Interoperability of Commercial and Open-Source Clouds for Bioinformatics Grid Computing with CloudBioLinux-CloudMan

Alex Richter[1], Konstantinos Krampis*[2], Enis Afgan*[3], Brad Chapman*[4]

[1] Informatics Department, J. Craig Venter Institute, 10355 Science Center Dr., San Diego, CA 92121, USA
[2] Informatics Department, J. Craig Venter Institute, 9704 Medical Center Dr., Rockville, MD 20850, USA
[3] Center for Informatics and Computing, Ruder Boskovic Institute, Zagreb, Croatia
[4] Dept. of Biostatistics, Harvard School of Public Health, 655 Huntington Avenue, Boston, MA, 02115, USA

Email: Konstantinos Krampis*- agbiotec@gmail.com; Enis Afgan*- afgane@gmail.com; Brad Chapman*- chapmanb@gmail.com;

*Corresponding author

Abstract

Background: Porting across cloud platforms of complex bioinformatic pipelines that are self-contained within Virtual Machine (VM) files that encapsulate the operating system, bioinformatic tools and software libraries, is straightforward using utilities available by vendors of open-source and commercial clouds. Parallel processing of large-scale datasets using bioinformatics pipelines running independently within multiple instances of a specific VM is a simple but efficient approach, with such evidence presented in the current study. Nonetheless, effort towards virtualization and replication within clouds of such grid computing found on institutional compute clusters has significant merit, given that a majority of current bioinformatics pipelines are tied to institutional grids and would need significant code refactoring to be used outside of such informatics infrastructures.

Results: In the present study, we leverage the CloudMan framework that allows bootstrapping of virtualized Sun Grid Engine (SGE) infrastructures using VMs on the Amazon EC2 cloud. Within the specific context of Cloudman and its interactions with the Application Programming Interface (API) of the commercial Amazon EC2 or the open-source Eucalyptus and Openstack clouds for the purpose of instantiating virtualized grids, we first examine API cross-compatibility among these clouds. Furthermore, we argue on the feasibility for development of similar frameworks to CloudMan that allow virtualization of current institutional bioinformatics infrastructures and facilitate deployment of current bioinformatic pipelines with minimal code refactoring across the different cloud platforms. Finally, we present test cases on performance and efficient utilization of underlying physical server resources for data analysis pipelines utilizing this framework.

Conclusions: Unlike newly developed web applications that can be designed from the ground up to utilize the cloud APIs and natively run within the cloud infrastructure, most complex bioinformatics pipelines that are in advanced development state can only be encapsulated within VMs along with all their software dependencies. Nonetheless, for taking advantage of the scalability offered by the cloud, additional frameworks similar to Cloudman are required that stand up virtualized grids and emulate the most common infrastructures found on institutional clusters where usually most of the bioinformatics pipelines run.

Background

Democratizing large-scale data analysis for next-gen sequencing

The growth in sequencing throughput and reduction of sequencing cost has shifted the challenge from data generation to data storage and analysis **??**. Considering only the field of genomics, tens of TeraBytes of data have been released in the recent five years [1]. Acquiring the reads during a whole genome shotgun sequencing project is only the first step, and must be followed by bioinformatic analysis that involves running post-sequencing data analysis software, such as read quality checks, trimming and barcode deconvolution, in addition to assembly, annotation, and output interpretation in order to generate scientific value from the sequence data.

Sequencing instruments are typically bundled with only minimal computational and storage capacity, that is sufficient for data capture during runs. In recent years, a new paradigm in genomic sequencing has emerged with small-factor, benchtop sequencers, that provide enough throughput for sequencing small to medium-sized genomes, but are priced at levels affordable for researchers running independent laboratories. Examples include the GS Junior by 454, MiSeq by Illumina and Ion Proton by Life Technologies, providing per run sequencing capacity at 0.035Gb, 1.5Gb and 1Gb respectively (review in [2]). Nonetheless, given the scale of genomic datasets, scientific value cannot be obtained from an investment in a sequencer, unless it is accompanied by an equal investment in bioinformatics infrastructure. Processing large amounts of genomic data requires access to a range of technical capabilities including expertise in information technology (IT), bioinformatics, and software engineering, as well as to large-scale computational and storage capabilities.

