Human in the Loop Virtual Machine Management on Comet

Gregor von Laszewski
laszewski@gmail.com

Fugang Wang, Geoffrey C. Fox, Shawn Strande, Christopher Irving, Trevor Cooper, Dmitry Mishin, Michael L. Norman

SDCS & Indiana University

Comet is supported by NSF grant: ACI \#1341698 Gateways to Discovery: Cyberinfrastructure for the Long Tail of Science.

What is Comet

- Target the long tail of science
- Focuses primarily on small and modest scale computing jobs,
- Those that require specialized software environments that are not found on traditional clusters.
- Science gateways
- Use Virtual Clusters (VCs) by leveraging existing batch queue, e.g. not OpenStack
- Near-bare metal performing computing resources
- Interactive experience as part of a human-in-the-loop management and usage strategy.

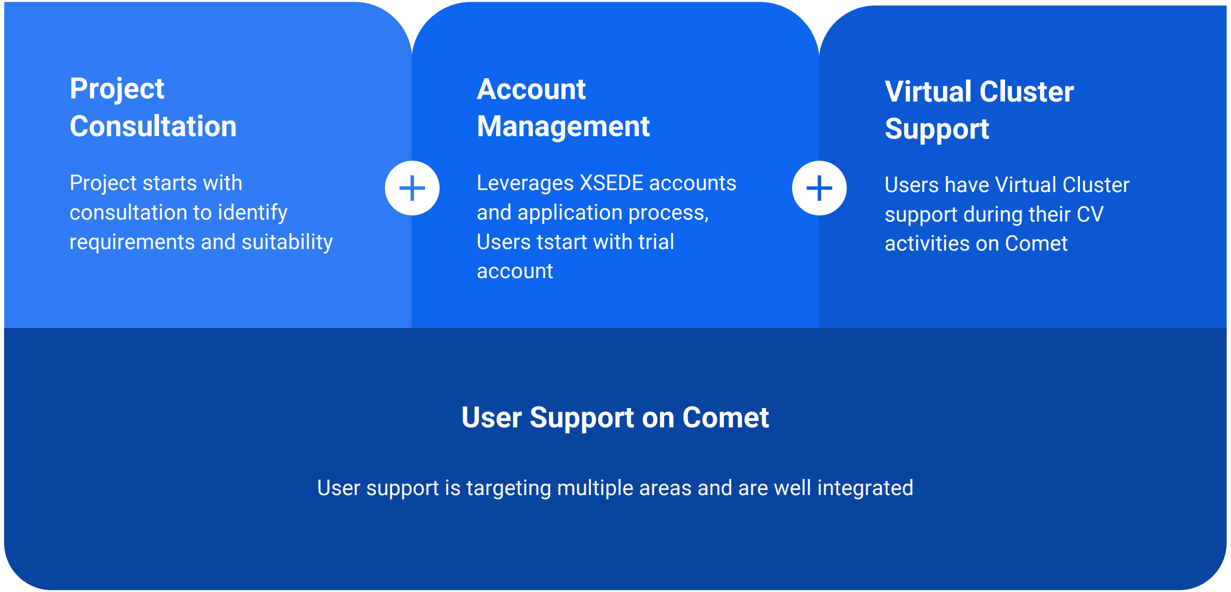
Hardware

- 2.76 Pflop/s peak; **48,784 CPU cores**; 288 NVIDIA GPUs; 247 TB total memory; 634 TB total flash memory
- Standard Compute Nodes (1944 total)
- Intel Xeon E5-2680v3 2.5 GHz dual socket, 12 cores/socket; 320 GB flash memory; 120 GB/s memory bandwidth
- **GPU Nodes** (72 total)
  - 36 K80 nodes: 2 NVIDIA K80 GPUs per node; dual socket, 12 cores/socket; 128 GB DDR4 DRAM; 120GB/s memory bandwidth; 320 GB flash memory
  - 36 P100 nodes: 4 NVIDIA P100 GPUs; dual socket, 14 cores/socket; 128 GB DDR4 DRAM; 150GB/s memory bandwidth; 400 GB flash memory
- Large-memory Nodes (4 total)
- **1.5 TB total memory**; 4 sockets, 16 cores/socket; 2.2 GHz
- Interconnect: Hybrid Fat-Tree topology; **56 Gb/s** (bidirectional) link bandwidth; **1.03-1.97 µs MPI latency**
- **7.6 PB Lustre-based Parallel File System**
- Access to Data Oasis
- High-performance virtualization

