SURFnet cloud computing solutions

Universiteit van Amsterdam

Master of Science in System and Network Engineering

Marvin Rambhadjan (marvin.rambhadjan@os3.nl)
Arthur Schutijser (arthur.schutijser@os3.nl)

March 12, 2010

1 Abstract

SURFnet is the primary supplier of advanced networking to Colleges, Universities and Research Institutions. SURFnet (and partners) wishes to optimize its computing capacity, and hope to realize this with the use of cloud computing. With the rising interest in cloud computing, a lot of new techniques are being developed and SURFnet wishes advice about which technique best fits their needs. In this document we research the five most promising cloud platforms. They are compared on their offload ability, live migration options, high availability options and others. A conclusions is drawn out of of this comparison, which cloud technique best fits the profile needed for SURFnet and its institutes/partners. From this comparison we describe several use cases in which this cloud platform would be useful.

Contents

1 Abstract . . . 1
2 Introduction . . . 4
2.1 General description of the project . . . 4
2.2 Goal and research questions . . . 5
2.3 Outline of this report . . . 5
3 Cloud computing . . . 6
3.1 Service models . . . 6
3.2 Deployment Models . . . 8
3.3 Open standards . . . 9
4 SURFnet . . . 10
4.1 Combining resources . . . 10
4.2 Requirements . . . 10
5 Techniques/Platforms . . . 12
5.1 Available cloud techniques . . . 12
5.2 Researched cloud techniques . . . 13
6 Hypervisors . . . 15
7 Test environment . . . 17
8 Eucalyptus . . . 19
9 OpenNebula . . . 25
10 AbiCloud . . . 32
11 OpenQRM . . . 37
12 VMware vSphere . . . 41
13 Comparison . . . 45
13.1 Compare matrix . . . 45
13.2 Overall outcome . . . 47
13.3 Scenarios . . . 47
14 Deployment . . . 48
14.1 Networking . . . 48
14.2 User Interface . . . 48
14.3 Scheduling . . . 48
14.4 Multi EC2 Site Support . . . 49
15 Use Cases . . . 50
15.1 Green IT . . . 50
15.2 Private Cloud . . . 50
15.3 Hybrid Cloud . . . 51

15.4 Community Cloud 53
16 Future Research 55
17 Conclusion 56
17.1 Advice 56

2 Introduction

Cloud computing is becoming an important concept in the IT world. The idea has been around for quite a while but lately more and more people are getting interested in it. This is because it is a concept, which could improve the computing efficiency in datacenters and thus save money. A lot of similar solutions have been used to make large (collaborate) computation power possible. These are clusters, grids and distributed systems.

There are different kind of ways to use cloud computing. It is possible to use this technique to outsource a critical business application so it will run on a "public cloud". In this set-up you will only pay for what you use. There are many cloud providers which implement this kind of service. It is called Software as a Service (SaaS). Besides software it is also possible to create a network infrastructure based on a cloud (IaaS). This way computing capacity is used more efficient than the traditional set up of a datacenter. Further more cloud computing could be used to make your datacenter more *green* and create redundant servers more easily.

2.1 General description of the project

SURFnet would like to be able to share computing resources between them and their institutions/partners. This way it is not needed for institutions (departments/researchers) to build a very large datacenter to be able to handle large amount or peaks of computing requests. When sharing the computing resources it is possible to "borrow" computing resources from other institutions to handle the load when needed. With cloud computing this is possible. This is why SURFnet would like advice about possible solutions. In this project available cloud solutions[1] are researched to implement shared computing resources. If the resulting cloud solution is a great success for SURFnet they will advice this technique[2] to the institutions. All of the them are then able to be part of the cloud and use more resources from each other without having to purchase extra hardware, storage, power, etc. by reusing overcapacity of datacenters. In this project we will not cover the ability to offer some sort of billing system for shared resources as SURFnet wants a cloud for their own internal services and share unused resources.

During this project we will come across many virtualization techniques, this is an important aspect of cloud computing. Although we will not cover this in too much depth as many information on this can be found on the internet and we are more interested in the available cloud platforms.

[1]Because all techniques treated in this project are relatively new, the available documentation and information is limited. This is why the Internet overall is our biggest supplier of information. This is the reason we quote a lot of people and websites and base our decisions on these sources.

[2]In this paper we will use both "technique" as "platform" to refer to the different programs/implementations available to implement a private cloud. In most cases "platform" will be used.

2.2 Goal and research questions

The research question for this project is:

Which Cloud Computing platform meets the requirements best, to combine multiple clouds to create a hybrid cloud for SURFnet?

Within this project the most promising cloud computing solutions (for SURFnet), which are available at the moment, are researched. These solutions are compared on their functionality (the full list can be found in section 4.2). At the end of this project an advice is given to SURFnet about the best solutions for them and how to realize their goal as stated before.

2.3 Outline of this report

In the next section cloud computing is fully explained which will cover the differences between the types of cloud techniques and how it is used. After this the requirements of SURFnet are listed (section 4.2) and a description of what SURFnet hopes to achieve with cloud computing. Based on these requirements a handful of available technique are chosen in section 5 which will be looked into. Finally all techniques and platforms are compared with each other to be able to give the best possible advice to SURFnet which can be found in section 17 (conclusion).

3 Cloud computing

The name cloud computing comes from the fact that in network designs large parts of a network are drawn as a cloud. For example the Internet is most often displayed as a cloud. It is displayed like this because you are not sure what is in there, and have no control over it. But when you put some data in you get some data out. This is the same with cloud computing, you are not sure where the servers (you are communicating with) are located but you get answers you requested back.

NIST defined cloud computing as following "Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models"[6]. The characteristics refer to On-demand self-service, Broad network access, Resource pooling, Rapid elasticity, and Measured Service.

The technique behind *"Cloud Computing"* is based on a collection of many old and a few new concepts in several fields, like computer grids, distributed systems as well as virtualization. Cloud computing has created much interest in the last few years because of its ease to scale your resources. It is possible to expand or shrink your resources on the fly and only pay for the resources you use. Cloud computing is a new fase in utility computing, which means packaging of computing resources, such as computation and storage, as a metered service similar to a traditional public utility (such as electricity, water, natural gas, or telephone network) [1].

It is possible to build a cloud on top of the existing hardware in a datacenter. Most cloud solutions rely on virtualization techniques. It is through these techniques, resources could be made available so quickly to customers. With the use of middleware all servers within a datacenter (or multiple datacenters) could be combined to act as a single pool of resources and deliver this to customers. These two techniques are the fundamentals which make cloud computing possible and such a success [2].

In the next section we will cover the different cloud services and possibilities.

3.1 Service models

Cloud computing has several layers in which you can operate. The more basic (and most used) layers are: software, platform and infrastructure. We will explain these three layers in the next section in more depth. These layers could run on top of each other, with software as the highest level and infrastructure at the bottom (Figure 1). This latter one could then use virtualization techniques to work more efficient (we will get back to this later on in this chapter).

Besides the three layers just mentioned, there are also other types of services like Data-Storage as a Service (DaaS) and Communication as a Service (CaaS).

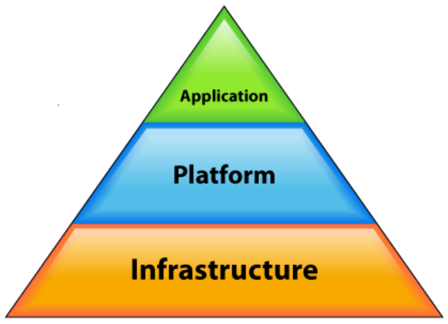

Figure 1: Cloud triangle.[3]

We will not cover these two in this paper only the more common and (for SURFnet) interesting ones, namely:

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

SaaS

With SaaS a company outsources an application to a SaaS provider. Applications are maintained and managed within the cloud of a provider. A company can rent this service for its own use. It is common in these kind of set-ups you pay only for what you use. Because this is build on top of a cloud, resources could be extended easily to customers needs.

SaaS exports the computational load from the users terminal to datacenters where cloud applications are deployed. This in turn decreases the restrictions on hardware requirements needed for the end-users, because they do not need to have big computing power, as this is done in the (remote) datacenter. This all means no upfront investment in servers and licenses [5].

Some examples of Software as a Service are: *Salesforce Customer Relationships Management (CRM) system* and *Google Apps*.

PaaS

"This form of cloud computing delivers development environments as a service. You build your own applications that runs on the provider's infrastructure and are delivered to your users via the Internet from the provider's servers."[5]

"The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations"[6].

Some examples of Platform as a Service are:*Salesforce Apex language* and *Google Apps Engine*.

IaaS

This layer is (mostly) used in combination with hardware virtualization. It is used to deliver customers on-demand virtual machines where they can run their own services on. It is possible for customers to build their entire infrastructure (processing, storage, networks, and other fundamental computing resources) on this cloud layer.

Several IaaS providers offer the ability to build an entire computer infrastructure on their cloud, Amazon Web Services deliver such a service[3]. Users are able to manage their infrastructure through an API delivered by the provider.

"The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls)."[6]

Some examples of Infrastructure as a Service are: *Amazon Web Services* and *GoGrid*.

3.2 Deployment Models

Cloud computing has several deployment models. Here are the definitions defined by NIST [6]:

Private cloud The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise.

Community cloud The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exist on premise or off premise.

Public cloud The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.

Hybrid cloud The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load-balancing between clouds).

[3]Amazon offers an IaaS public cloud to their customers under the name Amazon Elastic Compute Cloud (EC2), they also offer other cloud services but this is the most interesting for us. Through an API customers are able to manage their virtual machines running on this cloud.

