Using Virtualized Computing Resources with DIRAC

A. Tsaregorodtsev for the DIRAC Project

CPPM-IN2P3-CNRS, Marseille, ISGC, 9 March 2017, Taipei
The problem of management of cloud computing resources

VMDIRAC cloud resources manager
- Interfacing to cloud providers
- VM contextualization
- WMS pilot jobs in VMs
- VM life cycle management

Managing policies for cloud resources usage

Status and ongoing developments

Conclusions
Unlike computational grids, reserving and creation of resources (VMs) as well as managing their life cycle is the responsibility of the users

- Tools needed to do it as transparently as possible

Unlike commercial clouds, public cloud resources are not “unlimited”

- We have to share fairly limited resources between user communities without losing efficiency and flexibility of clouds

Resources in commercial clouds are more and more used for scientific computing

- How to use them together with public resources efficiently and within allocated budgets
DIRAC provides all the necessary components to build ad-hoc grid infrastructures interconnecting computing resources of different types, allowing interoperability and simplifying interfaces. This allows to speak about the DIRAC interware.
Pilot jobs are submitted to computing resources of various types by specialized Pilot Directors.

Running the same pilot jobs everywhere allows to integrate heterogeneous resources transparently for the users.

This is fully applicable to the cloud resources as well.
VMDIRAC extension

- VMDIRAC extension developed for Belle MC production system
  - Dynamic VM spawning taking Amazon EC2 spot prices and Task Queue state into account

- Now VMDIRAC is a general purpose service for VMs lifecycle management
  - Creation
  - Monitoring
  - Discarding
Cloud resources configuration

Grid sites

- LCG.CATANIA.it
- LCG.CBPF.br
- LCG.CC.fr
  - Description = IN2P3-CNRS Computing Center
  - Name = IN2P3-CC
  - SE = IN2P3-disk, DIRAC-USER
  - Coordinates = 4.8655:45.7825
  - Mail = grid.admin@cc.in2p3.fr
- CEs
  - ccreamceli09.in2p3.fr
    - wntmpdir = unset
    - architecture = x86_64
    - OS = ScientificSL_Carbon_6.7
    - S100 = 2685
    - Pilot = True
    - CEType = CREAM
    - SubmissionMode = Direct
    - OutputURL = gsiftp://localhost
- Queues
  - HostRAM = 48257
  - MaxRAM = 62918
- ccreamceli10.in2p3.fr
- ccreamceli11.in2p3.fr

Cloud sites

- Cloud.LUPM.fr
- Cloud.IPHC.fr
- Cloud.CC.fr
  - CE = cckeystone.in2p3.fr
  - Cloud
    - cckeystone.in2p3.fr
      - CEType = Cloud
      - ex_keyname = DIRAC_test
      - ex_security_groups = default
      - ex_force_auth_url = https://cckeystone.in2p3.fr:35357/v2.0/tokens
      - ex_tenant_name = htc-dirac
      - ex_force_auth_version = 2.0_password
      - ex_force_service_region = regionOne
      - ipPool = nova
    - Images
      - CentOS7-large
        - ImageID = 9df72f29-15d4-4433-b120-2dc084695100
        - FlavorName = m1.large
        - VO = biomed
      - SL6-large
        - CreatePublicIP = False
        - MaxInstances = 4
Cloud resources configuration

- Similar to other computing resources
  - Cloud sites
  - Cloud endpoints
  - VM images (OS + size/flavor)
    - Equivalent to batch queue description
    - Can have VO specific tags for matching with user payloads

- Contains all the necessary details for
  - VM creation with required properties
  - Job requirements matching to VM capabilities

- Cloud resource status monitoring
  - Included in the DIRAC Resource Status Service (RSS) in a similar way as any other computing resources
    - Same databases
    - Different testing commands and status evaluation policy plug-ins
    - Work in progress
Preferring standard “official” bare minimum images

- SL6, CC7, CernVM
  - CVMFS is installed while the contextualization if not present in the image
- Avoid image maintenance (security updates, etc.)
- VO specific software can be installed as part of custom contextualization

Custom images (appliances) are still possible

- Special OS
- Preinstalled DIRAC
- Preinstalled VO software
Cloud endpoint plugins to interact with particular cloud provides

Cloud endpoint abstraction

Implementation (IHEP, Beijing)

- Apache-libcloud
  - Catch-all library, but not really…
- Rocci
  - Using command line interface
  - Allow connections with GSI proxies
- EC2
  - Boto python API

More implementations are in the works

- OCCI, Google, Azur, IBM, …
VM submission

- **CloudDirector – VMDIRAC way**
  - Similar to SiteDirector for grid jobs submission
  - VM submission based on the Task Queue status
    - If there are waiting user payloads
    - VM properties corresponding to payload requirements

- **Vac/Vcycle (A. McNab)**
  - Not VMDIRAC
  - Used by LHCb
  - No a priori knowledge about the state of the Task Queue
  - Similar contextualization and pilots
Running Pods

- Cloud resources reserved for a particular production campaign
  - Particular sites, images, tags, etc
  - End and start dates of the campaign
  - Statically or dynamically allocated

- Suitable for commercial resources reservation for well-defined activities – production campaigns