Virtual Clusters on Comet

- Focus is on giving user a cluster of virtual machines
- Performance is close to bare metal
- Utilizes Infiniband
- Users decide how many of the allocated virtual machines they like to use
- build into accounting and monitoring system of comet

User Support for Virtual Clusters



Project Consultation

Project starts with consultation to identify requirements and suitability

Account Management

Leverages XSEDE accounts and application process, Users tstart with trial account

Virtual Cluster Support

Users have Virtual Cluster support during their CV activities on Comet

User Support on Comet

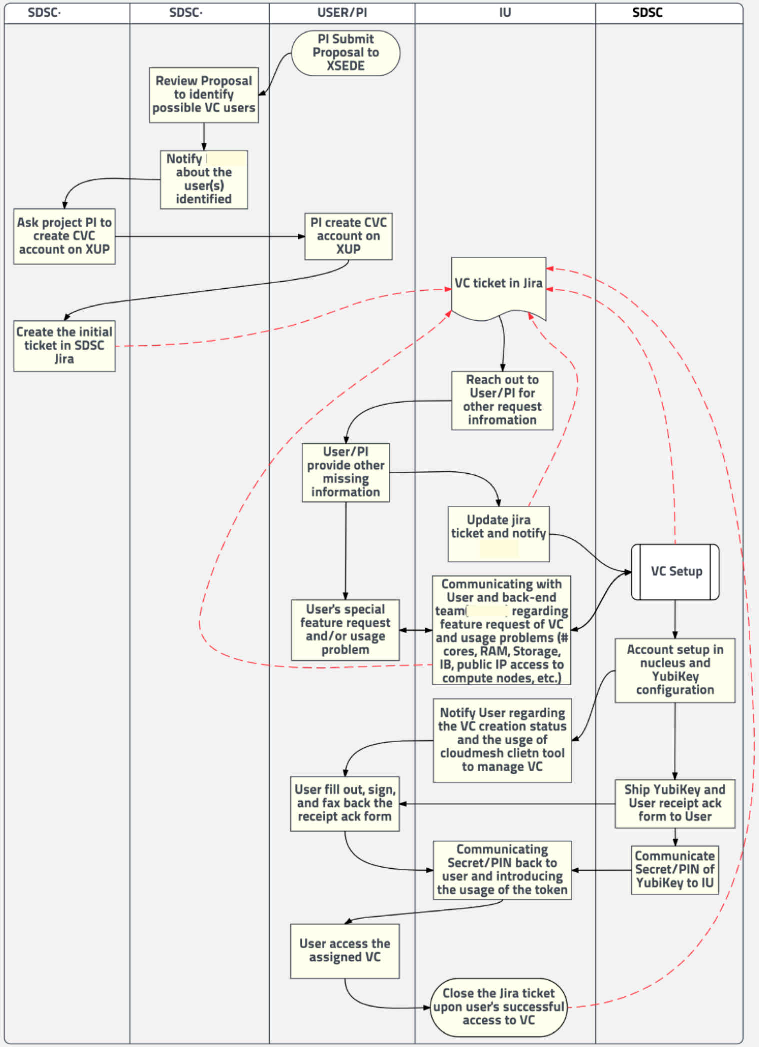
User support is targeting multiple areas and are well integrated

Project Consultation

- Is the project suitable for VC’s on comet?
- Is there enough expertise available <= this has been an issue
- Is there enough time to do the project?
- Are there alternatives that should be used instead?
- Is special support needed?

Account Management

- Well defined account management project
- Proven
- Integration with XSEDE
- But
  - using of YubiKeys for access to management node
- Integration of user consultation



SDSC·

SDSC·

USER/PI

IU

SDSC

PI Submit Proposal to XSEDE

Review Proposal to identify possible VC users

Notify about the user(s) identified

Ask project PI to create CVC account on XUP

PI create CVC account on XUP

Create the initial ticket in SDSC Jira

VC ticket in Jira

Reach out to User/PI for other request infromation

User/PI provide other missing information

Update jira ticket and notify

VC Setup

User's special feature request and/or usage problem

Communicating with User and back-end team(regarding feature request of VC and usage problems (# cores, RAM, Storage, IB, public IP access to compute nodes, etc.)