Centralized bioinformatic centers offering data analysis services to the community where expertise and computational resources are concentrated and economies of scale can minimize costs, have been established with support from federal grants in the US and Europe. Nonetheless, many of these centers are struggling to keep up with the volume and variety of data that is being produced, and in most cases analysis is restricted to sequencing data generated in-house, using genome samples submitted by community researchers. Offering a publicly accessibly analysis platform for 3rd party, independent laboratories that perform genome sequencing, is available only with some exceptions at these centers and is limited in scope (for example MG-RAST for metagenomics [3], [4]). Compute resources of a national scale such as TeraGrid [5] and XSEDE [**?**] provide large-scale computer clusters to the public, but support for bioinformatics software and installation of complex data analysis pipelines is limited.

The adoption of cloud computing technologies is providing an opportunity to change this paradigm by effectively democratizing the field, meaning that individual researchers and labs can now have access to the resources that were previously only available to the large sequencing and bioinformatic centers. Nonetheless, a problem arises in regards to configuring and utilizing bioinformatics software for next-generation sequencing, where significant expertise is required on Unix operating systems, programming languages, software compilation from source code, file systems and large-scale data management. For smaller laboratories this can become a significant obstacle, as in addition to coming up with the funds for building an informatics infrastructure with capacity to handle the large computations, they also need to hire trained bioinformaticians competent to install, configure and run sequence analysis software tools and manage the data, which can present a higher expense than that of acquiring the hardware.

Along these lines, virtualization technologies allow entire compute servers including the operating system and all the necessary software packages for data analysis, to be pre-installed, configured and ready-to-execute within a Virtual Machine (VM). This approach allows for making open-source software requiring complex dependencies and installation procedures widely accessible to the research community. A VM is essentially an emulation of a compute server, with virtual processors, memory and storage capacity, in the form of a single binary file that can be executed on top of a virtualization layer running on a physical compute server [6]. Virtualization technologies led to the development of cloud computing services, where remote computer server farms can be rented on an hourly basis by researchers and used for scalable, on-demand computation. Cloud services offer high-performance computer hardware with a virtualization layer, upon which a user executes VM servers [7]. Renting computational capacity from these services has the potential to eliminate many of the upfront capital and effort expenses of building technology infrastructure for genome sequencing informatics, and as result transform the analysis and data processing tasks into well defined operational costs.

Current State of Cloud Computing For Bioinformatics

In recent years, cloud based bioinformatics data analysis systems such as Galaxy [8], CloVR [9], Cloud BioLinux [10] and BioKepler [11] were released, allowing smaller laboratories and institutions to perform large-scale data analysis with genomic datasets. All these platforms are available on the Amazon Elastic Compute (EC2) cloud [12] , which provides data centers in US East and West regions, European Union, Asia and Australia [13], allowing any researcher worldwide to connect to the cloud end-point closest to their geographic boundaries.

The Amazon cloud allows users to run Virtual Machines (VM) with various capacities in terms of the underlying physical compute resources allocated to the VM, and the usage is billed on an hourly basis [14]. This particular Cloud service consists of thousands of computer servers with petabytes of storage, leveraging economies of scale to achieve low operational costs that in turn offers as savings to users. Furthermore, the Amazon Cloud has data centers in US East and West regions, European Union, Asia and Australia [13], providing researchers worldwide with the ability to tap into a large pool of computational resources, outside of institutional, economic or geographic boundaries.

