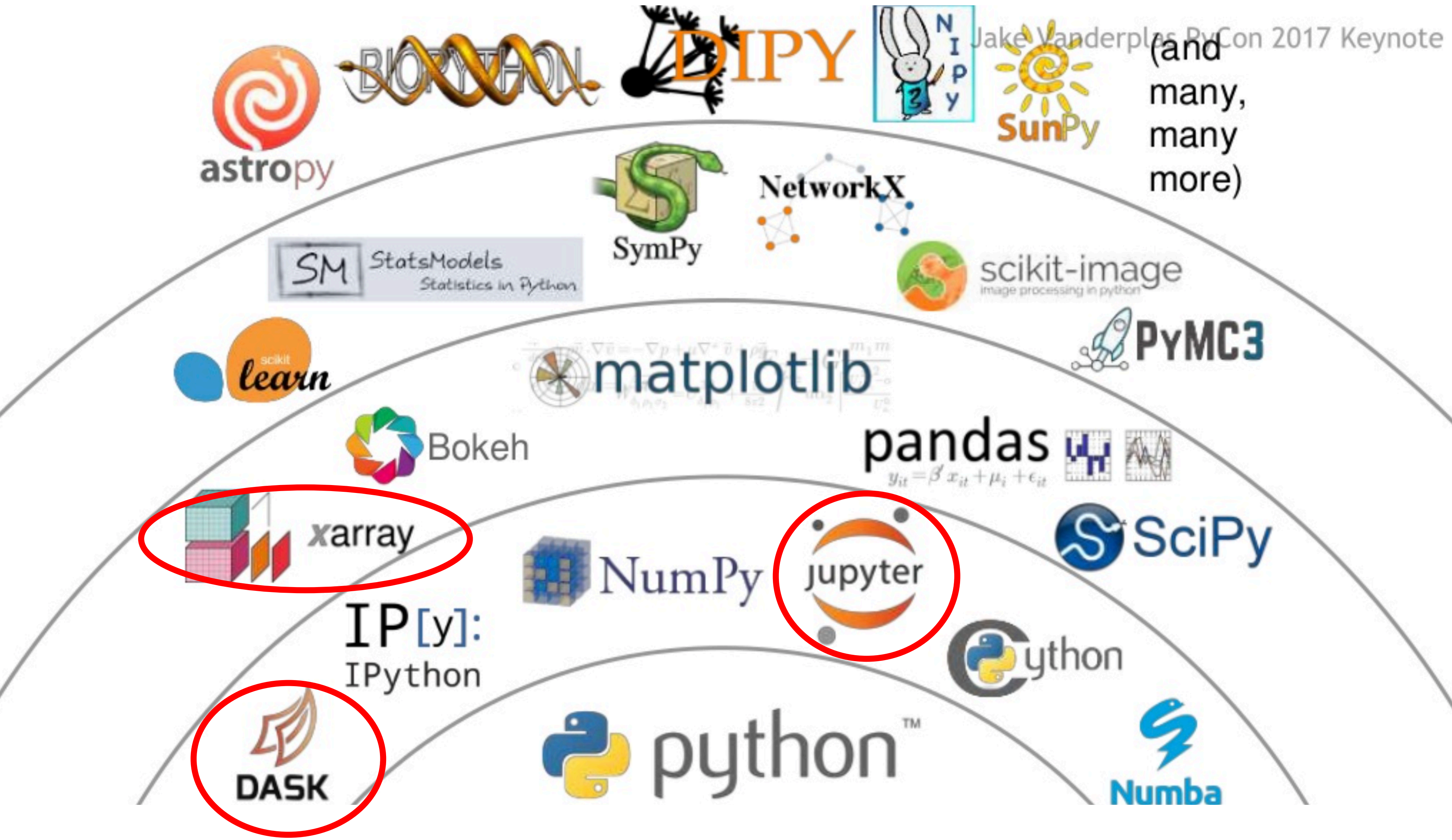
Step 1: Removed proprietary software dependency