3.3 Open standards

As cloud computing is making its entry into the world several open standards are introduced. These are meant to give guidelines to implement or develop cloud computing as best as possible. These standards are reviewed by multiple parties and (most) are free to use. When holding to these guidelines it will improve the overall product as these standards are based on best practices. Many of the open standards for clouds computing today are still in development but we will give a short highlight of the most interesting ones as found on cloud-standards.org[7]:

Cloud Security Alliance (CSA) "The Cloud Security Alliance was created to promote the use of best practices for providing security assurance within Cloud Computing, and provide education on the uses of Cloud Computing to help secure all other forms of computing".

Open Virtualization Format (OVF) "This specification describes an open, secure, portable, efficient and extensible format for the packaging and distribution of software to be run in virtual machines." This is managed by Distributed Management Task Force (DMTF).

The European Telecommunications Standards Institute (ETSI) "The focus is on scenarios where connectivity goes beyond the local network. This includes not only Grid computing but also the emerging commercial trend towards Cloud computing which places particular emphasis on ubiquitous network access to scalable computing and storage resources".

Open Cloud Computing Interface (OCCI) "The purpose of this group is the creation of a practical solution to interface with Cloud infrastructures exposed as a service (IaaS).". This is done by Open Grid Forum (OGF).

Object Management Group (OMG) "OMG's focus is always on modelling, and the first specific cloud-related specification efforts have only just begun, focusing on modelling deployment of applications & services on clouds for portability, interoperability & reuse".

Open Cloud Consortium (OCC) "development of standards for cloud computing and frameworks for interoperating between clouds, develops benchmarks for cloud computing and supports reference implementations for cloud computing, preferably open source reference implementations.".

Organization for the Advancement of Structured Information Standards "OASIS drives the development, convergence and adoption of open standards for the global information society".

4 SURFnet

"SURFnet is the primary supplier of advanced networking to Colleges, Universities and Research Institutions. Their main goal is to share (multimedia) information and data by connecting institutes (higher educations and research groups) in a safe and secure way. SURFnet is constantly searching for new innovations to improve their services" [8].

4.1 Combining resources

More and more companies want to see if cloud computing is a solution for their resource needs, the same with SURFnet and institutes (like other NREN[4]). As there are a lot of (big) cloud computing providers which offer interesting services to outsource for example office like applications, cloud computing is becoming more interesting. Companies like Google offers services like *Google Apps* (SaaS) and *Google Apps Engine* (PaaS). Customers (SURFnet and institutes) can use these services with less people for maintaining everything and with less computational resources needed within the company, which can save on the total cost.

Problems with these type of services are that companies want services managed internal, because there might be sensitive information involved. You want to know what happens with this kind of information when these services are outsourced to for example the United States.

The current state of affair with company (including SURFnet and its institutes) resources is that they are all spread over several departments and are not used to their full potential. Extra resources are also added (bought) by one department while others are only using a small portion. This is why outsourcing certain services to a public cloud becomes interesting. But because of legal issues and sensitive information this is not always an option. This is why SURFnet wants to implement cloud computing for themselves and reduce costs by sharing resources.

Because there already are large public cloud providers, which offer (for reasonable costs) the ability to offload services/requests to a public cloud, it would be useful to include these. Of course this is not always possible due to sensitive information but it would improve the overall flexibility of the cloud. SURFnet would like to have this option as a backup possibility.

4.2 Requirements

To sum up all requirements of SURFnet we set the following criteria to the available cloud computing techniques. Based on these criteria several cloud techniques and platforms are researched and compared.

[4]A national research and education network (NREN) is a specialised internet service provider dedicated to supporting the needs of the research and education communities within a country.[9]

An internal private cloud By using cloud technology they want to merge computing of several departments and/or other institutes to avoid unused overcapacity and share resources.

Offloading to other private cloud(s) Using their own cloud, they want to connect with other private clouds (one or more of their institutes) to be able to have more resources when needed, and the other way around.

Offloading to public cloud(s) If other private clouds can not handle the requested computing requests they would also prefer if it would be possible to offload some load to a public cloud like Amazon.

Through this project SURFnet is given advice by us on what cloud solution to implement which meets the requirements. With this cloud solution SURFnet will be able to design and implement a private cloud. After this seems a success for SURFnet, institutes connected to them could also implement this solution which enables the possibility to combine multiple private clouds into a hybrid cloud.

5 Techniques/Platforms

In this section we will give a general description about the available cloud platforms (which are available to us as this paper is written). From this list only a few are further researched. This section will explain all the relevant and important features of the researched platforms.

Given the requirements by SURFnet, we only focused on *Infrastructure as a Service* (IaaS) platforms. Only IaaS is able to deliver a private cloud as SURFnet wishes. All other cloud implementations generally make use of public cloud and their providers, and only use specific services like office applications. As described in the requirements (section 4.2) SURFnet would like to build a computing infrastructure on a cloud, so an IaaS is the solution for them.

5.1 Available cloud techniques

As cloud computing is rising, more and more companies try to profit from it. Large parties like Microsoft, Google, and Amazon try to offer cloud services. Also smaller (less known) parties come with their own cloud solutions both public and private. To find the best cloud solution for SURFnet we considered and looked into several cloud solutions and came with the following list[5]:

- 3Tera (AppLogic)
- AbiCloud
- Amazon EC2
- Aserver/DAAS
- Deltacloud
- Enomaly's Elastic Computing Platform (ECP)
- Eucalyptus
- Flexiscale
- Gandi
- GoGrid
- IBM cloudburst
- Nimbus
- OpenNebula
- Openqrm

[5]Besides these are a lot of other cloud products (as more and more are being developed) but we found these more interesting for SURFnet than others. As they are well known and/or offer a specific interesting service.

- Rackspace cloud (Mosso)
- Ubuntu Enterprise Cloud (UEC)
- VMware vSphere

5.2 Researched cloud techniques

Out of all available platforms, listed above (section 5.1), we made a small list of the platforms we will look into in more depth. As can be read in section 4.2 the platform has to match certain requirements.

We have chosen together with SURFnet the following platforms to research further. It came down to these because they are the most promising solutions as they offer most of the required features; supported operating systems, coöperation with other cloud/virtualization platforms, and have the biggest potential.

- AbiCloud
- Eucalyptus
- OpenNebula
- OpenQrm
- VMWare vSphere

Due to the limited time and documentation found about all platforms (because all platforms are relatively new) we have to make assumptions about the supported features. We assume if important features like *Live migration* (explained below) are not documented it is not supported by the given platform. We will compare the platforms on the following features[6]:

API We take into account if the platform supports multiple API's or an open standard for this.

Availability zones Is it possible to isolate parts of the cloud for specific purposes or customers.

Fault tolerance / fail-over Is there a feature to handle failing hardware? Running two instances of the same virtual machine (VM) on different physical servers, when one physical server fails the VM on the other server will take over. The same can be done for physical machines by making them redundant.

Live migration Moving a virtual machine while it is still running. Migration of virtual machines is always possible when it is shutdown, but there would be a high level of flexibility if this could be done while running.

[6]Techniques like Fault tolerance and Live migration rely on the used hypervisor. When a hypervisor supports this it could be used, although it will depend on the cloud platform if it will support this feature also as it needs to know if a instances is moving.

Monitoring Is there an ability to monitor running (virtual) machines, either with an internal tool or an external plug-in or program.

Multiple clouds Is there support for multiple clouds. Is it possible to offload processes to other private clouds or a (subscribed) public cloud (provider). This will be described at the start of every platform as it is an important feature.

Open Virtualization Format (OVF) "This is an open standard for packaging and distributing virtual appliances (software run within virtual machines). This standard is not bound to a hypervisor or processor architecture"[13]. Is this standard applied within the platform?[7]

Scaling Could the cloud infrastructure adapt to requests. Is it possible to expand or shrink the resources automatically when needed or can this easily be done by hand (Automatically shutdown or turn on virtual/physical machines when needed).

User management Can users be managed and assign privileges to them. Has the platform a graphical user interface to interact with the cloud?

Besides these features we will look into their marketshare, future development and overall potential. Based on all these point we will make a comparison found in section 13 to advice SURFnet.

[7]As this open standard is already implemented is some platforms we take this into account when comparing the platforms. When no other open standard is mentioned at the platform it is not supported (yet).

6 Hypervisors

The virtualization techniques that are used by the different cloud platforms from our research will be briefly explained below. We will not go into much depth, as there is enough knowledge about hypervisors within SURFnet and because this was not the main question of our research, altough this is an important layer in cloud computing.

Xen works directly on the physical hardware. It treats the old (host) OS as a guest OS. For Xen, Linux needs patches to make Linux a guest VM on the hypervisor.

KVM (Kernel-based Virtual Machine) is a patch to the Linux kernel, where the kernel supports a virtual infrastructure.

ESX and ESXi are “bare-metal” hypervisors. This means that they are installed directly on top of the physical hardware and partitions the resources of the underlying server.

VirtualBox is a general-purpose full virtualizer for hardware developed by Sun.

Linux VServer “is a virtual private server implementation done by adding operating system-level virtualization capabilities to the Linux kernel.”[44]

The VMware solutions ESX/ESXi both support their own features like vMotion, High Availability and others. These features are also supported when this hypervisor is used in an open source cloud solution. Although the configuration and management are supported by the VMware platform, it is not configurable within the open source cloud solution. This means that these features need to be used from outside the cloud solution.

Cloud	**Ubuntu Enterprise Cloud** v1.6.1	**Eucalyptus EE** v1.6.1	**VMware vSphere 4**	**abiCloud** v1.0.0-cr1	**openQRM** v4.6	**openNebula** v1.4
Platform	Xen, KVM	Xen, KVM, VMware	ESX, ESXi	VirtualBox, ESXi, Xen, KVM	Xen, KVM, VMware, Linux VServer	Xen, KVM, VMware

Table 1: Overview of supported hypervisors per platform.

