Computer Science • 13 (2) 2012 http://dx.doi.org/10.7494/csci.2012.13.2.103

J. Chwastowski
R. Grzymkowski
M. Kruk
M. Nabożny
Z. Natkaniec
A. Olszewski
H. Pałka
Z. Sobocińska
T. Sośnicki
M. Szostak
P. Syktus
M. Witek
P. Wójcik
T. Wojtoń
M. Zdybał
B. Żabiński

THE CC1 PROJECT – SYSTEM FOR PRIVATE CLOUD COMPUTING

Abstract *The main features of the Cloud Computing system developed at IFJ PAN are described. The project is financed from the structural resources provided by the European Commission and the Polish Ministry of Science and Higher Education (Innovative Economy, National Cohesion Strategy). The system delivers a solution for carrying out computer calculations on a Private Cloud computing infrastructure. It consists of an intuitive Web based user interface, a module for the users and resources administration and the standard EC2 interface implementation. Thanks to the distributed character of the system it allows for the integration of a geographically distant federation of computer clusters within a uniform user environment.*

Keywords virtual machine, cloud computing, private cloud, distributed computing system

1. Introduction

Providing computer infrastructure to end-users in an efficient and user-friendly way has always been a big challenge in the IT market. Cloud Computing is an approach that addresses these issues, and recently it has been gaining more and more popularity. A well designed Cloud Computing system delivers elasticity of the resources allocation and allows for the efficient use of computing infrastructure. The underlying virtualisation technology and the self-service type of access are the two key features that make the software independent of a specific hardware and enable a significant decrease in system administration effort. The growing popularity of Cloud Computing has led to the appearance of many open source systems offering such computing environments. Among them are Eucalyptus [1], OpenNebula [2], Nimbus [3] or OpenStack [4]. These solutions make it possible to construct a computing cloud in a relatively short time, and do not require a deep understanding of the virtualisation techniques and the network administration. The main drawback of using this type of toolkits is difficulty in its customisation to special needs. A significant effort is needed to implement some non standard features. In particular, the need for the re-implementation of the external toolkits in each new version of the base system makes the application development very hard and inefficient. Therefore, we propose a complete solution for a Private Cloud system with additional features suitable for scientific applications.

2. Project evolution

The CC1 Project at IFJ PAN started in 2009. The first step was devoted to a review of available cloud toolkits. Its aim was to select the toolkit with optimal functionalities to accomplish the project goals. At the time of evaluation in 2009 none of the tested systems (OpenNebula, Eucalyptus, Nimbus) offered a full set of required features. OpenNebula was chosen as the most promising package. After a successful implementation of a basic management of virtual machines (VMs), we faced problems during the implementation of more sophisticated features. One of the most important features missing was quota management. In Private Cloud, like CC1, which provides limited resources to multiple groups of users, quota management is needed for assigning limits for accessing CPU cores, memory and storage, and a proper accounting of their usage. Another feature missing was the capability to provide clusters of computer nodes instead of single machines. This is required by a modern physics and other scientific applications where data processing is so extensive that it requires running in an organized way on large assembles of computers, usually provided by local clusters, but which can be easily replaced by a cloud system. The deployment of a self-configuring "computing farm", including a ready-to-use batch system, was one of the main features of the system to be provided. In addition, the users should have the possibility to organise themselves into groups working on the same topic. This allows for a common use (sharing) of the VM images with installed dedicated

software. Needless to say, all the mentioned features should be accessible through a simple and intuitive Web based user interface. The VM management using popular interfaces like EC2 [5] or OCCI [6] was needed in order to automate and integrate our solution with external systems. The attempt to make an extension to the existing OpenNebula system with all the required features turned out to be a complicated and time-consuming process. In effect it would impose severe constraints on the scope of features to be implemented due to the fixed project schedule. Therefore, we made a decision to design and implement our own cloud computing system. The proposed solution for the CC1 is based on Libvirt [7], a lower level virtualisation toolkit which provides a fully functional set of VM management tools on a single physical node.

Python [8] was chosen as programming language for the project. A rich set of useful modules without the explicit dependence on the operating system made the implementation phase fast and allowed the software engineers to focus on the high level design process. The use of scripting language allowed for the on-line development and easy deployment of new package versions.

3. Structure of the CC1 cloud system

The schematic view of the system is shown in Fig. 1. The CC1 system was divided into two main layers: i) the access layer consisting of a Web based user interface (including administrator module) and standard EC2 plug-in, ii) internal layer for the management of the whole system. The characters of these two parts are complementary. The simplicity of the Web interface is achieved at the cost of a necessary complication of the underlying structure since it must take an appropriate decision in the case of various events or errors. This internal structure consists of two main functional blocks. The top element of the system is called the Cloud Manager. It receives all calls from the user interfaces (the Web browser based interface or the EC2 interface) and passes commands to the Cluster Manager. The Cluster Manager, running on each individual cluster, handles all low-level operations required to control virtual machines.