- Commercial software can indeed run on the AWS cloud
  - Intel compiler: https://software.intel.com/en-us/articles/install-intel-parallel-studio-xe-on-amazon-web-services-aws
  - MATLAB: https://www.mathworks.com/cloud/aws.html
  - IDL: https://www.harrisgeospatial.com/Results.aspx?q=AWS

- But there are strong reasons not to use proprietary software:
  - Financial burden on scientists
  - Annoying licensing process
  - Cannot easily share system across users, which misses an important point of cloud computing
  - It hurts reproducibility

- For compiler, we refactored legacy code in GEOS-Chem so it is compatible with GNU Fortran compiler v4.x - v7.x. Tuned the compile settings to make it as fast as ifort.

Open-source scientific Python stack as a replacement of MATLAB & IDL



The standard SciPy stack can do almost everything except regridding. Thus I built my own solution:

https://github.com/JiaweiZhuang/xESMF (60+ GitHub stars)

xESMF: Universal Regridder for Geospatial Data

pypi package 0.1.1 | build passing | codecov 95% | docs passing | License MIT | DOI 10.5281/zenodo.1134366

xESMF is a Python package for regridding. It is

- **Powerful:** It uses ESMF/ESMPy as backend and can regrid between **general curvilinear grids** with all ESMF regridding algorithms, such as **bilinear**, **conservative** and **nearest neighbour**.
- **Easy-to-use:** It abstracts away ESMF's complicated infrastructure and provides a simple, high-level API, compatible with xarray as well as basic numpy arrays.
- **Fast:** It is faster than ESMPy's original Fortran regridding engine in serial case, and parallel capability will be added in the next version.

- Can process data from most models including WRF, CESM, GFDL-FV3, GEOS-Chem
- Will be able to leverage distributing computing power on cloud platforms, via xarray and dask
- Currently doesn't understand the irregular mesh in MPAS. (suggestions welcome)

Users can launch a pre-configured system and run the model immediately

An already-configured system that can run the model correctly

- A Linux operating system
- Fortran compilers
- NetCDF libraries
- MPI libraries
- Scientific Python environment
- Environment variable configuration

Save

Amazon Machine Image (AMI)

A frozen "system image" containing all information of my system

Launch

User 1's server

Guaranteed to run the model correctly without any further configurations

User 2's server

User 3's server

…

Step 2: Achieved agreement between Harvard and AWS to host 30 TB of GEOS-Chem data for free

Registry of Open Data on AWS

GEOS-Chem Input Data

climate | weather | meteorological | environmental | air quality | sustainability

Description

Input data for the GEOS-Chem Chemical Transport Model. Including the NASA/GMAO MERRA-2 and GEOS-FP meteorological products, the HEMCO emission inventories, and other small data such as model initial conditions.

Update Frequency

New meteorological and emission data will be added when available.

License

http://acmg.seas.harvard.edu/geos/geos_licensing.html

Documentation

http://cloud-gc.readthedocs.io

Contact

http://acmg.seas.harvard.edu/geos/

Usage Examples

Resources on AWS

Description
Top-level directory for all GEOS-Chem data.

Resource type
S3 Bucket

Amazon Resource Name (ARN)
`arn:aws:s3:::gcgrid`

AWS Region
`us-east-1`

https://registry.opendata.aws/geoschem-input-data/

Step 3: Mitigated vendor lock-in, by using containers to quickly replicate the same environment on multiple clouds and supercomputers

Containers for web apps

docker

kubernetes

Containers for HPC

Singularity

Charliecloud

SHIFTER

Step 4: Engaged people to actually use cloud

- **The question:** We have already built the system and uploaded all input data. But how to let scientists use it?
- **The difficulty:**
  - AWS online docs are written for web developers. Most contents are unreadable for scientists or even HPC programmers who do not have too much IT knowledge.
  - There are very few cloud computing textbooks targeted at scientific computing (currently only https://cloud4scieng.org)
- **Two possible solutions:**
  - ➢ Build high-level interface and hide the underlying cloud infrastructure (see next pages for the issues with this approach)
  - ➢ Educate users on how to use the cloud (our solution)