In the table (2) below, only the interesting properties of the hypervisors are included. Mainly this mains that the VMs SURFnet is running are Microsoft Windows and Linux distributions. For this reason, the rest of the operating systems that are supported by the different hypervisors are not mentioned although this does not mean that this are the only operating systems that are supported by the hypervisors.

Name	**Developer**	**Host OS(s)**	**Guest OS(s)**	**License**
ESX	VMware	no host OS	Windows, Linux	Proprietary
ESXi	VMware	no host OS	Windows, Linux	Proprietary
KVM	Qumranet	Linux	Windows (needs v3.0), Linux	GPL version 2
Xen	Citrix Systems	Linux	(needs v3.0)	GPL
VirtualBox	Sun Microsystems	Windows, Linux	Windows, Linux	GPL version 2; full version with extra enterprise features is proprietary
Linux VServer	Community Project	Linux	Various Linux Distributions	GPL version 2

Table 2: Overview of hypervisors.

7 Test environment

For the five cloud platforms (5.2) we wanted to set up a test environment. The reason for this was for us to get some experience with them and get a feeling for there capabilities. For each of the platforms we wanted to test the following features:

- Manageable
- Virtualization features like, live migration and fault tolerance
- Offloading capabilities to another cloud
- Support for multiple guest operating systems

To set up these different platforms SURFnet offered us two servers within there test-datacenter. One of these servers had virtualization support from the CPU. This is why we used this server as the controller which ran all virtual machines. The network layout of the servers can be seen in Figure 2.

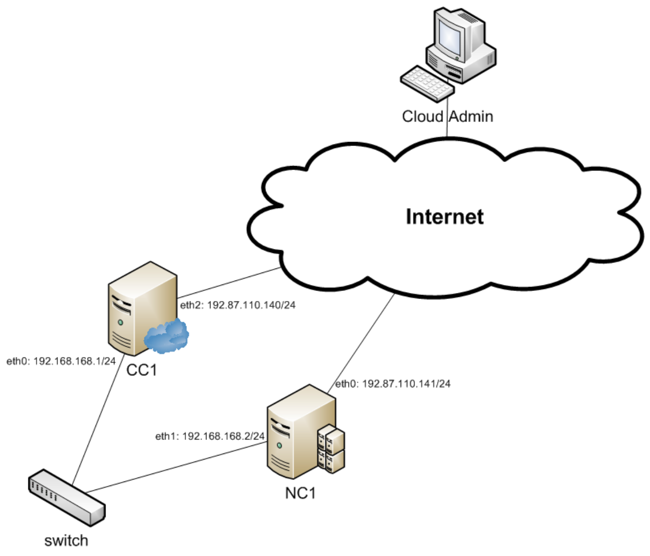

Figure 2: Server network layout.

Because these servers were unaccessable directly by us we later got laptops to test the platforms. This gave us some more flexibility due to the fact that we needed to contact someone in case something went wrong with the servers as there were behind closed doors to us. Both of these laptops had support for virtualization from the CPU.

The network layout of the laptops can be seen in Figure 3.

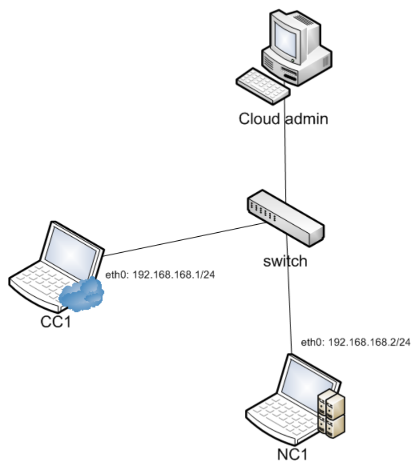

Figure 3: Laptop network layout.

As can be read later on we got some problems in setting up the platforms, this is why we used several different ways to set up a solid test environment. Besides the servers and laptops we also used virtual machines on our own laptops to test everything. This is why setting up all these environments costed us a lot of time and were not able to test all the features.

We will now describe the several cloud solution we tried. They are listed in the order in which we tested them.

- Eucalyptus
- OpenNebula
- AbiCloud
- OpenQrm
- VMWare vSphere

8 Eucalyptus

Eucalyptus is an open source software infrastructure for implementing cloud computing. It started as a research project in the Computer Science Department at the University of California, Santa Barbara [16]. Eucalyptus is an acronym for "Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems". "The EUCALYPTUS system is built to allow administrators and researchers the ability to deploy an infrastructure for user-controlled virtual machine creation and control atop existing resources. Its hierarchical design targets resources commonly found within academic and laboratory settings, including but not limited to small- and medium-sized Linux clusters, workstation pools and server farms" [17]. Eucalyptus tries to offer the same functionality as Amazon's EC2 services but within a private cloud and uses the same kind of (command-line) tools.

With Eucalyptus it is possible to build your own private cloud behind the firewall of your company. One of the strong points of Eucalyptus is that "it is highly modular and the API it uses is compatible with Amazon's AWS[8]" [18]. With this compatibility (with one of the largest public cloud providers) Eucalyptus offers great flexibility, this way you are not limited to your own private cloud and can easily switch.

There are two versions of Eucalyptus, the open source variant and a enterprise version. The main difference between the two is the support for VMWare's vSphere in the enterprise version[19] (more about vSphere in section 12). Currently the open variant of Eucalyptus only supports the virtualization techniques KVM and Xen. They hope to develop Eucalyptus further to support even more virtualisation platforms [17].

The open-source edition is available under a Free BSD license and the enterprise edition is a commercial product.

The edition we used to research was Eucalyptus version 1.6.1 as this was the latest version, at the moment of this project.

Product components

Eucalyptus consists of several components[16]:

- Cloud Controller (CLC): Is the entry-point into the cloud through a web interface for administrators, project managers, and end-users. The CLC is responsible for querying the node managers and virtualization of the underlying resources (servers, storage and network).
- Cluster Controller (CC): The CC can run on a cluster front-end machine, or any machine that has network connectivity to both the nodes and the CLC. It is responsible for managing a collection of node resources. It forms the front-end for each cluster defined in the cloud.

[8]Amazon Web Services (AWS) are a collection of services including cloud computing. Most of these services offer fast processing of large amounts of data.[20]

- Node Controller(s) (NC): Are the machines, which run the actual virtual machines. NC's control the execution, inspection and termination of VM instances on the host where it runs, fetches and cleans up the local copies of the instance images (kernel, root file system, and ramdisk image), and queries and controls the system software on its node (host OS and the hypervisor) in response to queries and control requests from the cluster controller.
- Storage Controller (SC): implements block-accessed network storage (e.g. Amazon Elastic Block Storage (EBS) and is capable of interfacing with various storage systems, such as NFS, iSCSI etc.
- Walrus (put/get storage): allows users to store persistent data, organised as eventually-consistent buckets and objects. It spans the entire cloud and is similar to Amazon S3 in functionality.

  The interface is compatible with Amazon S3 and supports the Amazon Machine Image (AMI) management interface. It provides a mechanism for storing and accessing both user data and virtual machine images.

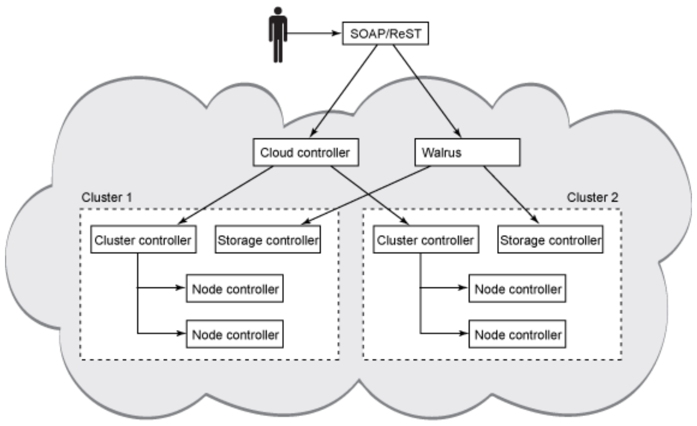

Figure 4: Topology of a multi-cluster Eucalyptus installation.[25]

Supported platforms

Host operating system "Eucalyptus runs on a wide range of popular Linux distributions, including Ubuntu, Red Hat Enterprise Linux, CentOS, SUSE Linux Enterprise Server, OpenSUSE and Debian"[21]. With the server edition of Ubuntu 9.10 it is implemented as the default cloud platform[22].

Guest operating system Most popular Linux distributions are supported as guest operating systems. It is also possible to run Windows as a guest operating system although this is not officially supported.

Hypervisor The supported virtualization techniques are: Xen and KVM. Only the Enterprise edition also supports vSphere. By default when installing UEC, KVM is used.

Features

"Eucalyptus is a cloud architecture that supports the same application programming interfaces (APIs) as public clouds like amazon" [21]. Eucalyptus is fully compatible with Amazon's Web Service cloud infrastructure. It is possible to offload applications to a public cloud like Amazon's EC2 or other cloud infrastructures. With the use of graphical tools like RightScale's web interface multiple clouds cloud be managed from a single location. [23] Eucalytus has a secure internal communication using SOAP with WS-security. [24] The topology of a multi-cluster environment can be seen in figure 4.

API Through the Amazon EC2 API Eucalyptus is able to be managed. As it tries to offer the same functionality as Amazon, Eucalyptus could also be managed by the same tools as Amazon.

Availability zones It is possible to build a cloud with multiple cluster controllers. Amazon has the ability to create availability zones by giving users the ability to determine where to place instances, this means clouds of customers can not access the clouds of other customers. Eucalyptus tries to offer a similar technique but in a different way. They separate zones by cluster so each cluster is a zone and bound to a single "datacenter" while Amazone's zones are much broader[16].