- User jobs can specify a RunningPod tag to chose those resources
  - Only jobs specifying this tag can run there
VM Contextualization

- **cloud-init mechanism**
  - Using a password-less certificate passed as user data
    - Long user proxy or service/host certificate
  - Using bootstrapping scripts similar to Vac/Vcycle
    - Using pilot 2.0
    - Setting up MJF environment if available on the site
    - Mounting attached disk storage
  - On the fly installation of DIRAC, CVMFS if necessary
  - Starting VM Monitor Agent
  - Starting one or more pilots to manage the VM job slots

- **SSH mechanism**
  - Bootstrapping by sending commands through an SSH tunnel
    - Requires public IP address
  - Mostly obsoleted, using cloud-init
Pilots in the VMs

- Same as any other pilots
  - DIRAC Pilot 2.0 framework
    - A set of commands for the DIRAC environment installation and setup, starting Job Agents interacting with the WMS central service
    - User communities can provide custom pilot commands in addition and/or in replacement of the standard ones

- Managing the VM CPU cores scenarios
  - Launching as many pilots as they are cores
    - Suitable for single-core payloads, à la grid jobs
  - Launching single pilot
    - Suitable for multi-core payloads occupying the whole VM
  - Single pilot with a PoolComputingElement plugin for payloads execution
    - Simple “batch system” to manage VM job slots
    - Can execute payloads with any requirements to the number of cores: single, exact number of cores or whole node occupancy
Managing VM life cycle

- VM Monitor Agent is launched in parallel with the pilot process during the VM bootstrapping
  - This is a watchdog for activities on the VM
  - Sends heartbeats and VM status information to the central VM Manager service
  - Can receive instructions from the central service as a response to the heartbeat
    - E.g., halt, drain and other commands
  - Monitors the VM status
    - CPU load
    - Pilots status via log files
  - Can be configured to halt the VM with different policies
    - Strict life time, à la batch system
    - Zero CPU load
    - No active payloads
VM management

1. VM Scheduler
2. VM Manager Service
3. Accounting Service
4. WMS Services
5. VM Monitor
6. Pilot
7. Logs

- Commands
- Heartbeats
- State updates
Managing resource usage policies

- Prerequisites for automatic enforcement of policies for cloud resources usage
  - Information about all the waiting payloads
  - Accounting for the history of resources consumption by users and groups
  - Communication channel to the VMs via the VM Monitor Agents

- This allows to manage resource allocation according to various policies
  - Static shares, simple but least interesting
  - Fair shares enforcements using “batch system” algorithms
  - Shares of different groups within the same VO
  - Shares between different VOs
    - Can be done by DIRAC provided as a service by multi-VO infrastructures
      - E.g., DIRAC4EGI
    - Requires delegation of inter-VO policies management to the DIRAC level rather than doing it on-site

- This is the work in progress
  - VM Scheduler service
Web VMDIRAC interface
Web VMDIRAC interface

- Monitoring
  - VM status and operation parameters

- Accounting
  - Number of VMs, executed jobs, data transfers, etc
  - Per site/endpoint, per image, per RunningPod

- Administration
  - VM manipulation by administrators
    - Start, halt, other instructions to the VM Monitor Agent
  - Public IP association to connect for debugging problems

- Work in progress
  - Refactor to make uniform with other computing resources monitoring
    - Abandon usage of Google tools
  - Expose more functionality in managing VMs
VMDIRAC is provided as part of multi-VO DIRAC services

- **France-Grilles DIRAC**
  - Using Cloud Federation resources in France (< 10 sites)
  - Communities: biomed, vo.france-grilles.fr
- **DIRAC4EGI**
  - Using EGI FedCloud resources
  - Communities: enmr-eu, training
- Same usage patterns as for the grid resources, same user experience

- **BES III, Belle II experiments**
  - Migrating from VMDIRAC 1.0 to VMDIRAC 2.0
  - Using clouds in production activity

- **CTA Collaboration**
  - Preparing a Data Challenge involving HNCloud resources
Ongoing development work

- Following evolution of cloud provider service interfaces
  - Moving towards using REST interfaces instead of third party bindings
- Flexible usage of VM CPU cores
  - Single vs Multi-core payloads on the same VM
  - Job masonry algorithms
- Enhanced VM Monitoring functionality
  - Graceful VM shutdown, draining, sending signals to the payload
- Enhanced web monitoring and administration tools
- VM Scheduler for flexible multi-community policies of cloud resources usage
  - Including scenarios for commercial cloud resources
VMDIRAC as an extension

- VMDIRAC is developed as the DIRAC extension in a common framework with other DIRAC subsystems
  - Easy access to various DIRAC services, databases, etc
  - Can be further extended with VO specific services and scenarios, web applications
  - Developers having experience with the DIRAC development framework can make contributions easily

- Helps making cloud resources usage completely transparent for the users
Conclusions

- More and more computing resources are now provided via Cloud technologies

- VMDIRAC is providing means for cloud computational resources description and integration with other resources within the same Workload Management System

- The VMDIRAC architecture allows to support complex multi-community policies of using cloud resources including fair sharing scenarios

- The system is under active development to follow evolution of the cloud technologies and to provide new functionalities to users and administrators of the DIRAC services

http://diracgrid.org