Overview of high-level cluster management tool

- There are many tools that emulate HPC clusters, including
  - CfnCluster (developed by AWS but not an official product)
  - StarCluster (the oldest tool but no longer maintained)
  - AlcesFlight (the most well-documented one right now)
  - ElastiCluster
  - EnginFrame
  - …

- They typically provide
  - Job schedulers (Slurm, SGE, PBS…)
  - "Auto-scaling" to automatically request and release nodes
  - Inter-node MPI connection
  - Some provide a GUI for cluster control

- Example: WRF on AWS with CfnCluster: https://github.com/aws-samples/aws-hpc-workshops/blob/master/README-WRF.rst

Issues with emulating a cluster environment

- They are leaky abstractions of many small AWS functionalities. Any debugging or customization will require the knowledge of underlying services.

- Research ≠ HPC. A lot of other research workloads will benefit from the understanding of cloud computing fundamentals:
  - Data analysis (e.g. after simulation) only need light-weight compute environment
  - Cloud is the go-to choice for machine learning (which is getting popular), but its parallel computation framework is different from traditional HPC/MPI (GPU computing, map-reduce type of distributed computing)

Our solution: teach users the minimum cloud fundamentals needed to get science done

- We are providing step-by-step, researcher-friendly documentations (http://cloud-gc.readthedocs.io). Mostly stick to low-level AWS concepts, so the skills are highly applicable to different workloads.

- GEOS-Chem users without prior cloud experience can go through a relatively complete workflow (run a short simulation, make plots, archive results) in an hour

- It can be easily adapted for other models like WRF and CESM, because the software requirement and research workflow for all atmospheric models are highly similar.

Remaining challenge and opportunities

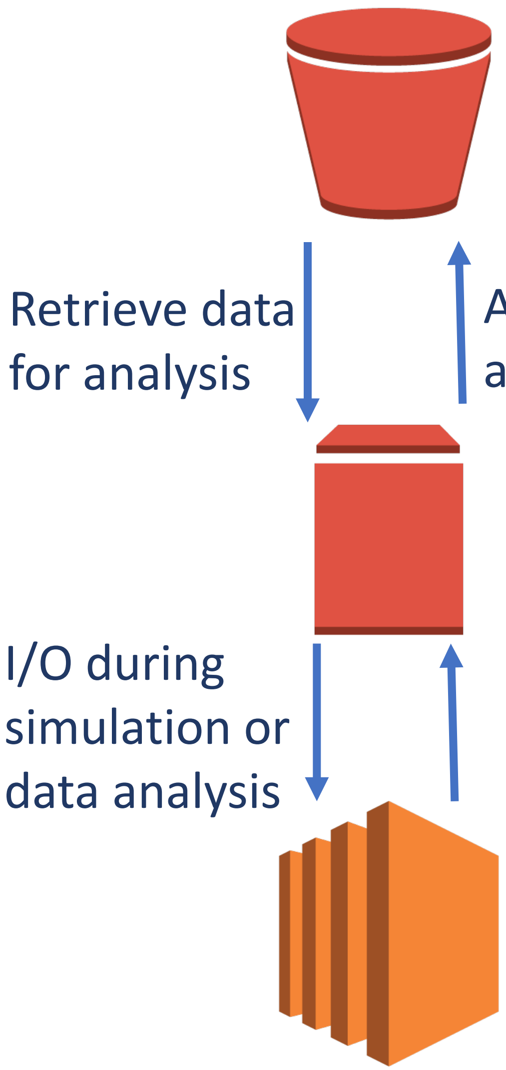
- **Data management workflow not optimal**
  - Need a “cloud-optimized” NetCDF to better utilize “object-based storage” in cloud

- **MPI cluster management is clumsy**
  - Single node works great, but user experience gets much worse when multiple nodes need to be managed.
  - Containers become less portable for cross-node MPI runs

- **Cost and funding of long-term, compute-intensive simulations**
  - The cloud is already great for light-weight experiments and for new users to learn a model.
  - Need coordination between universities, funding agencies and cloud vendors to better fund cloud computing resources.