Monitoring "The latest release of Eucalyptus makes a variety of relevant information available which can be used in open source tools like Nagios and Ganglia"[21].

User management Basic "Cloud Administrator" tools for system management and user accounting. [24] There are tools like Elasticfox and Hybridfox to be able to manage virtual machine instances.

Limitations

Eucalyptus has the following limitations:

Max. controllers Currently Eucalyptus only supports a single cloud and walrus controller. So redundancy for these controllers is not possible. There is also no redundancy for the cluster controller.

Instance features Eucalyptus has no support for managing machines on a cluster with features as Live migration, Fault tolerance and Auto scaling are not supported.

Open Virtualization Format (OVF) This is not supported (yet) as no documentation or information can be found about this.

MarketShare

Large companies like NASA and Lilly (a large pharmaceutical company) implemented a private cloud with success[18]. There is much interest in Eucalyptus as it is one of the first open source private cloud products. At the moment companies are running demo environments to test Eucalyptus. There is also much support from its open source community. So the development of the product is going rapidly.

Future developments

During our research, there is very much interest in Eucalyptus from both enterprises as well as researchers. It is our believe the development of Eucalyptus will continue in coöperation with Ubuntu. As we could not find any concrete information about future developments we can only hope features still missing will soon be implemented and the community will keep growing as the interest rises.

Proof of Concept

Because there is a lot of interest in Eucalyptus this seemed to us the most promising cloud solution so we wanted to test this first. We used the two servers described above (Section 7). The test environment of Eucalyptus was as shown in Figure 5.

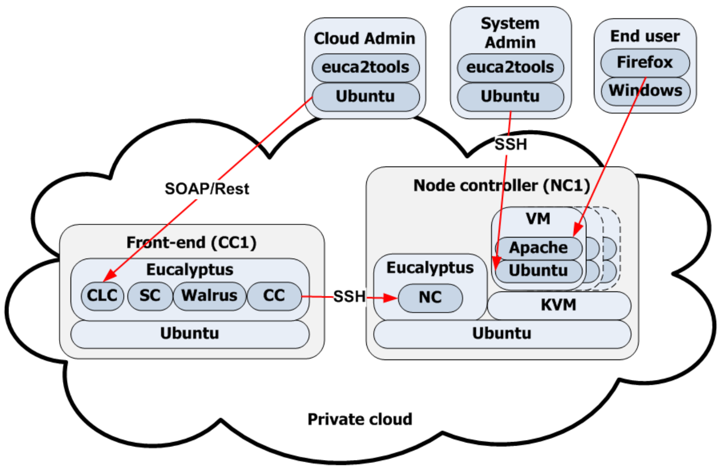

Figure 5: Demo environment of Eucalyptus(CLC: Cloud controller, SC: Storage Controller, CC: Cluster controller and NC: Node Controller).

The front-end of Eucalyptus consists of several processes. All of them could be split up and run on different servers in order to distribute the components of the infrastructure. This could also be done for performance reasons. We only used one server for the Front-end as our goal was not to measure the performance or availability. On both servers we used Ubuntu 9.10 as this came packed with Eucalyptus by default. This is also one of the reason we believed Eucalyptus could be the best solution for SURFnet as it got good support from the large community of Ubuntu.

For the Node Controller we used the virtualization technique, KVM. When installing Eucalyptus with Ubuntu, KVM is used by default and for our test environment this was good enough. With the tools euca2tools we were able to manage the Eucalyptus cloud. Their are also several other tools like Elasticfox to manage instances running on your cloud. We used Elasticfox for a moment but preferred the standard tools as these gave us more control.

Installing of Eucalyptus was fairly easy as it can be done during the installation of Ubuntu. Configuring Eucalyptus on the other hand was a whole other story. While configuring we ran in to a lot of problems, these include:

- Connecting the node controller to the front-end.
- Running one or more instances
- Connecting to a running cloud instance

We spend a lot of time on getting Eucalyptus working. We installed an image from the image store that is implemented in the Eucalyptus web interface. When we tried to start the image, we got the error message: Not enough resources. So we were not able to run the installed image after following the tutorial.

We looked in logs of both machines and noticed problems from the connection between the Front-End and the Node. So we made sure the authentication on both servers was done right. We created new keys for the Front-End and copied them to the authorized_keys file of the node. After this the problem of "not enough resources" was fixed, but the next problem came up; The instance was started and after a time pending, it terminated.

On the Internet we found more Eucalyptus users, which had the same problems and were running the same Ubuntu Enterprise Cloud version we used. Although nobody really had an explanation for the problem, it seemed like a bug. We looked at it with one of our supervisors (Paul Dekkers), which previously deployed an test environment with an older version of Eucalyptus. It looked like all these problems came from network problems. For some reason, these problems could not be fixed easily.

In the end we were able to get some instances running and connect to them. This gave us the ability to really test Eucalyptus. Our success did not last very long. Once we were configuring several instances and trying to create a Windows instance (As only Linux is supported by default) the server failed on us. For some reason we were not able to get Eucalyptus running again after this. We could not find the cause of the problem and decided to leave it and go on with another cloud platform.

We were able to test the following features of Eucalyptus:

Manageable With the given tools we were able to manage Eucalyptus easily once we had it running also with nice graphical tools like Elasticfox. With these tools the cloud could be managed from everywhere.

Virtualization features like, live migration and fault tolerance As we had only one Node Controller we were unable to test these features also due to the fact we got Eucalyptus only running for such a short time.

Offloading capabilities to another cloud Again due to the limited time and Eucalyptus not running properly we could not test this. Although we did not get everything running 100% we were able to see the possibilities Eucalyptus had. The only features we found of offloading was the migration to an Amazon cloud. This is very limited as SURFnet also wants to be able to offload to other private clouds[9].

Support for multiple guest operating systems By default Eucalyptus only supports Linux guests. We tested several different Linux guests with success (CentOS, Ubuntu and Fedora). After this we tried to set-up a Windows client although after building one we were not able to test this because Eucalyptus was not functioning properly anymore.

Result

Due to the very limited time we were not able to test Eucalyptus any further as we had three other platforms to test (We did not test vSphere as can be read in section 12). Nonetheless we were able to make a conclusion from the information we learned from our experience with Eucalyptus. The support for Eucalyptus is good as there are a lot of manuals on how to configure it, although problem-support is still limited as not to many people tested Eucalyptus yet and posted their results. We also noticed the offloading capabilities were very limited. Which is the main reason for SURFnet to create a cloud. Ubuntu Enterprise Cloud offers great potential, but for now, it is not mature enough to run it in a production environment. SURFnet first has to specialise its administrators for using Eucalyptus, because multiple times we encountered problems and they were hard to find the cause in logs and on the Internet.

[9]In the literature part of our research you can read that Eucalyptus only supports offloading to Amazon.

9 OpenNebula

OpenNebula is a standard-based Toolkit to build your Cloud. OpenNebula can primarily be used as a tool to manage a virtual infrastructure in a datacenter, which is usually referred to as a *Private Cloud.* It supports *Hybrid Cloud* to combine the local infrastructure with public clouds, which enables highly scalable environments. The private cloud can collaborate with both partner and commercial clouds [26]. For Hybrid Clouds, OpenNebula supports cloud plugins for Amazon EC2 and ElasticHosts connectors. These features in OpenNebula enable a scalable hosting environment.

The deployment of an hybrid cloud remains fully transparent to the infrastructure users. Users can still use the same Cloud interface. The federation is performed at the infrastructure level *(Infrastructure as a Service)*. "OpenNebula is distributed for use under the terms of the Apache License, Version 2.0."[26]

The version we used to research was OpenNebula version 1.4 as this was the latest version, at the moment of this project.

Product components

OpenNebula consists of three main components [30]:

- The Core: "a centralized component that manages the life-cycle of a VM by performing basic VM operations (e.g. deployment, monitoring, migration or termination). The core also provides a basic management and monitoring interface for the physical hosts".
- The Capacity Manager: "a pluggable module that governs the functionality provided by the OpenNebula core. The capacity manager adjusts the placement of VMs based on a set of pre-defined policies. The default capacity scheduler implements a simple matchmaking policy and supports user-driven consolidation constraints".
- The Virtual Access Driver: "Used to provide an abstraction of the underlying virtualization layer. It exposes the basic functionality of the hypervisor (e.g. deploy, monitor or shutdown a VM). Thus, OpenNebula is not tied to any specific environment, providing a uniform management layer regardless of the virtualization technology used".

Supported platforms

Host operating system OpenNebula supports the most popular Linux distributions (Ubuntu, Red Hat Enterprise Linux, Fedora, SUSE Linux Enterprise Server). Binary packages are available for these distributions. C++ and Ruby are the two languages used in OpenNebula, most components use only C++ or Ruby exclusively, so you only need to know one of them.

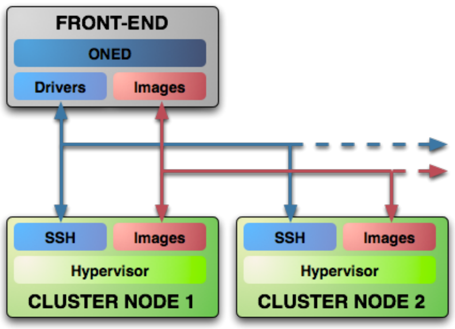

Figure 6: Topology of the architecture of OpenNebula.[28]

Guest operating system There are no template (virtual) images available which could be used after the installation and configuration of OpenNebula. These images need to be created afterwards. So there is no OS which is supported by default.

Hypervisor The supported virtualization techniques are: Xen, KVM and VMware.

Features

OpenNebula has a highly flexible architecture and interface which makes it possible to interact with and integrate other tools and products, like Eucalyptus, Amazon, ElasticHosts and Nimbus [27]. It has support for OGF's[10] Open Cloud Compute Interface specification [26].