Researchers with access to a computing cluster at their home institution have the option to implement their own private Cloud platform and run VM servers build locally or downloaded from remote Clouds, using the open-source Eucalyptus [15] or OpenStack [16] Clouds. OpenStack is the official Cloud bundled with the Ubuntu Linux operating system [17] and can be easily installed on clusters running other Linux versions [18], and similarly for Eucalyptus [19]. Both of these Cloud platforms offer identical Application Programming Interfaces (API) with Amazon EC2, and applications developed either on Eucalyptus or OpenStack work seamlessly on the Amazon Cloud and vice-versa. Furthermore, users have the option to copy and execute VM server snapshots across installations of these Clouds as it has been demonstrated for the Cloud BioLinux VM [10]. For users who do not own a local compute cluster, neither have the available funds to lease computing time from Amazon, the government-funded Magellan cluster [20] provides an OpenStack Cloud where they can apply for a account. In addition, a number of academic computing centers in both the US and other countries have similar clusters with OpenStack installed [21], where scientists could request access, in addition to the the Eucalyptus Community Cloud [22]. Finally, an option for users is to to run the VM servers on a desktop computer, using virtualization software such as VirtualBox [23], that is also open-source and can be installed in a single step on Windows, Mac or Linux computers. The Cloud BioLinux project for example, provides VM server snapshots that run on both private and Commercial clouds, and VirtualBox [24].

We have recently expanded Cloud BioLinux by adding support for advanced users through a VM development framework for building and distributing customized bioinformatics virtual machines, adding to the community aspect of the project and enhancing the available features for collaborative bioinformatics VM development. For this purpose we implemented a modular software management system that simplifies the process of selecting the bioinformatics tools to be included in the VM, automates building of a new VM with the specified software, and seamlessly deploys across different cloud platforms. The overall goal is to offer a system for maintaining a range of specialized VM setups for serving different computing needs within the bioinformatics community, and allow researchers to focus on the next challenges of providing data, documentation, and the development of scalable analysis pipelines.

Open-Source Cloud Computing PlatForms As Private Clouds

Architecture of Eucalyptus Clouds and Installation at J. Craig Venter Institute

The Eucalyptus open-source cloud [15] offers a fully compatible Application Programming Interface (API) with Amazon's EC2 cloud, including the Simple Storage Service (S3) [25], Elastic Block Store (EBS) [26] and additional services such as Auto Scaling and CloudWatch. The Eucalyptus API is installed and runs on a Cloud Controller (CLC) physical machine, and both Eucalyptus-specific tools are available for interacting with the API [27] while the Amazon tools [28] are fully compatible as well. The CLC is essentially a software layer that besides serving the API and allowing interaction with the cloud, also tracks all data on the status of physical and virtualized computing resources, in addition to being the coordinator for the other Eucalyptus service that constitute the cloud. For large cloud installations, more than one CLC instances can be in place across different physical machines, ensuring scalability and load balancing when a large number of users interact with the cloud API, while also safeguarding the cloud operations in case of hardware failure.

In more detail, one of the cloud's software layers providing information to the CLC is the Cluster Controller (CC). The CLC delegates to the CC the task of gathering statistics from the physical nodes of the compute cluster running the Virtual Machines (VM), such as available disk space, number of VM running on each physical node, and compute load. In large cloud installations, similarly to the CLC multiple CC instances can run concurrently on different physical machines for fail protection and to balance the load of keeping track of tens of VMs being started, terminated and accessed by a large number of users. The cloud installation at the J. Craig Venter Institute (JCVI) is comprised by four identical physical compute servers (16 core, 32GB memory, 1TB disk each), and one was used as the cloud's head node to run a single instances of the CLC and CC components.

The Amazon S3 storage corresponding service on Eucalyptus is called Walrus, and both deviate from the typical file system of disk drives, as they instead use data objects that can be accessed by internet-identifiable unique Uniform Resource Location (URL). Specifically, a file with public permissions on a Walrus service for a cloud that resides outside of an institutional firewall, can be accessed by simply pasting its URL on a web browser, but for upload, modification, or for bulk data operations or for files that require authentication, one of the two Application Programming Interfaces (API) must be used to access the service: the first is based on the Representational State Transfer protocol (REST, [29]), while the second on the Simple Object Access Protocol (SOAP, [30]). Alternatively, users can interact with the service through various desktop client applications that provide a graphical front-end, available for all different operating systems [31–33].