Technical details on remaining issues (Backup slides)

Issue 1: Data management workflow not optimal
(The simplest workflow on AWS cloud shown below)



Cloud object storage (S3 bucket)

- Persistent, cheap, no size limit
- Not a standard file system (think about Dropbox)

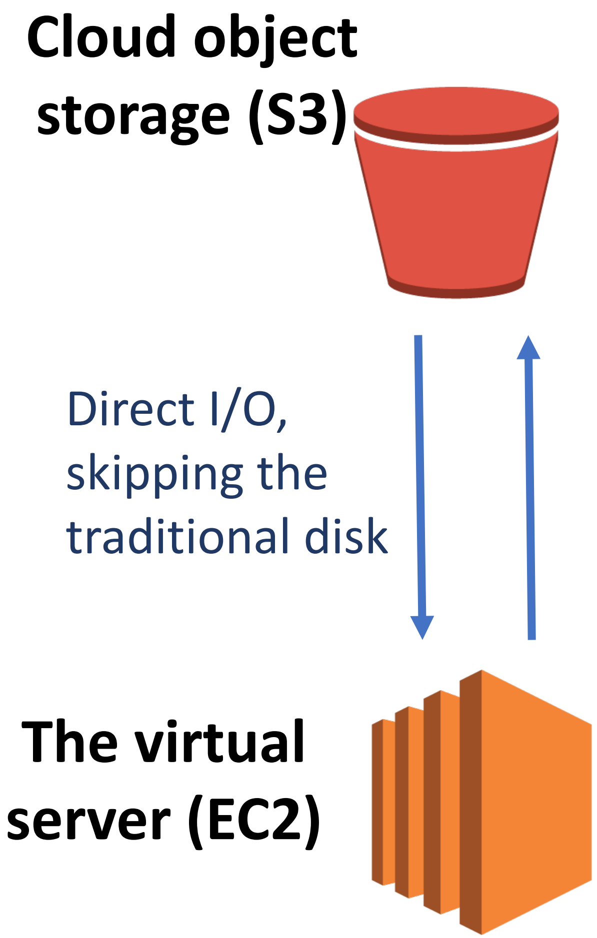
Traditional disk storage (EBS volume)

- Acts like a normal file system
- Ephemeral, more expensive, has size limit
- Not easy to share across servers

Virtual server (EC2 instance)