Account setup in nucleus and YubiKey configuration

Notify User regarding the VC creation status and the usge of cloudmesh clietn tool to manage VC

User fill out, sign, and fax back the receipt ack form

Ship YubiKey and User receipt ack form to User

Communicating Secret/PIN back to user and introducing the usage of the token

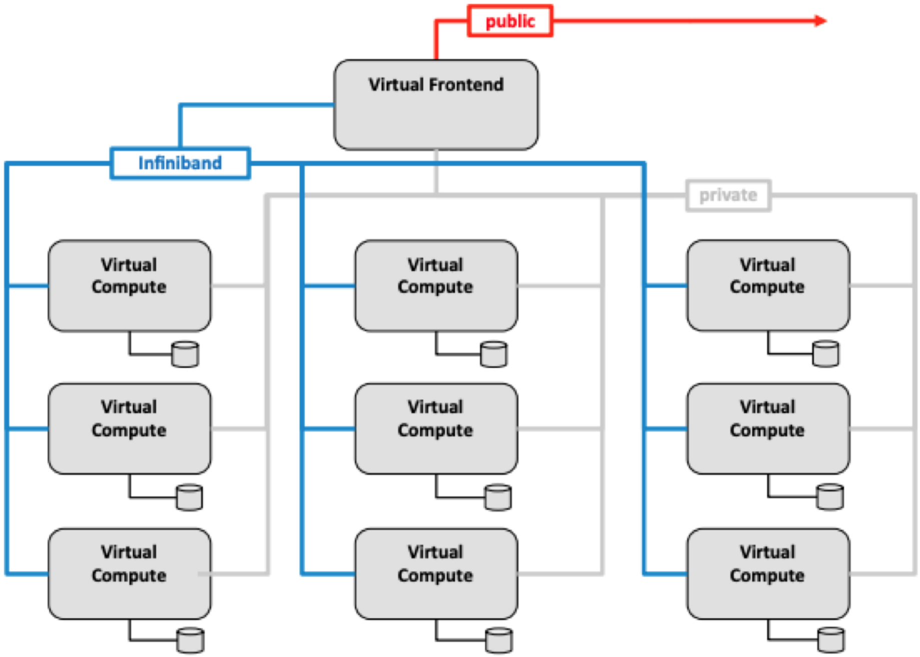
Communicate Secret/PIN of YubiKey to IU

User access the assigned VC

Close the Jira ticket upon user's successful access to VC

