Table of Contents

Clouds

Introduction

An popular model to provide computing resources in industry nowadays are clouds. They come in different manifestations,

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)
- Software as a Service (SaaS)

There are many definitions on the market about what a cloud is. Often, they are defined via their features. Some ideas of Clouds, specifically the Infrastructure as a Service approach, can be of interest for managing sites, in particular if this approach is combined with large scale virtualization of computer centers. Virtualization is often called an enabling technology for IaaS clouds. It has to be stressed though that virtualization, which is reality at many sites already even for production services, must not be confused with cloud computing.

Several sites have some limited experiences with clouds already. As an example, CERN is running an infrastructure called "lxcloud" for more than a year now, which is used for virtualizing parts of the batch resources, and which provides an EC2 access as a pilot service for selected users from different VOs. The infrastructure has been scaled up during a scalability test up to 16,000 virtual machines, all configured as standard worker nodes, and connecting to LSF. This exercise unveiled possible issues on the application side, in this case LSF, which turn up when scaling up a highly dynamic environment.

An internal IaaS cloud is fully transparent to users. Grid services are simple applications which use machines provided by the internal IaaS cloud, and Grid service managers become users of this general infrastructure of the site.

The picture changes, if sites decide to open their internal cloud to specific users, by granting them access to their resource interface.

Experiments view of clouds

Commercial providers offer computing resources as IaaS. Prominent examples are Amazon and others. Several experiments have started to exploit such resources. From the experiments this requires some work to make their frameworks work on such environments.

Use cases:

- satisfy peak requests eg before important conferences, if all sites are full
- speed up of analyzes by buying additional resources from elsewhere
- small universities who only occasionally need larger amounts of computing resources

Based on these experiences, some experiments have expressed their interest in having an equivalent entry point to sites. The feasibility of this has been demonstrated for example by ATLAS, making use of small existing pilot installations, for example at CERN (

Sites view of clouds

For sites, one interesting aspect which often goes along with cloud computing models is virtualization. On many sites, this is already used at a large scale even for production services, specifically for critical services.

This is usually entirely transparent for the end users and the experiments. As many sites are under heavy pressure to reduce their operational cost while being forced to grow without getting additional man power resources, it is natural that virtualization efforts are being extended. Several sites have started to experiment with virtualization of batch resources. Different approaches to this exist:

pure IaaS approach

In this approach, in the extreme case all physical resources are installed as hypervisors, and resources are used in an IaaS way even for internal services. The batch system becomes just one application among others which uses this infrastructure, and worker nodes are virtualized. This gives the maximum flexibility in placing resources. Virtual machine requests and placements can be done using a cloud controller such as OpenStack [1], OpenNebula [2] or similar. Which one to use is a site decision. Often, these tools come with a cloud interface, which implements a subset of the EC2 entry point from Amazon. This way, resource access in an Amazon like style is fairly cheap for the sites.

Advantages

- very flexible and extensible to virtually all services in the computer center
- efficient resource usage by supporting life migration of running VMs, based on the load of the hypervisors
- clear separation of resources and applications
- industry way approach
- requires fairly little site-specific, non-standard developments

Disadvantages

The disadvantages of this approach are mainly in the application layer.

- scalability of some applications, specifically batch. In the extreme case where each worker node has 1 CPU and maps to 1 physical core results in a large multiplication factor in terms of the number of worker nodes
- current implementations of cloud schedulers are not yet as sophisticated as it will probably be required
- accounting and billing schema needs to be changed
- applications need to be able to cope with dynamic coming and going of nodes, to be able to fully exploit all advantages of this approach.

running payload in a VM

In this approach, it is the batch system scheduler which is in control of the resources. Jobs are not directly started on the physical hardware, but a virtual machine is started instead which then runs the payload. This approach provides virtualization of batch resources, but has nothing to do with clouds. It is mentioned here though because some sites have tried this approach in the past.

Advantages

- Accounting works as before
- Fair share and scheduling remains unchanged

Disadvantages

- scalability may be an issue if this approach is to be extended for the whole site
- conceptually complex approach

The IaaS approach is a modern and emerging technology which should be evaluated further, as it promises larger scalability of sites, improved resource usage, and a way to make WLCG less special than it is now, while being compatible with the current way of working (if implemented properly).

lxcloud [4] pilot service

CERN's lxcloud is effectively a bunch of KVM hypervisors with a minimal software setup. The available local disk space is bundled in a logical volume, and used to prestage images and provide local scratch space for virtual machines running on it. The machines are standard CPU servers of the kind which are used in CERN batch farm. OpenNebula [1] is in use to manage virtual machines on this infrastructure. A large fraction of the resources are reserved for a virtual batch farm, the rest is available via OpenNebula's EC2 interface to selected users from different VOs and communities at CERN. lxcloud is designed for speed. Only endorsed images are allowed to be run there, users cannot upload their own images.

Aspects of Infrastructure as a Service (IaaS)

Dynamicality

One of the main features of a cloud is that it is very dynamic. Virtual machines come in and go out at a high rate, and change their identity or task. At CERN this feature has been tested now for over more than a year now, by running a small fraction of the batch farm on a cloud like infrastructure. The virtual batch worker nodes are automatically updated, and allow for the deployment of user transparent intrusive updates, which do not require any service manager intervention. This large gain of operational cost goes on cost of a fairly low cost in performance.

The dynamic approach requires that virtual machines have a limited life time, and are returned (=destroyed) after this period has finished. In the case of an EC2 access where users can start instances of virtual machines, this is not easy to do unless a proper accounting and billing system is introduced. If the site decides to enforce a limited life time on virtual machines launched this way, a mechanism is required which allows to inform the application running inside the VM to determine the remaining life time. This issue has been partly addressed already by the HEPiX [3] virtualization working group.