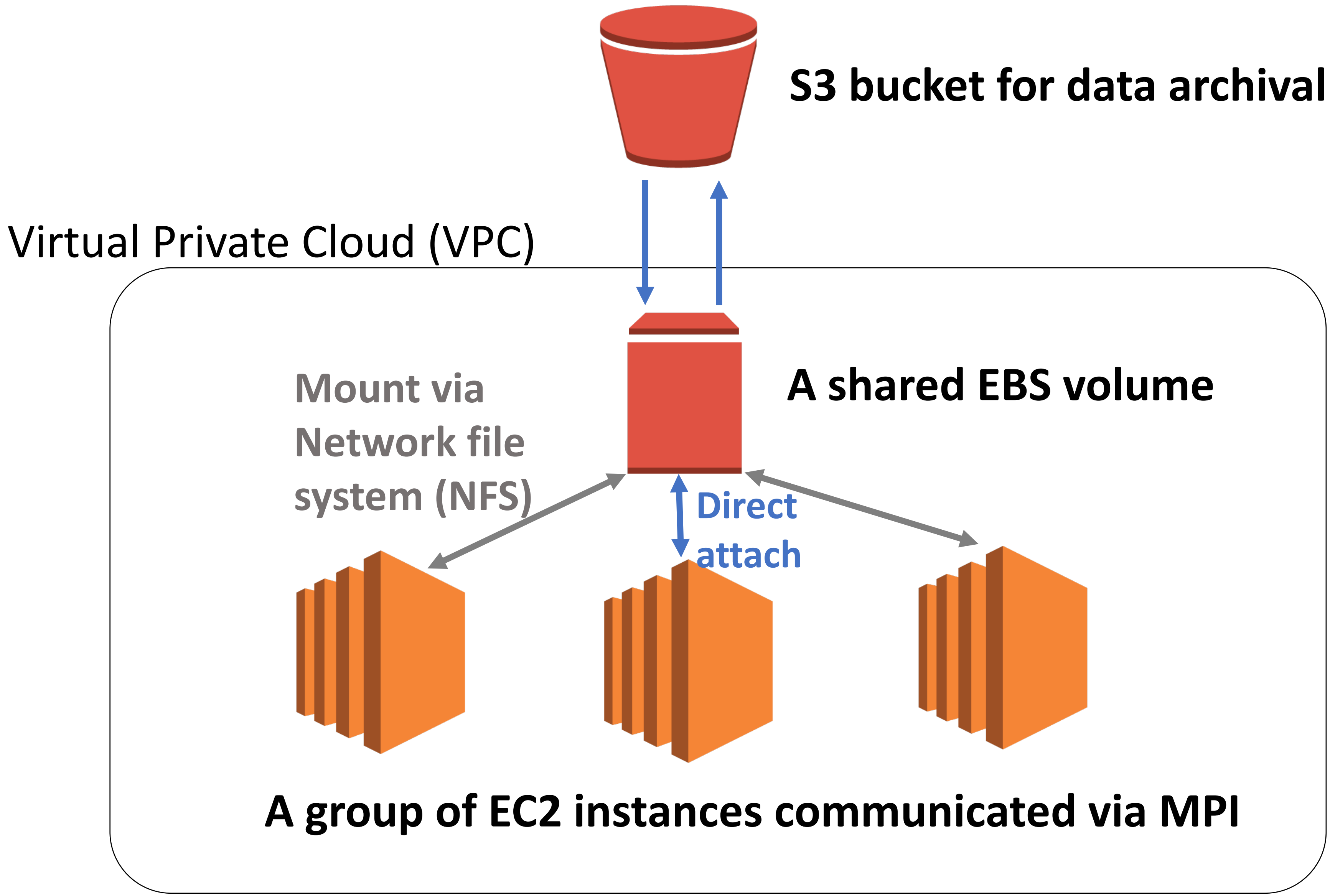
- Can spin up and down quickly
- Need additional storage for data persistence

Managing EBS volumes is awkward. Can we directly use object storage?



- Can directly mount S3 via FUSE (Filesystem in Userspace)
- Current performance is miserable (read http://matthewrocklin.com/blog/work/2018/02/06/hdf-in-the-cloud)
- Need a “cloud-optimized” NetCDF. See the Pangeo project for some workaround (http://pangeo-data.org)
- This framework might remove the need of a shared disk for MPI clusters (see the next slide)

Issue 2: Managing HPC clusters is clumsy (A minimum cluster architecture shown below)