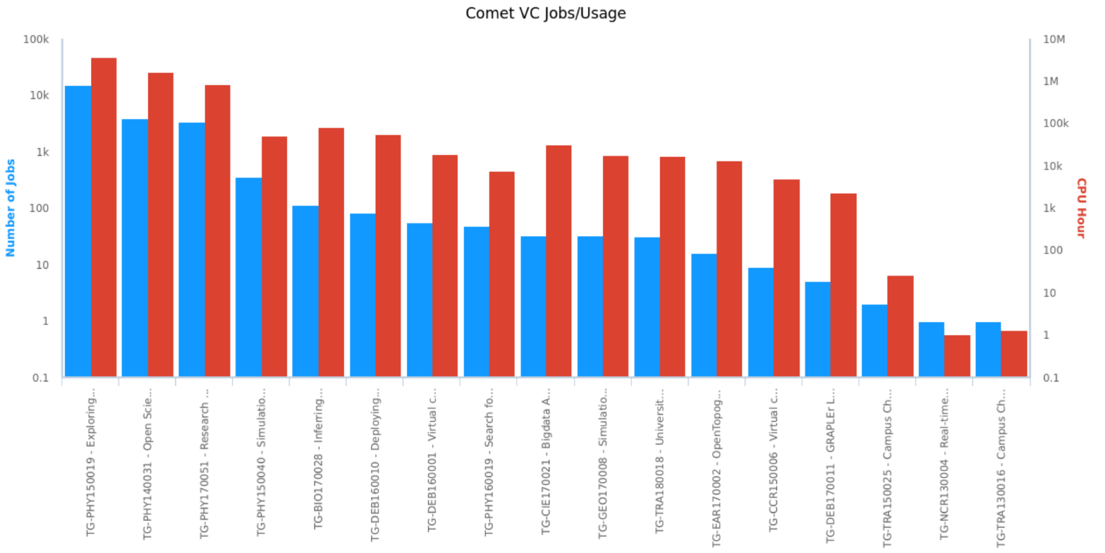
VCs on Comet

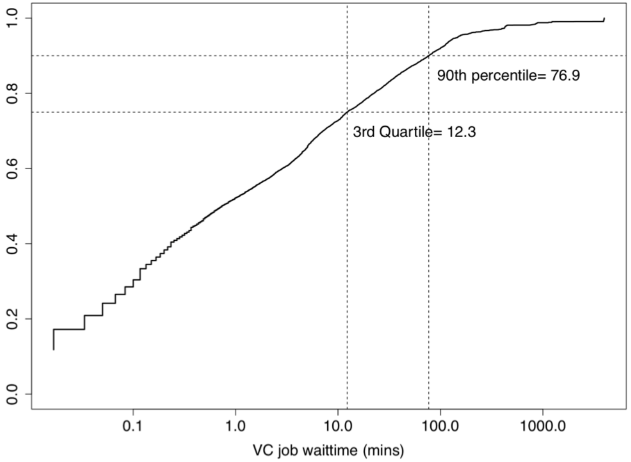
- User gets a virtual cluster
- management through frontend
- Users get n cluster nodes (dependent on need).
- cluster nodes can be user or not
- VCs are all managed by queueing system invisible to the user
- Can use all of the clusters backend services and performance



Virtual Cluster Monitoring

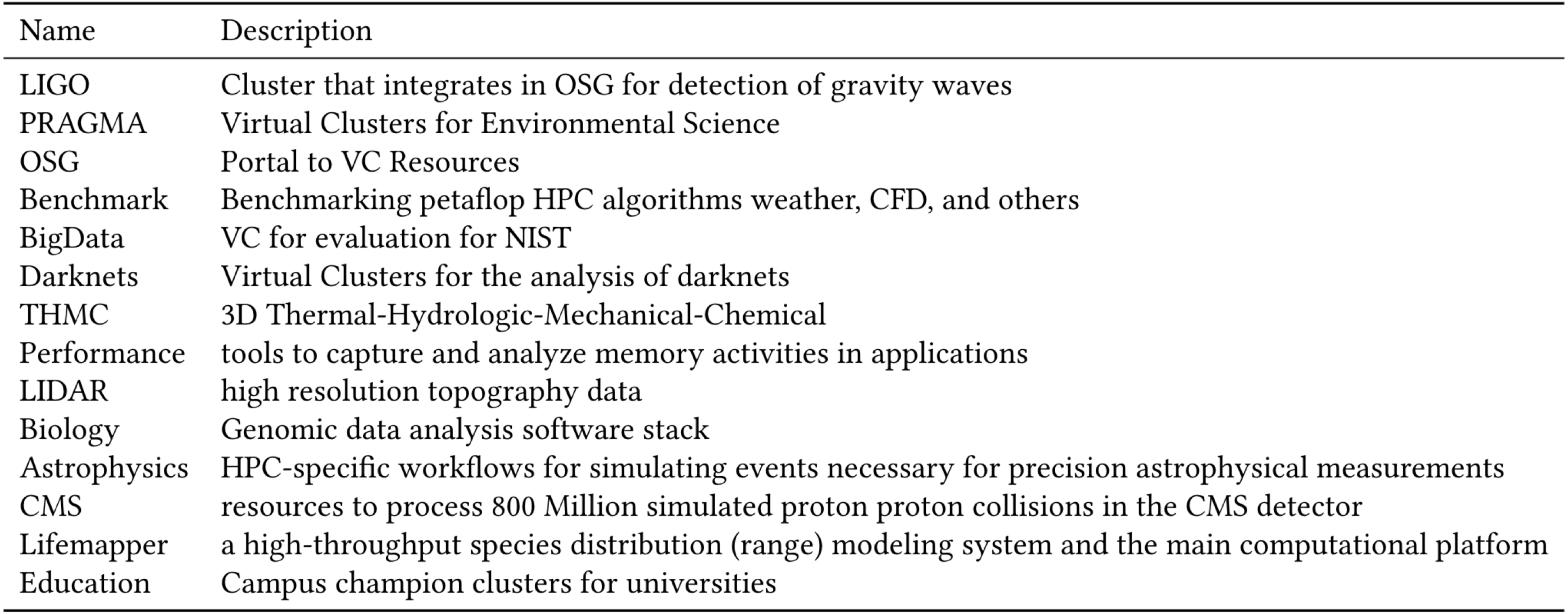
- XSEDE monitoring
- XDMoD integration available
- Special Monitoring
- relatively low wait time