Scheduling

Within an pure IaaS infrastructure it is easy to create virtual machines of a given type. This dynamic approach can also be used to dynamically adapt the system to user requirements on Operating system, SMT jobs and similar. This dynamical approach for batch requires an additional scheduler, from now on called a "Batch Cloud Factory" which has access to job requirements of pending jobs in the batch farm, and which can be tuned by the batch service manager.

Accounting

The performance of the virtual machine should be known by the application running on it. The CPU factor in a virtualized environment depends not only on the performance of the hypervisor but also on the number of VMs running on it, in case resources are overcommited.

Accounting, specifically for virtual batch nodes, becomes more tricky if life migration is supported and used, and if living machines are migrated to different hardware. In the case of a virtual batch farm, in such a case the CPU factor will change while jobs are running, and the batch scheduler needs to be aware of this.

In any case the application running on the VM should have a way to find out about the performance of the VM it is running on. This problem has been discussed and partially solved within the HEPiX virtualization working group. The approach is that there is a pre-defined file which specifies the HS06 value for the VM

which is filled in during contextualization at startup of the VM.

If sites offer an alternative access to their resources to VOs, these resources should be deduced from their normal batch shares.

Relation to other fields of the WG

Whole nodes scheduling can be implemented as virtual worker nodes with only one job slot but different sizes. Starting such resource should be one of the tasks of the "Batch Cloud Factory". It needs to be able to intercept know about the request for full nodes. Several possibilities exist:

- The BLAH interface on the CREAM CE directly launches such VMs via the EC2 interface on the local cloud. This assumes that the EC2 access uses the same physical resources as the virtualized batch farm
- The BLAH interface on the CREAM CE informs the "Batch Cloud Factory" that it needs such resources, and how many. The jobs will be dispatched only once these resources are made available.
- The "Batch Cloud Factory" examines pending jobs for there job requirements, and launches VMs as needed, including those for the full node requests. It is not necessary to have a dedicated queue for this.

Cloud access methods

Current open source cloud schedulers come with partial implementations of Amazones EC2. It should be stressed that EC2 as such is not a standard, and can be changed by Amazon at any time. Attempts to solve this exist. Implementation of OCCI exist for example in OpenNebula. While in general, free and standarized protocols should be preferred over commercial protocols like EC2, the latter still seems to be a good starting point as such an entry point is fairly easy to provide for the sites.

The working group should come up with a recommendation for the cloud access.

Authentication and authorization

Direct cloud access to a site requires authentication and authorization. This is usually user/password based. With OpenNebula authentication via x509 is possible, and based on the DN. Authentication and authorization is a field which requires further investigation and developments.

Clouds and finite resources

The typical cloud model assumes the availability of virtually infinite resources. If no free resources are available to satisfy an incoming request, the user is immediately told to go away. Incoming requests are not queued up, and consequently, there is no concept of "fair share" in this model. However, it is possible to setup fixed quotas per user or user group in some systems (like OpenNebula).

Supported Node types

Supported node types for EC2 access or for whole node scheduling should be defined by the working group. They must be unique across sites to avoid confusion. We propose:

tag name	CPUs	physical Memory [GB]
vm_small	1	2
vm_medium	2	4

vm_large	4	8
vm_huge	8	16
vm_xhuge	16	32

If tags in the batch system are used, these should match with the EC2 VM templates above to avoid confusion. The available scratch space depends on the hardware available at the sites. It can be constant for all types of virtual machine flavors above.

[To be discussed: should the naming conventions be in line with what are people used to from other providers ? eg m1_small]

Applications

Anything which makes use of the IaaS infrastructure of the site, if or not it is open to users via a cloud interface, is called an application. In most cases this will be a virtual machine which is derived from an image. The virtual machine guests, in particular if they are long lived, need to be managed by as central system such as Puppet or Quattor or similar, for an easy deployment of operating system updates and configuration changes, security fixes etc.

The application software can be either managed in the same way as the operating system, or it can be shipped with the image. Another option is to distribute them via CVMFS. The use of AFS is problematic in a highly dynamic environment at large scales because of the way the protocol works.

The highly dynamic environment also has an impact on some applications which need to support this. An example is the use of dynamic worker nodes which can join and leave the batch farm, change their identity and CPU factors on the fly. Not all batch systems can deal with that in an efficient way.

Data locality

A general problem which needs to be thought of in the cloud approach is data locality.

Recommendations

- Sites are free to use virtualization. If they do, it must be transparent to users
- Sites which decide to go beyond virtualization, and setup an internal cloud, are encouraged to open part of the resources via an EC2 access to VOs which are interested in this.
- If these cloud resources are to be deduced from the batch resources of the individual VOs, fixed quotas per VO should be setup to ensure that each VO get their pledges.
- Free protocols for Cloud access are to be preferred over proprietary formats like EC2. Realistically though, for the time being EC2 seems to be the most popular access method.
- virtual machine sizes should be in line with fixed sizes for whole nodes. These should be unique across all sides.
- In the described IaaS models, experiments need to integrate their frameworks with clouds, and sites need to adapt their applications to use internal clouds. We recommend to found a body which involves all parties to synchronize these efforts, with the aim to avoid independent developments and multiple solutions for the same problems.

References

[1] [http://openstack.org][OpenStack]

[2] [http://opennebula.org][OpenNebula]

[3] [http://hepix.org][HEPiX]

[4] [https://twiki.cern.ch/twiki/bin/viewauth/CloudServices][lxcloud]

[5] Template for reports

-- UlrichSchwickerath - 19-Jan-2012

-- UlrichSchwickerath - 05-Mar-2012

This topic: LCG > WLCGWlMgmtTECloudComputingDraft
Topic revision: r1 - 2012-03-05 - UlrichSchwickerath