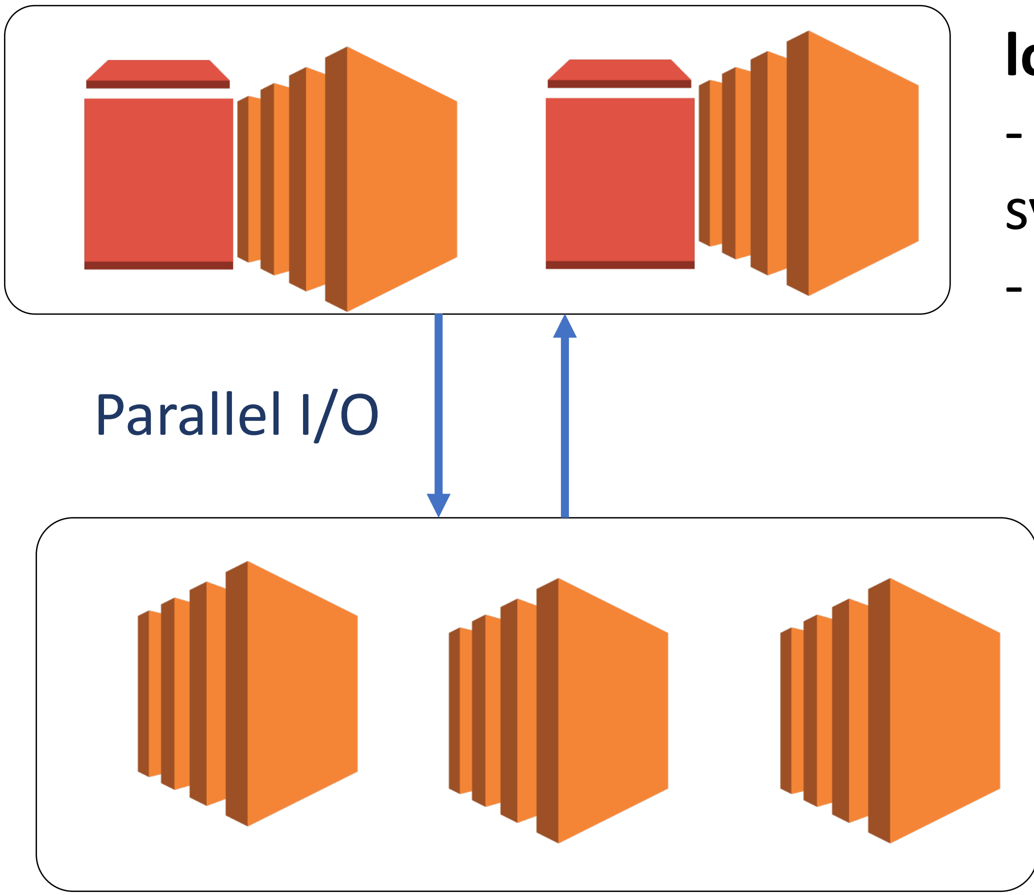
Notes on building a cluster on AWS cloud

- High-level cluster management tools (e.g. CfnCluster) are very heavy and hard to customize. For developers, setting up an MPI cluster manually is actually more painless. See steps at https://glennklockwood.blogspot.com/2013/04/quick-mpi-cluster-setup-on-amazon-ec2.html

- High-level tools might still be useful for users. Need to investigate “which is the best” & “do we actually need them”.

- Containers become less portable for cross-node MPI runs. Singularity uses host MPI which must be compatible the MPI inside container.

Can further build a dedicated parallel file system server, for I/O heavy simulations