OpenNebula can offload to remote public cloud providers like commercial clouds, like Amazon EC2 or ElacticHosts for which connectors are available. To use this, EC2 must be installed on the Front-End. To offload an instance to EC2 site, the image has to be uploaded on S3. It is also able to offload to partners that run their own OpenNebula private cloud instance.

When offloading using the EC2 cloud with OpenNebula, there are a few limitations[26]:

- Since there is no direct access to the dom0, this can not be monitored (you do not know where the VM is running on the EC2 cloud).
- You can lauch a maximum of 20 instances, because the EC2 API in OpenNebula is still in beta (although you can request more at Amazon).
- Snapshotting, restoring and migration are not available.

[10] "The Open Grid Forum (OGF) is the community of users, developers, and vendors leading the global standardization effort for grid computing" [7].

OpenNebula services are hosted in virtual machines, and are submitted, monitored en controlled in the Cloud. The control can be done by several different virtual infrastructure interfaces:

- Command line interface
- XML-RPC API
- Libvirt virtualization API or any of its management tools

The support for the EC2 Query API makes it possible to manage the cloud using any EC2 Query Tool of utility to access the private cloud.

It is possible to define various EC2 sites to allow OpenNebula to manage EC2 availability zones or to use other private clouds offering an EC2 interface. OpenNebula provides dynamic "cloudbursting" to any cloud with Amazon EC2 interfaces, including Eucalyptus-based clouds.

API OpenNebula is compatible with the Amazon EC2 interface and is designed to support additional client-side interfaces. It has an implementation of a subset of EC2 Query API and OGC OCCI API for the Cloud user interfaces. OpenNebula is also a flexible cloud service since it allows implementations of new cloud interfaces or develop new connectors.

Availability zones It is possible to manage multiple Amazon's EC2 availability zones with OpenNebula. "From OpenNebula 1.4 onwards it is possible to define various EC2 sites to allow opennebula the managing of EC2 availability zones or even the use of various private clouds offering EC2 interfaces"[26]

Fault tolerance / fail-over Fault tolerance is managed by a persistent back-end database, that stores the hosts and their VM information[26]. All virtual machine (VM) images are stored at the front-end (the Core) of OpenNebula. As soon as a machine is started, it is copied (cloned) to a cluster node which will then run the VM[28]. When the image is closed, its copied back to the repository.

(Live) migration OpenNebula can support live migrations, although it has to be implemented with *Shared FS*. With Shared FS, OpenNebula supports the full advantages of the hypervisor capabilities and the OpenNebula Storage module (i.e. no need to always shutdown virtual machines when moving an image. "OpenNebula can work without a Shared FS. This will make you to always clone the images and you will only be able to do cold migrations. However this non-shared configuration does not impose any significant storage requirements" [26].

Monitoring The core is a centralised component which managers use to monitor and manage the virtual machines and physical servers [29]

Open Virtualization Format (OVF) OpenNebula recommends the usage of this standard as it simplifies the coöperation with other virtualisation platforms.

Scaling "Dynamic resizing of the physical infrastructure by adding new hosts, and dynamic cluster partitioning to meet capacity requirements of services" [26]. It is not fully automated but the cloud could expand or shrink easily.

User management OpenNebula does not come with a graphical user interface. Instead of this a command line interface (CLI) is available. "Multiple user support and access-right control for Virtual Machines and Virtual Networks"[26].

Limitations

OpenNebula has the following limitations:

Graphical User Interface OpenNebula does not come with a graphical user interface by default, although the EC2 Query API is supported.

Max. front-end The front-end controller of OpenNebula can not be made redundant. It is possible to modify the source code to support this but on itself this is not possible. When this server will fail all nodes wil keep on running but can not be managed till the frond-end is up again. OpenNebula does have a good recovery procedure so when the front-end is up again everything is back to normal.

Max. EC2 instances OpenNebula is also able to offload to public EC2 clouds. When SURFnet wants to use this functionality, they need to take a few drawbacks in mind. Since EC2 is still a beta in 1.4, it is only possible to launch at most twenty simultaneous instances. Further are the functions for snapshotting, restoring, or migration not available with EC2.

MarketShare

There is a growing community behind OpenNebula. It also has a lot of sponsors which support there project, including: The Reservoir Project and The NUBA Project [26]. As there is a lot of interest on the web about OpenNebula. We expect its development will go rapidly. It is still a new product so it is mostly used within demo environments. With its great features and continues development we expect this will grow to an interesting and promising cloud solution.

Future developments

OpenNebula is currently working on supporting VirtualBox. They will implement this feature at version 1.4.2. Other developments are: SUN Cloud API, vCloud API, Grid service manager, LVM and SAN support. Other improvements are a security framework and the possibility to create SLAs [27].

OpenNebula is also planning on implementing more open standards like CSA, DMTF, OMG, and OASIS (see section 3.3 for explanations)[32]

Proof of Concept

We used CentOS 5.3 as the platform to build the OpenNebula cloud. After we installed all the dependencies of OpenNebula, we installed the OpenNebula 1.4 packages on a CentOS 5.3 platform. The following configuration is done in order to built the cloud environment. We used this test environment to research how OpenNebula can be implemented optimal for the requirements of SURFnet. In this section, the test environment is explained after this in section "Deployment", the implementation to meet the needs of SURFnet is explained. The descriptions in this section are based on information found on the OpenNebula website[26].

Installing OpenNebula 1.4 was a quite big task, because we first had to find out, which platform to run it as several platforms are supported. On the official OpenNebula website, we found all the dependencies of the different platforms. We decided to go with CentOS 5.3 as these packages could easily be used and found.

Nodes

The nodes were configured with KVM as a virtualization platform, we used this because it was easy to set up and use. The nodes only had one interface for both public and private connections. The dependencies for the nodes we needed were the ruby and ssh packages. In order to communicate with each other, the Front-End and all the Node Controllers need to have the same user and group settings.

Security

The OpenNebula Front-End communicates with the Node Controllers using SSH, so this package needs to be installed. The following thing is to create a keypair on the Front-End and copy the public key to the Node controllers *authorized_keys*, to communicate via SSH without entering a password everytime. OpenNebula uses the *oneadmin* user for managing everything, so his keypair needs to be known by all the nodes.

Storage

In order to use the full hypervisor capacity (features like live migration) and OpenNebula Storage Module (no need to clone images), the image repository will be exported using a *Shared FS*. Because we only had one node, we could not test the live migration feature. The image repository was placed on the same machine as the Front-End.

The following hierarchy on the Front-End file system is recommended by OpenNebula:

/srv/cloud/one	OpenNebula installation & clones for the running VMs
/srv/cloud/images	Master images & repository

Table 3: OpenNebula folder configuration.

/srv/cloud needs to be exported to all the Node Controllers with read/write permissions. The Node Controller mount this Front-End directory to **/srv/-cloud**

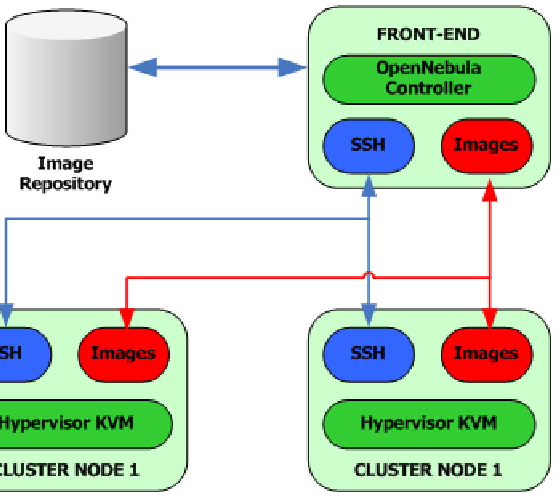

Figure 7: Structure of OpenNebula (based on information found on[26]).

Manageable OpenNebula is only manageable from the command line with a default installation in contrast to for example Eucalyptus, which has a nice graphical user interface. OpenNebula has an Open Cloud API (OCI) which makes it possible to manage the cloud with an EC2 Query user interface, but we did not implement it, because the command line had enough options to perform our tests.

Virtualization features like, live migration and fault tolerance As we only had one Node Controller we were unable to test these features also due to the fact we spend a lot of time implementing OpenNebula.

Offloading capabilities to another cloud Due the limited time we could not test this. Because the Eucalyptus cloud was not running properly, we could not offload the instances from OpenNebula to the Eucalyptus cloud. We also did not had the possibility to use the Amazon EC2 cloud to migrate instances to the public cloud, although we could find documentation about how to realize this.

Support for multiple guest operating systems We installed a CentOS 5.3 image to run on OpenNebula. Unlike Eucalyptus, we had to built our own images where Eucalyptus supports some default ready to use images. We only created one image and ran this in the cloud.

Result

Although we did not set up a complete OpenNebula cloud as test environment, we could draw a good conclusion about the possibilities and extensibility of OpenNebula. OpenNebula is specially designed to act as a cloud platform and is highly modular. For this reason it is quite a big job to implement it, but gives you on the other hand great possibilities to customize it. For example, there is a very limited schedular implemented by default. This schedular can easily be replaced by for example Hiazea. This schedular tool is specialized in scheduling the resources of OpenNebula.

10 AbiCloud

AbiCloud is a private cloud solution developed by Abiquo which has build up some experience in grid computing over the years [10]. With AbiCloud you are able to manage your (virtual) datacenters through an easy to use graphical user interface.

AbiCloud is able to monitor and manage physical and virtual machines. Users can create applications within the cloud to use these as a Software as a Service product. This product is also very flexible and easy to extend: "An Open platform that allows the easy integration of third party applications (monitoring, security, billing, etc.) in order to facilitate the creation of specific private clouds"[11].