From the very beginning the system was developed to enable the geographically distributed federation of private clouds. The assumption was to provide a consistent administration of users, a mechanism for the replication of resources on demand (VM images and disk volumes) in order to achieve an optimal utilisation of resources within the federation. The basic elements of the distributed structure have been already implemented and tested. It includes the replication of user accounts, switching between sessions on various Cluster Managers and the separate administration of individual Cluster Managers.

3.1. User interface and administrator module

The Web based user interface has been designed to give easy access to the CC1 system and its resources. At present most of the planned features have already been implemented.

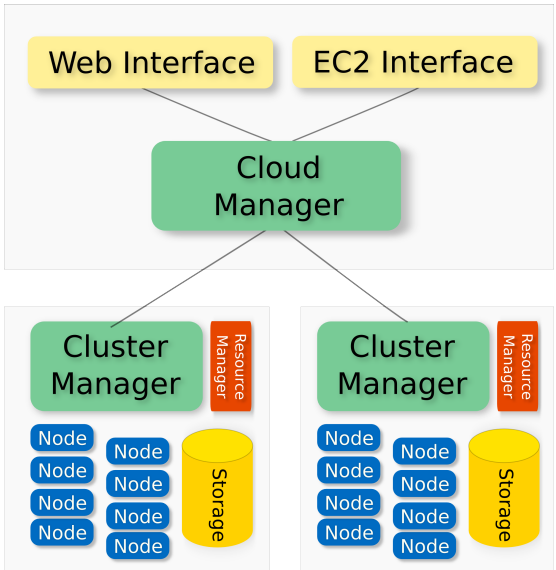

Figure 1. Overview of the CC1 system structure

User accessible functionalities include:

- virtual machine creation and management,
- elastic IP addresses – public IP addresses that can be assigned dynamically to a VM and then re-allocated on the fly to another VM,
- virtual storage – non-volatile disk volumes that can be attached to a VM,
- virtual clusters ("farms") – automatically created clusters of virtual machines with a pre-configured batch system for job processing,
- contextualisation – a mechanism to interact with a VM using direct communication between the Web interface and the VM operating system that allows for operations like changing passwords, injecting the SSH keys, performing system shutdown or executing a configuration script.
- groups of users – self-organizing groups of people with the ability to share VM images among themselves.

The main panel of the Web interface presented to the user after successful registration and authentication is shown in Fig. 2.

The dark menu bar contains sub-menus providing actions for active VM instances, saved VM images, VM resources (public IP numbers, virtual disks), and groups. A typical user session may consist of the following steps. First, the necessary resources are created on requests from the "VM resources" sub-menu i.e. virtual disk creation and IP public address allocation. Next a virtual machine can be created from public, group or private VM images. As is shown in Fig. 3, the desired parameters, like VM template (the required number of cores and RAM size), the name and description

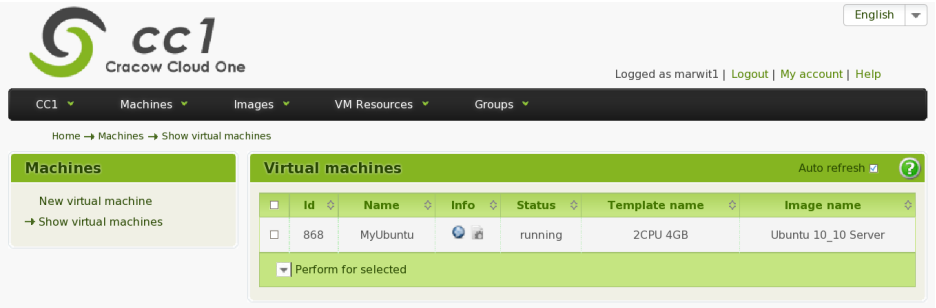

Figure 2. The main panel of user Web interface. The dark bar contains sub-menus for various system actions (from left to right: CC1, Machines , Images, VM Resources, Groups). Below, a list of a running VMs is displayed by the action invoked from "Machines" sub-menu (one running VM in the list)