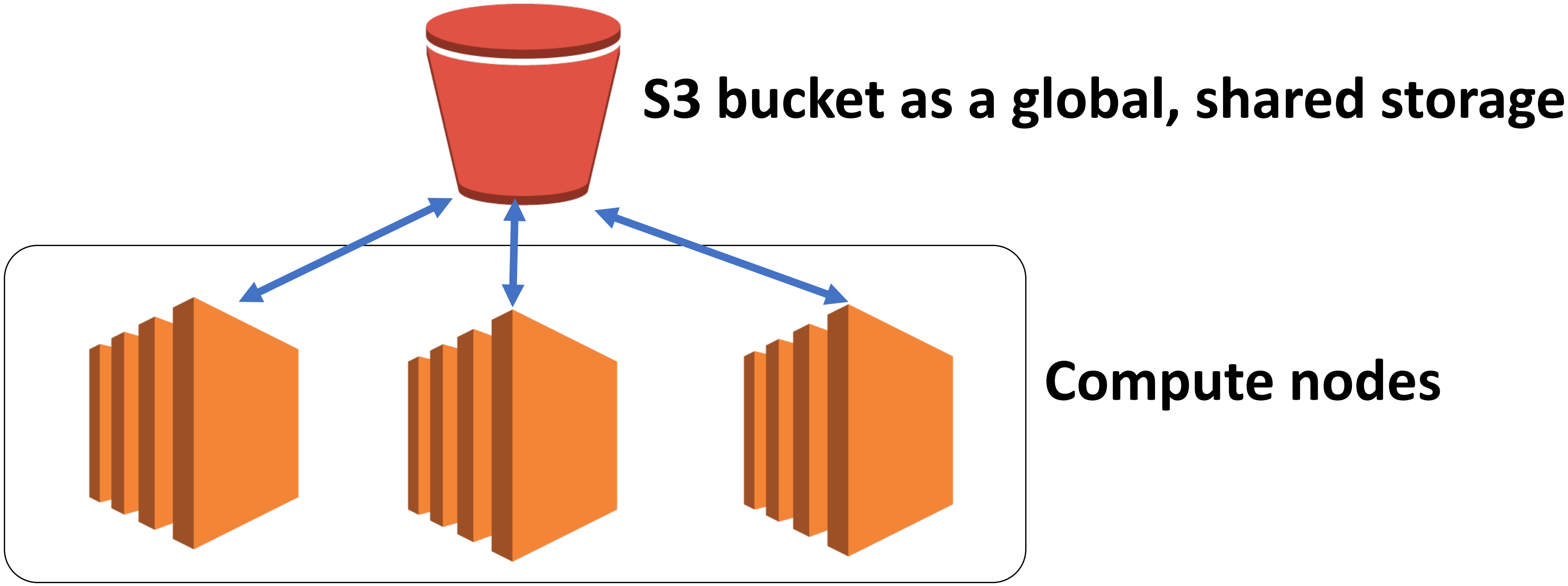
An array of EC2 servers with local storage

- Configured with parallel file system like Lustre and BeeGFS
- Emulate traditional HPC center

Compute nodes
(the file system's client)

Further reading: Enabling a parallel shared filesystem on an Alces Flight cluster

Is it possible to make the infrastructure simpler?



- No need to maintain and pay for dedicated disk storage and file server

- This is the architecture adopted by the Pangeo project (focus on data analysis, using dask and Kubernetes containers). Not sure if that can be adopted for HPC/MPI workloads

Issue 3: Cost and funding

- The cost of AWS cloud is as low as the hourly cost on NASA Pleaides cluster, if
  - Parallel scaling is not an issue (e.g. single node)
  - The spot instance pricing model is used

(Source: *Evaluating the Suitability of Commercial Clouds for NASA ' s High Performance Computing Applications : A Trade Study*, *NASA report, 2018*)

- Special funding programs exist, but far from enough
  - NSF BIGDATA program
  - AWS research credits
  - Azure "AI for Earth"
  - …

- Need coordination between universities, funding agencies and cloud vendors to better fund cloud computing resources. (e.g. University of Washington offers great support: https://itconnect.uw.edu/research/cloud-computing-for-research/)

Reference

Mehrotra, Piyush, et al. "Performance evaluation of Amazon EC2 for NASA HPC applications." *Proceedings of the 3rd workshop on Scientific Cloud Computing*. ACM, 2012.

Jung, K., Cho, Y.-K., and Tak, Y.-J.: Performance evaluation of ROMS v3.6 on a commercial cloud system, Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-270, in review, 2017.

Evangelinos, Constantinos, and Chris Hill. "Cloud computing for parallel scientific hpc applications: Feasibility of running coupled atmosphere-ocean climate models on amazons ec2." ratio 2.2.40 (2008): 2-34.

Molthan A L, Case J L, Venner J, et al. Clouds in the cloud: weather forecasts and applications within cloud computing environments[J]. Bulletin of the American Meteorological Society, 2015, 96(8): 1369-1379.

Chen X, Huang X, Jiao C, et al. Running climate model on a commercial cloud computing environment: A case study using Community Earth System Model (CESM) on Amazon AWS[J]. Computers & Geosciences, 2017, 98: 21-25.