Due to its front-facing role in serving data objects through identifiable URLs over the internet, Eucalyptus is designed to run the Walrus service from the same physical node of the CLC, which provides access to the cloud's API for the user's applications. Walrus can be also accessed from a VM running inside the cloud, in the case a user application is requesting data objects from the service via their URLs. At JCVI's cloud, we run the Walrus service on the same physical compute server as CLC and CC, the simplest configuration available for Eucalyptus given the small size (four server nodes) of our cluster. According to recommendations on the Eucalyptus documentation [34], for larger cloud instances the Walrus physical storage should be connected a Network Attached Storage (NAS, ref) system, allowing scalability during high user loads and data object requests.

Another role of the Walrus is to serve as the VM image store, where compressed splits of binary files composing a VM template are stored, copied and joined at the physical node that boots the VM. In our experience with the small Eucalyptus cloud where a NAS was not used but rather we run Walrus within the limitations of the a single disk drive on a physical server, and Eucalyptus copies binary splits of the VM template to the physical nodes during VM booting, the process times out and fails for VM larger than 100GB in size. We have found that by using the caching mechanism available in Eucalyptus and placing a VM template from Walrus on the CC [35] allowed us to start VMs of size 200GB and up within a few minutes.

The Node Controller (NC) component of Eucalyptus, is the cloud layer that boots and runs the VMs on the physical compute servers of the cluster, and constantly provides information to the CC on the available compute resources and number of VMs on the server. At JCVI we have configured our Eucalyptus cloud with 4 NCs, and therefore we essentially provided a master physical server running all cloud components (CLC, CC, Walrus, NC), and 3 worker servers running a single NC each.

The workflow involving the different Eucalyptus cloud components discussed so far to boot a VM is as following: using the command line tools available either by Eucalyptus or Amazon or any of the available client applications, a call is made to the cloud API through the CLC for booting a VM with specified computational capacity (within the limits of the physical compute resources, and as allowed from the cloud's administrative interface); the CLC checks its registry for the availability of resources, and routes the request to a specific CC (if more than one CC are available), that has under its management a set of NC with available resources to run the VM with requested capacity; the CC checks whether the VM is cached locally and if not a copy is initiated from Walrus and passed over to the NC, while an unique IP address is assigned to the VM; finally, once the NC boots the VM its IP address and periodic updates of its status are communicated back through the same chain to the CLC and the user's application through the API.

The last component of Eucalyptus is the Storage Controller (SC), which corresponds to the Elastic Block Store (EBS) storage option on the Amazon cloud. This provides virtual disk drives with identical filesystems to physical drives (POSIX-compliant [36]), multiples of which can be attached to a running VM. While the Amazon EBS provides virtual disk drives up to 1 TeraByte (TB), on Eucalyptus the maximum size can be set through a configuration parameter and also depends on the space available on the underlying physical hard drive on the SC machine. To the best of our knowledge no upper limit to the sizes of the virtual drives has not been established for Eucalyptus cloud system, and in theory the size of a single virtual data volume can match the remaining capacity of the underlying physical hard drive. For JCVI's cloud where the SC had available a 1000GB drive, we have successfully started two EBS volumes at 250GB, one at 150GB and a few smaller ones without any performance problems.

The EBS volumes persist past VM termination and be re-attached to a new VM, making them ideal for storing persistent data such as relational or bioinformatics databases. An already attached EBS volume cannot be shared between VMs, unless sharing is through a software layer such as the Network File System (NFS) is used on the VM that has the volume attached. Users can create snapshots from EBS volume, which are placed on the Walrus storage first providing backup, while also can be used as template images to create multiple identical copies of the original EBS volume. Furthermore, on the Amazon cloud, EBS snapshots are copied in triplicate across three data centers in different geographic zones of the S3 storage.