AbiCloud has both a community version as well as an enterprise version. The main difference between these two is the support and with the enterprise edition it is possible to manage multiple datacenters. Also a special version for Service Providers is released called xSP, this has some key features to deploy a public cloud[12].

AbiCloud has a CPAL 1.0 license for the user interface part of the clients (Adobe Flex). For the server components a MPL 1.1 is used for the services running on this.

The version we used to research was AbiCloud version 1.0.0RC3 as this was the latest version, at the moment of this project.

Product components

AbiCloud consists of three components[11]:

- AbiCloud server, responsible for API virtualization for flex clients and database management. Users communicate through this front-end and able to manage Users, the physical server infrastructure, appliance repository, and virtual appliances.
- AbiCloud WS (Web Services), responsible for virtual applications management
- Virtual System Monitor WS (VMS), is the event un/subscriber manager and the plug-in monitor manager.

Supported platforms

Host operating system AbiCloud is written in Java, which means it is platform independent. The supported operating systems by Abiquo are: Linux Ubuntu, Linux CentOS, Microsoft Windows XP, and Mac OS X. Only Ubuntu and CentOS are supported for the node controllers (which run the virtualization techniques).

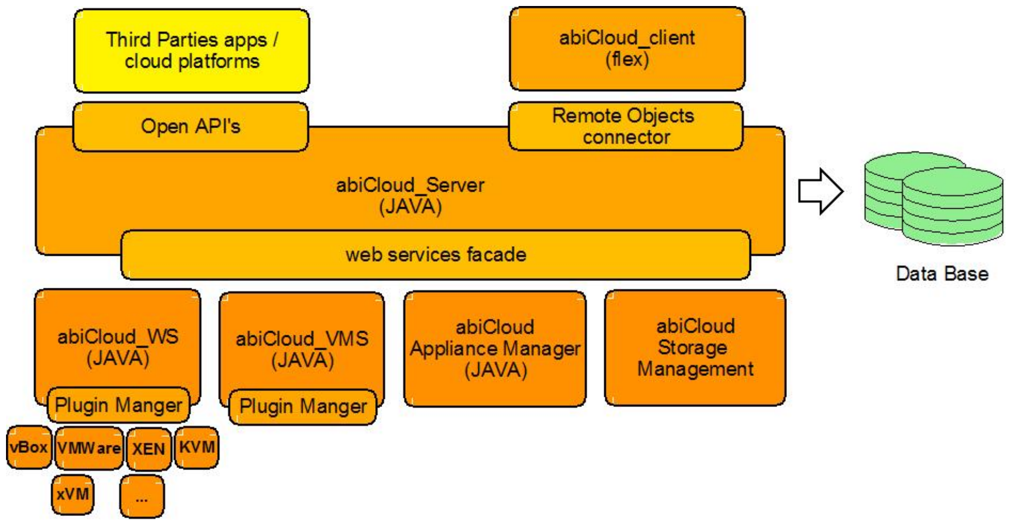

Figure 8: Platform architecture of AbiCloud.[11]

Guest operating system AbiClouds supports most popular operating systems, these include: OpenSUSE, Ubuntu, Fedora and CentOS.

Hypervisor The supported virtualization techniques are: VirtualBox 2.2, VMWare, XEN, and KVM.

Features

AbiCloud implements the Sun Open Cloud API. It is also possible to develop applications to self-manage cloud resources. When using the enterprise edition it is possible to link multiple clouds together. "It will be able to facilitate the integration between clouds (federation)" [11].

API AbiCloud has an open API which enables the integration of other third party applications. It makes use of Sun's Open Cloud API.

Availability zones AbiCloud supports the ability for users to create their own virtual datacenter within a physical datacenter, in this virtual datacenter an isolated cloud infrastructure is used to deploy cloud applications[11].

Monitoring AbiCloud has some build in functions to view the statistics of virtual and physical servers. Also because AbiCloud is flexible it is possible to use other monitoring tools.

Open Virtualization Format (OVF) Virtual images can be shared between datacenters with the Open Virtualization Format. With the use of this standard virtual appliances can be shared between different hypervisors.

"AbiCloud is designed to work with the OVF standard (nowadays with OVF 1.0 specifications)."[14].

Scaling Users have the capacity for scaling, management, automatic and immediate provision of servers, storage, networks, virtual network devices as well as applications [11]. AbiCloud allows to automatically modify soft-limits to scale the infrastructure."It provides mechanisms for provisioning of virtual machines in the pick load on the clusters" [11]. Unfortunately it does not support power management for physical servers.

User management AbiCloud has a very user-friendly graphical user interface build with Adobe Flex. It is possible to manage the entire cloud with this interface. Third party applications could also be used to manage AbiCloud. It is possible to manage users, which can manage processes within the cloud.

Limitations

AbiCloud has the following limitations:

Max. datacenters The community version works only with one datacenter[11].

Information Documentation is very limited. Information about AbiCloud can only be found on their own website, this makes it hard for us to make an objective decision. Although it is an open source project, the community is very small. There is only little documented experience with this product which makes it hard to determine what the real value is of AbiCloud. Some parts of the documentation and information found is in Spanish (Abiquo is a Spanish company).

Live migration There is no support for live migration.

Fault tolerance / fail-over AbiCloud itself does not support this feature, no documentation can be found about this.

MarketShare

AbiCloud is mainly active in Spain (Some of their support and documentation is in Spanish and translated to English). Their main focus is on hosting providers as well as enterprises which want to implement cloud solutions. We suspect AbiCloud is not really active outside Spain as we could only find two Spanish customers on their website, further more not much information can be found about AbiCloud, so it is a small player in the cloud computing market, but one with interesting potential.

Future developments

AbiCloud is still fully being developed. Some of the features they want to implement in future versions as stated on their website ([11]), are:

- Statistics and monitoring: An interface permitting the integration of any kind of monitoring information generated by the cloud with any other monitoring application or the creation your own scale scripts.
- Auto-scale: define when and how to scale your application in order to prevent problems in the system and not depend on third party tools to increase the infrastructure.
- Networking: agglutinate network functionalities to manage the virtual datacenter.
- Security: Extra security modules and functionalities.
- Billing information: Interface that allows recovery of all information generated by the cloud in order to monitor user activity.
- Notification system: To create alerts in the system to inform users.

Proof of Concept

To test AbiCloud we installed this on virtual machines on our own laptop using Sun's Virtual box. We used a preconfigured installation of AbiCloud to test drive the system ([15]). This installation is especially designed for this. We only tested AbiCloud for a short time as we noticed OpenNebula and Eucalyptus were much more promising. This is also why we only tested AbiCloud on virtual machines so we could continue testing OpenNebula and Eucalyptus on the physical machines.

AbiCloud consists of a Front-end and several machines attached to it which are managed. The default installation was using a modified version of OpenSuse. As soon as AbiCloud was running we noticed it was more a program for managing large datacenters. Abiquo is also known for its datacenter products. But we could quickly see this would not meet the requirements of SURFnet. Managing servers and creating applications to deploy is nicely supported by Abicloud. The creation of a private cloud is possible but offloading to others did not seem possible. We also had problems finding good information and manuals for AbiCloud. Only on their own website and (very limited) forum, we could find some information. Parts of this website were also not working or non existent.

Manageable AbiCloud is very easy to manage, the graphical user interface is very useful.

Virtualization features like, live migration and fault tolerance In the current version of AbiCloud we could not find features for live migration or fault tolerance of any kind.

Offloading capabilities to another cloud We could not test this due to the limited time, but for what we have seen multiple datacenters are supported which offers the ability to create multiple clouds.

Support for multiple guest operating systems AbiCloud has wide support for multiple operating systems. But this is obvious as it is a datacenter management program by heart.

Result

From our experience we could say AbiCloud could be useful to manage one or more datacenters but does not have the potential to meet SURFnet's requirements.

11 OpenQRM

"openQRM is a very comprehensive and flexible Open Source Infrastructure Management Solution. It is a fully pluggable architecture focuses on automatic, rapid- and appliance-based deployment, monitoring, high-availability, cloud computing and especially on supporting and conforming multiple virtualization technologies"[37].

openQRM has no features of itself. Instead it provides all kind of features by installing plugins, which turns openQRM in a mighty tool for infrastructure-management. These plug-ins feature virtualization platforms, storage, monitor, and control/deploying a computing cloud. A full list of al plug-ins for openQRM v4.x can be found at their website[37].

OpenQRM has two versions an enterprise edition and open source. "openQRM Enterprise consolidates the technical competence of the core members and developers of the openQRM Project, experienced with every detail of this Open Source solution, to supply expert knowledge for custom, sustainable Data Center setups in best-practice. Our focus is to lower the Total Cost of Ownership (TCO) for IT departments using a proven Open Source framework"[38].

"openQRM is published under GPL. This means, it's free software and you are not only allowed to use and distribute it, but even encouraged!"[38]

The version we used to research was openQRM version 4.6 as this was the latest version, at the moment of this project.

Product components

For an advanced High Availability openQRM 4.6 setup, you need the following machines:

- Two openQRM-server systems
- High Availability remote database server (like MySQL, Oracle or DB2)
- One or more High Availability Storage-servers
- Physical systems that are managed by openQRM to run the systems.