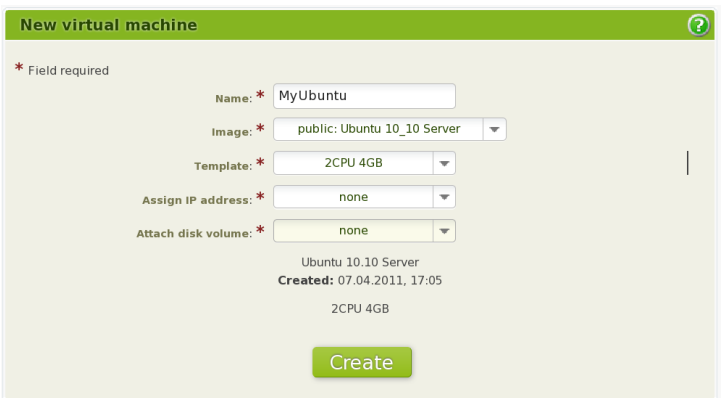

Figure 3. The panel for virtual machine creation

of the VM, have to be specified at the creation phase. The previously created disk volume and allocated IP address can be declared at this stage as well. After the successful start of a virtual machine the user can set a password or inject SSH keys and then log into the machine. When the work is completed, the VM can be saved or destroyed. The IP address is deallocated and the disk volume is closed.

Although in stable conditions the cloud systems should not require frequent intervention of the administrator, a good interface to the routine tasks, like system configuration, reaction to hardware problems, management of users is a big advantage. The CC1 administration module has a wide set of available actions. Authorized accounts may access the administrator panel which allows for:

- activation and blocking of user accounts,
- setting the quotas (CPU, memory, storage) for individual users,

- access to the accounting system based on points counted for the use of resources,
- creation or removal of VM templates
- configuration and deployment of a new local cluster,
- management of individual nodes of the cluster (configuration, activation, blocking)
- management of all active virtual machines and saved images
- real time monitoring of the use of resources.

The Web interface has multi-language support. Currently, English and Polish are implemented.

3.2. Cloud Manager and Cluster Manager

The Cloud Manager is an upper steering layer unique in the whole system that the Web interface interacts with, Fig. 1. The lower layer consists of a set of Cluster Managers – one for each of the physical clusters.

The role of the Cloud Manager is relatively "light" and focused on global features which are common for all Cluster Managers. It uses a central database containing basic information about the users and groups, a table of available Cluster Managers and their status and performs a restricted set of tasks related to the Cluster Manager as a whole. The execution of specific functions is delegated to a control process of the Cluster Manager, Cloud Manager acting only as a proxy for Web interface requests. The main tasks of the Cluster Manager include:

- Management of virtual machines. All the tasks along the life cycle of a VM are performed. The VM creation request invokes a scheduler to find the physical node that fits the request. Then, the VM image is transferred to a local disk of the node and a boot procedure for the VM is started. All information about the status of the physical node, status of VM and its resources are kept in the Cluster Manager database. The granularity of the database allows for the coexistence of nodes with inhomogeneous hardware and to some extent inhomogeneous software. The usage of Libvirt gives a good level of operating system independence. It also makes possible to run various Virtual Machine Managers on different cluster nodes, currently implemented KVM (default) and Xen.
- Management of user images. The virtual machine images and virtual disk images can be created by the users. The necessary functions for handling these tasks are provided (save, delete, edit description). Apart from public VM images available to all, a private VM image can be uploaded to the system and registered in the database.
- Dynamic assignment of public IP addresses. For optimal use of public IP addresses an elastic IP model is implemented. It allows for the dynamic assignment of public IP to already running VM instance without a need to modify network configuration. This allows for the circulation of one IP address over many VMs.
- Direct interaction with a VM instance via contextualization mechanizm (mentioned in Sec. 3.1).

3.3. Implementation of EC2 interfaces

The EC2 interface developed by Amazon for the management of their Elastic Compute Cloud has been adopted by many other cloud computing systems as a kind of a standard interface. It allows for the so-called "cloud bursting", giving a possibility to quickly extend resources from external systems to cover peak loads on a local system. The CC1 system already supports the management of AMIs (Amazon Machine Images), instances, key pairs, and elastic IP addresses. It is planned to implement also the support for the Elastic Block Store (EBS) volumes, the spot instances and security groups. The CC1 system supports the multiple, isolated zones managed by the Cluster Managers and Resource Managers (RM). Each zone has its own pool of a virtual machine configurations – the VM templates. The zone administrator (Cluster Manager administrator) can define the Amazon Instance Types in a way similar to the VM templates management.

The CC1-EC2 module accepts requests over the HTTP or HTTPS protocol. The authentication method is based on the signature generation. To generate a signature a special string has to be created. Its structure is similar to that of the HTTP request. It contains the access key id, the action, the signature info, the API version and also other parameters. Additionally, the request has to be signed with the AWS secret key. Only then can a request to the EC2 queries server be properly processed.