EBS volumes can be also used as the main file system of a VM, allowing to use large-sized bootable VMs without the timeout issue we observed when using using Walrus-backed VM templates. The disadvantage is that EBS volumes are placed on the SC that on the Eucalyptus cloud resides on a different physical server than the NC. Therefore, only the memory and CPU of the physical server running the NC service is utilized for the VM, while the EBS storage has to be attached to the VM over the network from the SC. This has the disadvantage of significantly increasing the network traffic within the cluster that runs the cloud, while creating an additional bottleneck based on the storage available on the SC. This approach does not leveraging the scalability offered with addition of local storage when more physical servers running NCs are added to the cluster as worker nodes.

Finally, Eucalyptus provides a VMware Broker enabling deployment of VMs on VMware clusters, commonly found on institutional infrastructures. The VMware Broker works by mediating all interactions between the CC and VMware hypervisors layer on the cluster, essentially treating the hypervisor as a NC that is passed instructions to start or stop VMs given by user application through the cloud API.

OpenStack

Enis and collaborators. Describe the OpenStack architecture and components (Nova, S3 etc) that it provides. Example private cloud installations that use it.

Results and Discussion

Porting the CloudBioLinux-CloudMan Framework to Private Clouds

The CloudMan Implementation For Scalable Computing on The Cloud

Enis Afgan: technical description of CloudMan as it stands for Amazon EC2. For example it uses two data volumes, pulls the code on boot from S3, how does it work with snapshots etc. This will serve as leeway for when describing how we mapped these components to the other platforms, see below.

Porting CloudBioLinux-CloudMan to Eucalyptus

Alex, technical description of changes to the CloudMan EC2 codebase to run on Eucalyptus. How components that CloudMan uses on EC2 where mapped to Eucalyptus. Specific nagging that Eucalyptus does with running CloudMan - i.e. different responses from the API returned when doing calls via boto, and changes in the cloudman codebase in order to take care of these. Also is there something that sysadmins that set up Eucalyptus should pay attention too (in the settings, in order to leverage those applications that create scriptable infrastructures by calling the API)

While adapting Galaxy-Cloudman for use on the Eucalyptus private cloud, we ran into problems common to migrating between systems that both track an evolving standard; some features of AWS are not implemented in Eucalyptus, and some have differences in implementation. The primary absence was the ability to add arbitrary tags (key-value pairs) to running instances (virtual machines) or volumes (virtual disk drives). The primary difference in functionality is that AWS uses Xen virtualization, while Eucalyptus uses KVM virtualization. This meant that, while AWS can attach a new volume to an instance at an arbitrary device id (e.g. /dev/xvdf), Eucalyptus will always attach at the next available device id (e.g. /dev/vdb), regardless of what id was specified in the attachment request. We were forced to write code to emulate tagging and detect attachment device ids within the Cloudman infrastructure.

Porting CloudBioLinux-CloudMan to OpenStack

Enis Afgan: technical description of changes to the CloudMan EC2 codebase to run on OpenStack. Changes that have be made to code in order to communicate with the API correctly.

Execution and Performance of Bioinformatic Pipelines on Private Clouds

An HMM-BLAST Pipeline for Metagenomic Annotation at JCVI

Ntino... I got the numbers, need to write it.

A Variant Calling Analysis Pipeline at HSPH

Enis Other Example of Pipeline

Conclusions

. . .

Author's contributions

Alex Richter and Konstantinos Krampis worked on the technical aspects of porting CloudMan to the Eucalyptus Cloud. Enis Afgan and .... worked on porting CloudMan on the Openstack Cloud. Brad Chapman implemented the parallel sequence data analysis pipelines used for benchmarking the two different Clouds.

Acknowledgements

Text for this section . . .