1 to 2018-12-31 Src: XDCDB. Powered by XDMoD/Highcharts

Virtual Cluster Applications

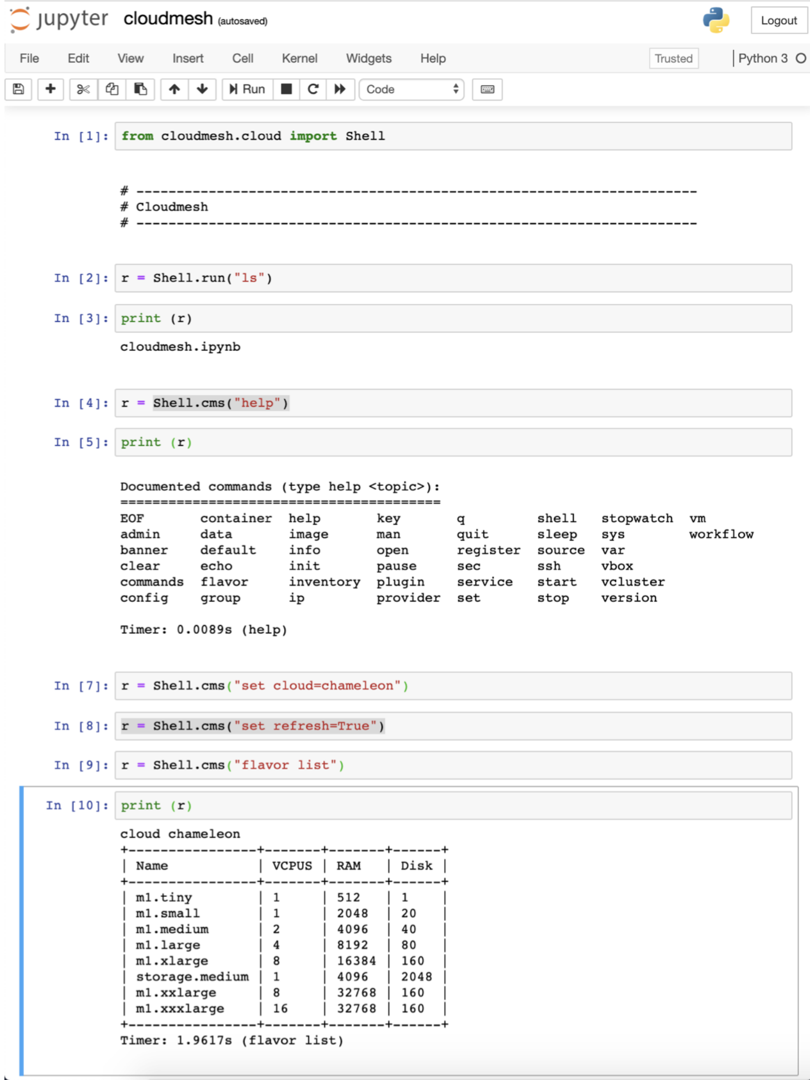


Name	Description
LIGO	Cluster that integrates in OSG for detection of gravity waves
PRAGMA	Virtual Clusters for Environmental Science
OSG	Portal to VC Resources
Benchmark	Benchmarking petaflop HPC algorithms weather, CFD, and others
BigData	VC for evaluation for NIST
Darknets	Virtual Clusters for the analysis of darknets
THMC	3D Thermal-Hydrologic-Mechanical-Chemical
Performance	tools to capture and analyze memory activities in applications
LIDAR	high resolution topography data
Biology	Genomic data analysis software stack
Astrophysics	HPC-specific workflows for simulating events necessary for precision astrophysical measurements
CMS	resources to process 800 Million simulated proton proton collisions in the CMS detector
Lifemapper	a high-throughput species distribution (range) modeling system and the main computational platform
Education	Campus champion clusters for universities