Supported Platforms

Host Operating System Host systems are widely supported by openQRM. It is written is the languages; C, Java, JavaScript, PHP, Perl, Unix Shell. The binary packages are available for the different Linux the platforms. "openQRM 4.x comes with a solid support for different linux distribution like Debian, Ubuntu, CentOS and openSuse. A single openQRM server can manage the provisioning of servers from those different linux distributions seamlessly."[37]

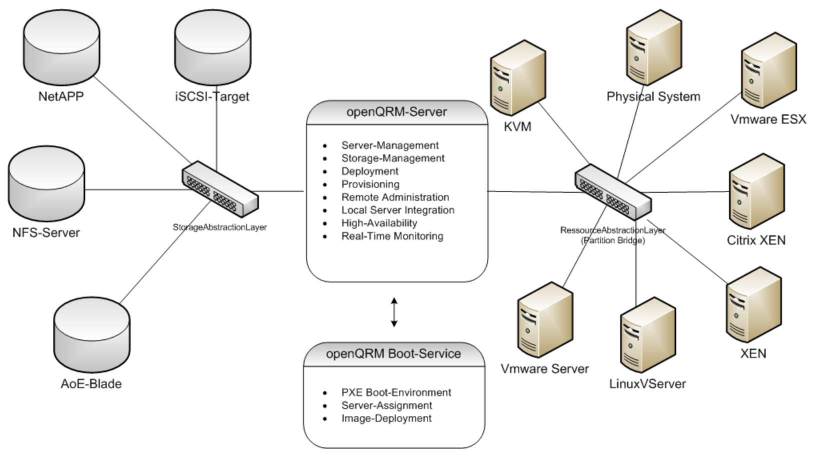

Figure 9: Topology of the architecture of OpenQRM.[37]

Guest Operating System "Ready-made-server-images via the image-shelf plugin. To get started quick and easy openQRM now provides ready-made and known-to-work server-images for Debian, Ubuntu, CentOS and openSuse. Therefore we added an image-shelf plugin which allows the system administrator to fetch servers easily via the web-interface. Public or custom image-shelf servers can be used, meaning you can either fetch server-images from our public image-shelf server or provide your own image-shelf server with custom images. openQRM also supports Windows virtualization."[37]

Hypervisor The supported virtualization techniques are: VMware, Xen, KVM and Linux VServer. These can be managed transparently via openQRM. openQRM also supports physical hosts, which is of course not a virtualization platform, but is a handy feature when migrating from a physical to virtualized datacenter.

Features

openQRM supports the Amazon EC2 API adapter. This makes it possible to build a hybrid cloud for offloading to Amazon EC2 via the Amazon EC2 connector. This makes it also possible to offload to other openQRM instances and offloading to Eucalyptus clouds and Ubuntu Enterprice Cloud.

API As said before, OpenQRM supperts the Amazon EC2 API.

Fault tolerance / fail-over openQRM supports "N to 1" fail-over. This means that you only need to have one stand-by server for N HA-servers. This saves N − 1 idle servers. You only need one (or a few) stand-by servers instead of one (HA-server) on one (backup server) to implement high availability in the cloud environment instead of a back-up server for every HA server.

"You can just save ALL your stand-by servers and just bring up a virtual-machine as stand-by. In case of problems HA-appliances will then fail-over from physical to virtual. You can also fail-over from virtualization technology A to technology B (e.g. from KVM to VMware vm's)"[37]. In order to accomplish high availability for physical servers, the cloud configuration consists of two systems to run a redundant openQRM-server.

(Live) migration openQRM supports various migration scenarios. It is also possible to migrate a server applicance from one virtualization technology to another virtualization technology. This makes the migration support for this platform a very interesting feature.

- Physical to Virtual
- Virtual to Physical
- Virtual to Virtual

Monitoring Fully automatic Nagios configuration (single click) to monitor all systems and services. From openQRM 4.1, Nagios is developed to support completely automatic configuration, which maps the entire openQRM-network and creates the Nagios configuration for the whole system.

Scaling openQRM supports auto-scaling through a powermanagement plug-in it is able to manage physical servers.

User management openQRM has a user-friendly web interface. It is called the "Datacenter Dashboard" and was redesigned in openQRM 4.6 to provide a clean and complete overview about the status and performance of all managed subsystems.

Limitations

OpenQRM has the following limitations:

Availability zones The documentation says nothing about availability zones, this is not supported.

Open Virtualization Format (OVF) OpenQRM has no implementation for OVF. Its virtual images have their own format.

Documentation Limited documentation is available. OpenQRM mainly focuses on their general product, with the cloud computing part only being a plug-in not much documentation and experience can be found on that.

MarketShare

OpenQRM is more of a management platform for datacenters, it is also mostly used for this. Just recently the cloud feature is introduced. OpenQRM already has a lot of sponsors which make it possible to continue the development as it is no longer under support of the company Qlusters, which went bankrupt. The community for openQRM is still growing as new improvements are developed, although this is mainly done for management of clusters as this is their main focus.

Future developments

As OpenQRM is further developed new features can be requested at the SourceForge page for openQRM. To name a few: support for VirtualBox, easy Windows support, and Network Configuration for Appliances.

Proof of Concept

For openQRM we had set up the latest version openQRM 4.6 and installed the packages on CentOS 5.3. Again we installed a Front-End and a Node using the tutorial from the openQRM website `https://www.openqrm.com/`. We used the laptops to set up this test environment.

We already noticed during the literature research that this platform is mainly used as a management tool. The platform does have some cloud plug-ins that can be installed, but it is not really a cloud platform. All the cloud techniques can be installed as plug-ins.

Manageable openQRM is very easy to manage, the graphical user interface is very useful.

Virtualization features like, live migration and fault tolerance A nice feature openQRM has is migrating physical machines to virtual machines and the other way around.

Offloading capabilities to another cloud We could not test this due to the limited time and we had no other cloud platform fully running.

Support for multiple guest operating systems openQRM has wide support for multiple operating systems, which comes from the fact that it is really a management tool.

Result

openQRM is a nice management platform, but does not meet the requirements of SURFnet for building Hybrid Cloud solution. For this reason, we did not research this platform any further.

12 VMware vSphere

Unlike the other described products VMware vSphere is installed directly on the hardware without an underlying operating system. With this it is possible to use most of VMware's virtualization techniques to build a private cloud within a company. vSphere consists of several products, to name a few: ESX/ESXi Hypervisor, Distributed Resource Scheduler (DRS), VMotion, VMware LifeCycle Manager. These are all widely used tools within companies to virtualize their datacenters[39]. Already many companies have implemented this technique from VMware including SURFnet and their relations. This is why it became interesting to see if vSphere solutions are useful to set up (a) private cloud(s).

VMware has been working on virtualization techniques for over ten years. They already have a lot of experience and powerful solutions for virtualization. Since cloud computing mostly consists of the use of virtualization there is a good chance vSphere is the best available solution. vSphere offers the most wanted techniques for datacenters like, High availability, live migration, fault tolerance, backup solutions and are easy to use. For the cloud infrastructure it offers vCloud, which is a product which has the ability to federate multiple clouds. With vCloud express it is possible to also offload workload to a public cloud. The only downside to this is that only a few public cloud providers support this [39].

There are four different editions of VMware VSphere, namely standard, advanced, enterprise, and enterprise plus. We mainly looked into the enterprise edition of VSphere as it supports all required abilities for SURFnet [40].

The version we used to research was vSphere version 4 update 1 as this was the latest version, at the moment of this project.

Product components

An VMware vSphere setup consist of the following components.

- A vCenter server is responsible for managing the datacenter. This provides many essential datacenter services such as access control, performance monitoring, and configuration.
- The datacenter consists of several nodes which run ESX(i). These nodes run all the virtual machines which are controlled by the vCenter suite.

Supported Platforms

Host Operating System VMware vSphere has its own hypervisor, namely ESX and ESXi.

Guest Operating System Most of the popular and recent operating systems are supported[11].

[11] For a full list check this website [42]

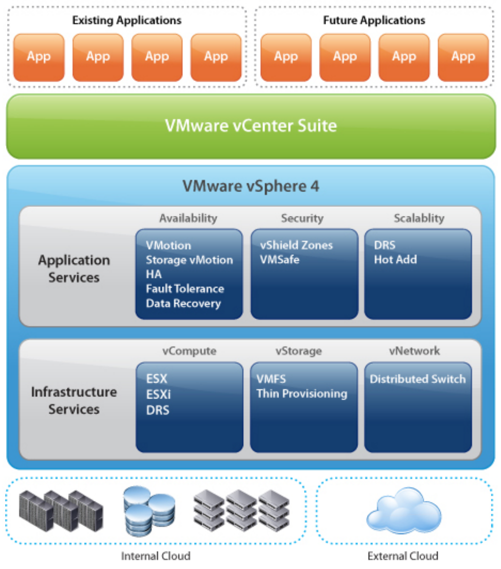

Figure 10: Overview of vSphere[39].

Features

vSphere supports the coöperation with multiple clouds with the use of vCloud. vCloud needs to run in every datacenter in the cloud. Offloading to public clouds is very limited, only a few cloud providers offer offloading from VMware vSphere clouds. vSphere can offload to other vSphere instances althrough HA is not available anymore.

A list of required functions and features of vSphere are listed below [12]:

API vSphere has a rich management console. With the enterprise plus version it is possible to extend this with 3rd party API's.

Availability zones Both the enterprise as the enterprise plus edition allow the ability to create isolated resource pools.

Fault tolerance / fail-over vCenter server controls this feature within the cloud.

(Live) migration vMotion enables the ability to move virtual machines from one physical machine to another even when the virtual machine is still running.

Monitoring With their monitoring capabilities you are able to view the computing load of a physical or virtual server.

[12]See this interesting video for quick introduction of the capabilities of vSphere [41]

Open Virtualization Format (OVF) VMware supports this open standard and implemented this in its vSphere technology.

Scaling With the use of Distributed Resource Scheduler (DRS) the server load will automaticly be devided over all resources.

User management vSphere has management software which could be installed on several client machines to manage the VSphere servers. From a single client, connections could be made to several datacenters to manage all virtual host servers.

Limitations

VMware vSphere has the following limitations:

Host limitations With all the features vSphere has, there are a few limitations. The maximum limit of hosts per High availability cluster or DRS cluster is 32[43]. In large datacenters this could be a problem which should be kept in consideration.