References

1. Mason C, Elemento O: **Faster sequencers, larger datasets, new challenges**. *Genome Biology* 2012, **13**:314.
2. Loman N, Misra R, Dallman T: **Performance comparison of benchtop high-throughput sequencing platforms**. *Nature Biotechnology* 2012, **30**:434–439.
3. Aziz R: **Subsystems-based servers for rapid annotation of genomes and metagenomes**. *BMC Bioinformatics* 2010, **11**(Suppl 4):O2.
4. *Using clouds for metagenomics: A case study*, IEEE 2009.
5. Wilkins-Diehr N, Gannon D, Klimeck G, Oster S, Pamidighantam S: **TeraGrid science gateways and their impact on science**. *Computer* 2008, **41**(11):32–41.
6. Uhlig R, Neiger G, Rodgers D, Santoni A, Martins F, Anderson A, Bennett S, Kagi A, Leung F, Smith L: **Intel virtualization technology**. *Computer* 2005, **38**(5):48–56.
7. Armbrust M, Fox A, Griffith R, Joseph A, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, et al.: **A view of cloud computing**. *Communications of the ACM* 2010, **53**(4):50–58.
8. Goecks J, Nekrutenko A, Taylor J: **Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences**. *Genome Biol* 2010, **11**(8):R86.
9. Angiuoli S, Matalka M, Gussman A, Galens K, Vangala M, Riley D, Arze C, White J, White O, Fricke W: **CloVR: A virtual machine for automated and portable sequence analysis from the desktop using cloud computing**. *BMC Bioinformatics* 2011, **12**:356.
10. Krampis K, Booth T, Chapman B, Tiwari B, Bicak M, Field D, Nelson KE: **Cloud BioLinux: pre-configured and on-demand bioinformatics computing for the genomics community**. *BMC Bioinformatics* 2012, **13**:42.
11. *Distributed workflow-driven analysis of large-scale biological data using biokepler*, ACM 2nd international workshop on Petascal data analytics 2011.
12. **Amazon Web Services Elastic Compute Cloud (EC2)** [htpp://aws.amazon.com/ec2].
13. **Amazon EC2 Cloud global regions** [http://aws.amazon.com/ec2/regions].
14. **Amazon EC2 Cloud Pricing** [http://aws.amazon.com/ec2/pricing/].
15. **Eucalyptus Open Source Cloud Platform**[http://open.eucalyptus.com].
16. **OpenStack Open Source Cloud Platform:**[http://www.openstack.org].
17. **Ubuntu Linux Operating System:**[http://www.ubuntu.com/Cloud].
18. **OpenStack Installation Documentation:**[http://docs.openstack.org/essex/openstack-compute/starter/content/CentOS-de1592.html].
19. **Eucalyptus Cloud Installation Packages**[http://www.eucalyptus.com/download/eucalyptus].
20. **OpenStack at Magellan Cloud:**[http://www.alcf.anl.gov/magellan].
21. **OpenStack Cloud installations**[http://openstack.org/user-stories/].
22. **Eucalyptus Community Cloud**[http://www.eucalyptus.com/eucalyptus-cloud/community-cloud].
23. **VirtualBox desktop virtualization software:**[http://www.virtualbox.org].
24. **Cloud BioLinux community site:**[http://www.cloudbiolinux.org].
25. **Amazon Simple Storage Service**[http://aws.amazon.com/s3].
26. **Amazon Elastic Block Store**[http://aws.amazon.com/ebs].
27. **Eucalyptus Cloud developer tools** [http://open.eucalyptus.com/wiki/Euca2oolsGuide].
28. **Amazon Cloud API tools**.
29. Fielding R: **Representational state transfer (REST)**. *Architectural Styles and the Design of Network-based Software Architectures. University of California, Irvine* 2000, :120.
30. **Simple Object Access Protocol (SOAP):**[http://www.w3.org/TR/soap].
31. **CuberDuck Amazon S3 cloud storage client**[http://cyberduck.ch/].

32. **S3browse** [http://s3browser.com/].

33. **Amazon S3 Firefox browser extension**[https://addons.mozilla.org/en-US/firefox/addon/amazon-s3-organizers3fox/].

34. **Eucalyptus Cloud Documentation**.

35. **Caching Virtual Machines on Eucalyptus**.

36. Mathur A, Cao M, Bhattacharya S, Dilger A, Tomas A, Vivier L: **The new ext4 filesystem: current status and future plans**. In *Proceedings of the Linux Symposium*, *Volume 2*, Citeseer 2007:21–33.