Jupyter Integration

Cloudmesh can easily be integrated into jupyter

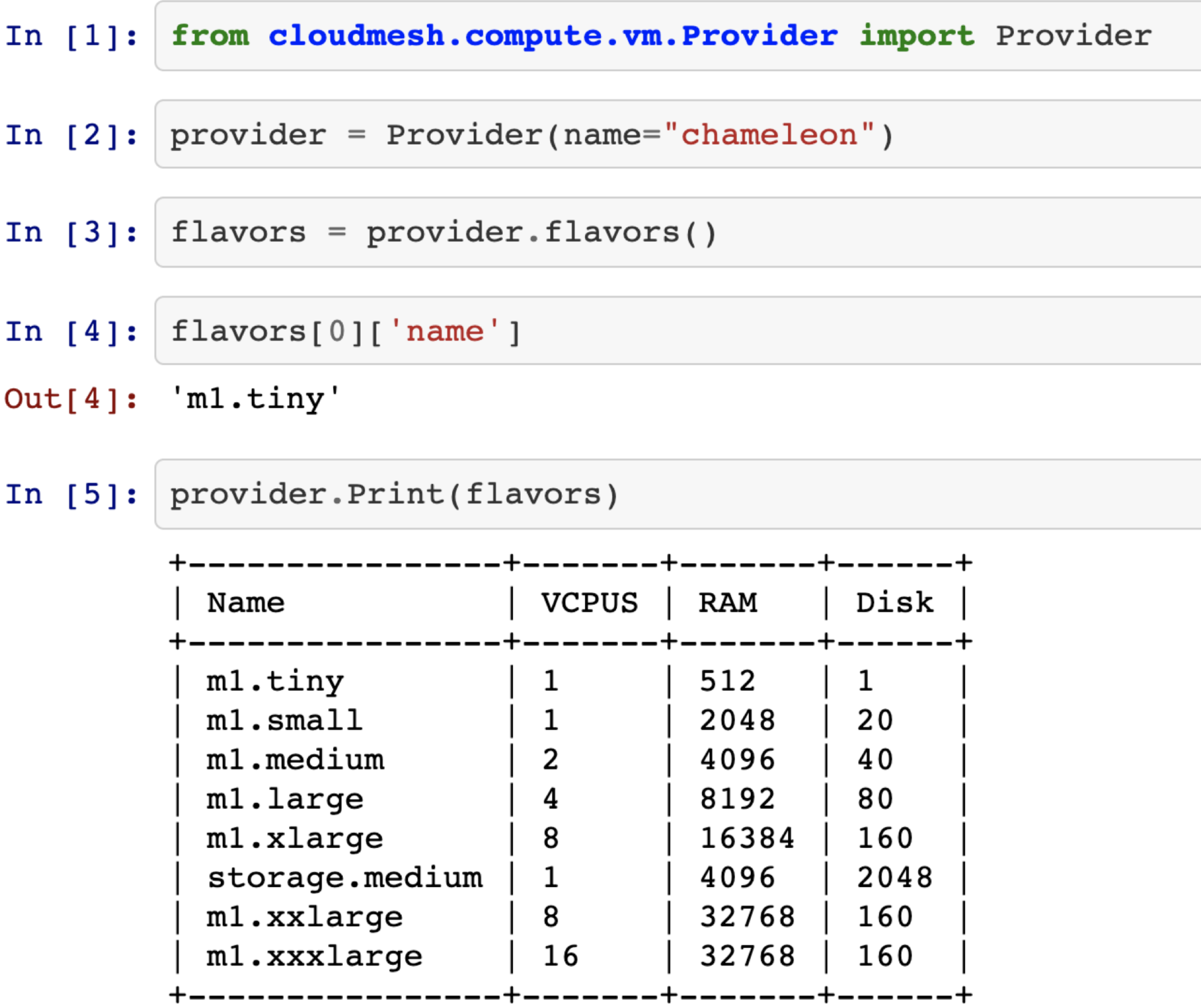
The command shell is readily accessible via an API call



```
In [1]: from cloudmesh.cloud import Shell

        # ----------------------------------------------------------------------
        # Cloudmesh
        # ----------------------------------------------------------------------

In [2]: r = Shell.run("ls")

In [3]: print (r)

        cloudmesh.ipynb

In [4]: r = Shell.cms("help")

In [5]: print (r)

        Documented commands (type help <topic>):
        ========================================
        EOF       container  help       key       q         shell   stopwatch  vm
        admin     data       image      man       quit      sleep   sys        workflow
        banner    default    info       open      register  source  var
        clear     echo       init       pause     sec       ssh     vbox
        commands  flavor     inventory  plugin    service   start   vcluster
        config    group      ip         provider  set       stop    version

        Timer: 0.0089s (help)

In [7]: r = Shell.cms("set cloud=chameleon")

In [8]: r = Shell.cms("set refresh=True")

In [9]: r = Shell.cms("flavor list")

In [10]: print (r)

        cloud chameleon
        +-----------------+-------+-------+------+
        | Name            | VCPUS | RAM   | Disk |
        +-----------------+-------+-------+------+
        | m1.tiny         | 1     | 512   | 1    |
        | m1.small        | 1     | 2048  | 20   |
        | m1.medium       | 2     | 4096  | 40   |
        | m1.large        | 4     | 8192  | 80   |
        | m1.xlarge       | 8     | 16384 | 160  |
        | storage.medium  | 1     | 4096  | 2048 |
        | m1.xxlarge      | 8     | 32768 | 160  |
        | m1.xxxlarge     | 16    | 32768 | 160  |
        +-----------------+-------+-------+------+
        Timer: 1.9617s (flavor list)
```