3.4. System security

Since the users are granted *root* access to the operating system of their VMs, the security of the cloud computing systems has to be kept at a high level. Its purpose is to cover three main security areas:

- security of the naitive operating systems running on the physical nodes of the cluster,
- security of the individual VM instances,
- security of the cloud computing system itself.

The first two areas can be covered by the standard security measures, passive and active protection mechanisms at the level of the whole cluster as well as individual nodes. The security of the operating systems of the VM is in the hands of the users. Although certain restrictions can be enforced like the use of strong passwords, proper settings of external firewall or active monitoring of network traffic, it has to be assumed that a compromised VM may appear in the system frequently. Therefore, proper isolation of virtual and physical resources has to be enforced, and the security of the functional blocks of CC1 system has to be ensured.

Access to the CC1 system is provided via the Web interface over a secure HTTP protocol. The communication between Cloud Manager and Cluster Manager proceeds by opening Web sessions authenticated by passwords. A significant effort has been put to minimize the risk of "man-in-the-middle" types of attacks for both Web access to the system and internal CC1 system communication. The interaction between the Web interface and VM operating system (contextualization) is encrypted. Although

the security of the CC1 system is already at a reasonable level, further improvements are planned.

4. Summary

In summary, a fully functional cloud computing system was designed and constructed at IFJ PAN, Krakw. Its implementation is well advanced, the project being close to reach its first milestone of making it available for use by the local community. Presently, the system is undergoing intensive testing. The production quality system (Private Cloud) will be made available to researchers of IFJ PAN at the beginning of 2012. The next step is to build a federated system with universities expressing their interest in the project.

Acknowledgements

The project was financed by grant POIG 02.03.03-00-033/09-04 from the structural funds provided by the European Commission and the Polish Ministry of Science and Higher Education (Innovative Economy, National Cohesion Strategy).

References

[1] Daniel Nurmi, Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, Dmitrii Zagorodnov (University of California Santa Barbara), *Eucalyptus: A Technical Report on an Elastic Utility Computing Archietcture Linking Your Programs to Useful Systems*, UCSB Computer Science Technical Report Number 2008-10. Eucalyptus Project website: `http://www.eucalyptus.com/` and documentation therein.

[2] Borja Sotomayor, Rubén S. Montero, Ignacio M. Llorente, Ian Foster *Virtual Infrastructure Management in Private and Hybrid Clouds*, IEEE Internet Computing, vol. 13, no. 5, pp. 14–22, Sep./Oct. 2009. OpenNebula website: `http://opennebula.org/`.

[3] Nimbus Project website: `http://www.nimbusproject.org/doc/nimbus/`.

[4] OpenStack website: `http://docs.openstack.org/`.

[5] Amazon Elastic Compute Cloud website: `http://aws.amazon.com/documentation/ec2/`.

[6] Open Cloud Computing Interface website: `http://occi-wg.org/`.

[7] Libvirt – the virtualisation API: `http://libvirt.org/docs.html`.

[8] Python Programming Language – Official website: `http://www.python.org/`.

Affiliations

J. Chwastowski
Institute of Teleinformatics, Cracow University of Technology, ul. Warszawska 24, 31-155 Kraków, Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

R. Grzymkowski
Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

M. Kruk
AGH University of Science and Technology, al. Mickiewicza 30, 30-059 Kraków, Poland

M. Nabożny
Institute of Teleinformatics, Cracow University of Technology, ul. Warszawska 24, 31-155 Kraków, Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

Z. Natkaniec
Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

A. Olszewski
Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

H. Pałka
Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

Z. Sobocińska
Institute of Teleinformatics, Cracow University of Technology, ul. Warszawska 24, 31-155 Kraków, Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

T. Sośnicki
Institute of Teleinformatics, Cracow University of Technology, ul. Warszawska 24, 31-155 Kraków, Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

M. Szostak
Institute of Teleinformatics, Cracow University of Technology, ul. Warszawska 24, 31-155 Kraków, Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

P. Syktus
AGH University of Science and Technology, al. Mickiewicza 30, 30-059 Kraków, Poland, Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

M. Witek
Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

P. Wójcik
AGH University of Science and Technology, al. Mickiewicza 30, 30-059 Kraków, Poland, Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

T. Wojtoń
Institute of Teleinformatics, Cracow University of Technology, ul. Warszawska 24, 31-155 Kraków, Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

M. Zdybał
Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

B. Żabiński
Institute of Nuclear Physics PAN, ul. Radzikowskiego 152, 31-342 Kraków, Poland

Received: 31.12.2011
Revised: 25.01.2012
Accepted: 23.04.2012