Simple API

Super simple API that allows integration with jupyter notebooks very easily



```
In [1]: from cloudmesh.compute.vm.Provider import Provider

In [2]: provider = Provider(name="chameleon")

In [3]: flavors = provider.flavors()

In [4]: flavors[0]['name']

Out[4]: 'm1.tiny'

In [5]: provider.Print(flavors)

+----------------+-------+-------+------+
| Name           | VCPUS | RAM   | Disk |
+----------------+-------+-------+------+
| m1.tiny        | 1     | 512   | 1    |
| m1.small       | 1     | 2048  | 20   |
| m1.medium      | 2     | 4096  | 40   |
| m1.large       | 4     | 8192  | 80   |
| m1.xlarge      | 8     | 16384 | 160  |
| storage.medium | 1     | 4096  | 2048 |
| m1.xxlarge     | 8     | 32768 | 160  |
| m1.xxxlarge    | 16    | 32768 | 160  |
+----------------+-------+-------+------+
```

Scientific Impact metrics

- We can alanyls your organizations scientific impact metrics based on publications
- We have a unique metric that can compare your peer groups based on publication venues

- This is different from just i-ndex

- We have done this for
  - XSEDE
  - NCAR
  - Blue Waters
- We could do this based on
  - Department
  - Research group
  - Researcher

Lessons Learned

- Advantages: Software
  - Superb software, integrates well with existing clusters, including cloudmesh
  - Good user support, easy to get access and use a VC
  - Backend does not have have “create my comet cluster elsewhere” (not part of funded project)
- Issues: Users
  - User knowledge to be a system admin managing the cluster is limited
    - Knowledge, manpower, staff retention at organizations or projects
- Issues: Shifting Community Interest
  - Community wanted to learn OpenStack
  - Shift from VMs to containers

References

- Cloudmesh: http://github.org/cloudmesh
- Cloudmesh Comet Plugin: https://cloudmesh.github.io/cloudmesh-comet/
- VC user guide:
  - https://cloudmesh.github.io/cloudmesh-comet/comet.html#comet-userguide
- VC CLI reference documentation:
  - https://cloudmesh.github.io/cloudmesh-comet/command_comet.html#comet-command