Multiple clouds Only a hand full of providers offer the ability to use vCloud and thus offload workload to public clouds. So this is very limited.

MarketShare

VMware is one of the largest providers in virtualisation solutions. A big advantage of this is that all their products are well tested and documented. Also it will be easy to find administrators for their solutions. We expect vSphere will quickly grow to one of the industry leading cloud computing solutions as it already offers many wanted features.

Future developments

vSphere keeps getting developed, as we speak support for more guest operating systems and hardware is added. As vSphere becomes more popular the more it will be supported by public cloud providers. We expect the abilities of vCloud to grow and more and more public cloud providers will support it.

Proof of Concept

We did not set up a test environment for vSphere because of the limited time and we were able to see a demonstration given by one of the employees of SURFnet. SURFnet is currently using vSphere for its virtualization capabilities. This is why it was for us interesting to check its cloud possibilities.

During this demonstration we got to see how easy it is to manage multiple sites from one location. Also features like live migration and fault tolerance were very easy to configure. The monitoring capabilities were also very extensive and everything could be monitored easily. All the features shown to us are more virtualization features, and as we did not want to spend much time in setting up a vSphere environment we used information provided to us to determine the use for vSphere. From our literature study we found that vSphere was great for the virtualization of servers but not yet ready to be used as a cloud like SURFnet wishes it. As could be read before there is to little support for vSphere/vCloud yet.

To get back to the points we wanted to test we can answer the following:

Manageable Very user friendly interface. Multiple datacenters could be easily managed from any location.

Virtualization features like, live migration and fault tolerance It has all the modern virtualzation techniques useful for managing a datacenter like High Availability, Live migration and Fault tolerance. VMware is one of the top players on this at the moment.

Offloading capabilities to another cloud It is possible to move virtual servers although not live migration to other datacenters. But only other vSphere datacenters are supported. This does not make it too flexible.

Support for multiple guest operating systems The list of supported operating systems is very big. Almost all Windows and Linux variants are supported.

Result

VMware vSphere has great virtualization techniques for managing a datacenter/private cloud although due to the fact that there is almost no support for public clouds or other private cloud solutions we advice another solution for SURFnet.

13 Comparison

In this chapter we discus all previous described cloud platforms and compare them. All results are displayed in a matrix (table 4) which gives a clear outcome of the possibilities of the different platforms. With these results we will base our conclusion on which platform best meets SURFnets requirements. This conclusion can be read in the next section (17).

13.1 Compare matrix

In this matrix (Table 4), all the properties of the researched cloud platforms are summerized. The deployment models are explained in the first part of the matrix. These are the primary requirements of SURFnet and so to our research. After this part, the availability of the platforms is explained. These properties will be an important factor to make a trade-off between the different platforms in terms of availability requirements. The next part is about the features the platforms offer, followed by their management tools.

The final part is about the deployment and standards of the platforms, followed by our opinion of the product potential. We graded the product potential of the products by found information. For every platform we did research on the market share, which consists of sponsors, customers, running demo environments and the community behind it. Further we looked at the future development, which consists of the potential of the product and the current and expected features. For this compare matrix, we graded the product potential with a - - (very bad), -, 0, + or ++ (very good).

This matrix is based on all found information which is described in the section: 8 (Eucalyptus), 9 (OpenNebula), 10 (AbiCloud), 11 (openQRM) and 12 (VMware vSphere (vCloud)). For a full explanation about the terms used in the matrix see section 5.2

	abiCloud 1.0.0-RC3	**Eucalyptus 1.6.1**	**OpenNebula 1.4**	**openQRM 4.6**	**VMware vSphere 4**
Deployment Model					
Private cloud	yes	yes	yes	yes	yes
Offload to private (same platform)[1]	yes	no	yes	yes	yes
Offload to private (other platforms)	no	no	Eucalyptus & UEC	Eucalyptus& UEC	no
Offload to public[2]	no	yes	yes	yes	limited providers
API[6]	Sun Open Cloud	EC2	EC2 & OGC OCCI	EC2	vCloud
Availability					
Redundant Front-End	yes	no[3]	no	yes	yes
Fault tolerance/fail-over[4]	no	no	yes	yes	yes
Features					
Availability Zones	yes	yes	yes	no	yes
(Auto) resource scaling	Manual	Manual	Dynamic[5]	yes	Automatic[10]
Live migration	no	no	only with Shared FS (SAN/-NAS)	no	only with Shared FS (SAN/-NAS)
Management					
User Management	web interface	web interface	cli 3rd party GUIs	web interface	yes[7]
Monitoring	Build-in	support for 3rd party	support for 3rd party	Nagios included	internal(and support for 3rd party)
Licenses and Standards					
open source	yes	yes	yes	yes	no
Enterprise	yes	yes	no	yes	yes (only)
Open Virtualization Format (OVF)	yes	no	recommended[8]	no	yes
Product Potential					
Marketshare	-	+	+	0[9]	++
Future development	+	++	++	+	++

Table 4: Comparison of the 5 cloud platforms.

[1] Another private cloud created using the same platform
[2] Possible public IaaS cloud prividers like Amazon EC2
[3] Eucalyptus has several front-end components (cloud- and cluster controller) which could be run on different machines. Failure of the cluster controller has a bigger impact then of the cloud controller (you loose all you instances).
[4] Running redundant VMs or servers in case one fails.
[5] Dynamic scaling of private infrastructure to meet fluctuating demands through their own scheduler, this could be replaced by third party products.
[6] Many platforms support the EC2 API, it is possible to use this API to interact between the different APIs this way.
[7] vSphere has a GUI and has the ability to manage users through Microsofts Active Directory
[8] OpenNebula recommend the usage of OVF.
[9] OpenQRM is mostly used for management of clusters, this makes it hard for us to determine how active it is in cloud.
[10] VMware uses Distributed Resource Scheduling (DRS) to scale their network.

13.2 Overall outcome

The main outcome from this “compare matrix” is that there is not a platform available, which is able to offload to both a public and private cloud together with a high availability (redundant Front-End) set-up. When these features are required, no platform will meet this. Our expectation is that OpenNebula or Eucalyptus will be the first platforms to support the combination of these features, because of the active community behind them.

13.3 Scenarios

There are different scenarios for which a solution will fit best. These scenarios will depend on the ability to offload to other clouds and the availability of the platforms.

Private Cloud When you want to build a private cloud for internal use, we will recommend VMware vSphere. The biggest drawback of this product is that it is not possible to offload to public clouds. When this will not be used, VMware vSphere will offer more features than the other platforms, including availability and the richest set of features. Another important factor here is that VMware's market share is great and is a leading virtualization player over the years.

Private Cloud with offloading abilities VMware vSphere is not able (or very limited supported) to offload to public clouds. When this is a requirement OpenNebula 1.4 a good option. Offloading is well supported for this platform. A drawback of this cloud platform is that it has no redundant Front-End, which make this platform inappropriate for high availability cloud platforms. A feature OpenNebula does support is live migration of the VMs. This feature together with better support to offload to other private clouds we advice OpenNebula over Eucalyptus.

14 Deployment

In this section we will describe several option, which could be used by SURFnet when implementing OpenNebula as a cloud platform. A large part of the implementation we already covered in section 9. In this section we cover extra possibilities which would be useful for SURFnet.

14.1 Networking

In order to avoid potentially vulnerability for compromised VMs, it is possible to set up the IEEE 802.1Q protocol to use VLANs. OpenNebula provides an infrastructure to dynamically create new LANs for the VMs. With this network isolation, you can put any network service in the virtual networks, like a DHCP server to supply every virtual network with its own IP range.

14.2 User Interface

On the OpenNebula Cloud API (OCI) layer, the Amazon EC2 Query web service is implemented. The EC2 Query Service is installed on the Front-End. With this interface, you can use any EC2 Query tool or utility to access the private cloud. This EC2 Query Service runs on a normal HTTP connection by default although if extra security is needed, SSL can be used to secure this. For SSL support, the following setup is needed:

- SSL certificate
- HTTP proxy for SSL
- EC2Query Service configured to accept the requests from the proxy

On this way, you can get a secure connection to your OpenNebula cloud using the SSL certificates. [26]

14.3 Scheduling

Hiazea can extend OpenNebula's scheduling capabilities. It is able to support advanced resource reservation and queuing of best effort requests. It facilitates leases of resources as VMs with a variety lease terms. The OpenNebula scheduler daemon can be easily switched off and replaced by Haizea. Haizea is also published under the Apache License 2.0. "OpenNebula is a virtual infrastructure manager that enables the dynamic deployment and re-allocation of virtual machines on a pool of physical resources. OpenNebula and Haizea complement each other, since OpenNebula provides all the enactment muscle (OpenNebula can manage Xen and KVM virtual machines on a cluster, with VMWare support) while Haizea provides all the scheduling brains" [36].

14.4 Multi EC2 Site Support

The Front-End will run an EC2 adapter in order to offload to the other clouds. The OCCI adapter will also be installed to offload to OpenNebula nodes. Be sure, the EC2 adapter is installed only on the Front-End.

The Hybrid Cloud Deployment will be fully transparent to infrastucture users. An added public cloud like Amazon or another instance of OpenNebula will appear as a normal host in the OpenNebula system.

To install offloading to public clouds, you need to have a working account for AWS and signed up for EC2 and S3 services.

The Cloud will be able to offload VMs to Amazon EC2 or ElasticHosts as public clouds and to offload to other private OpenNebula and Eucalyptus clouds.

The OpenNebula OCCI enables you to launch and manage virtual machines in the OpenNebula installation. This OpenNebula OCCI service is also implemented on the **OpenNebula Cloud API (OCI)**.

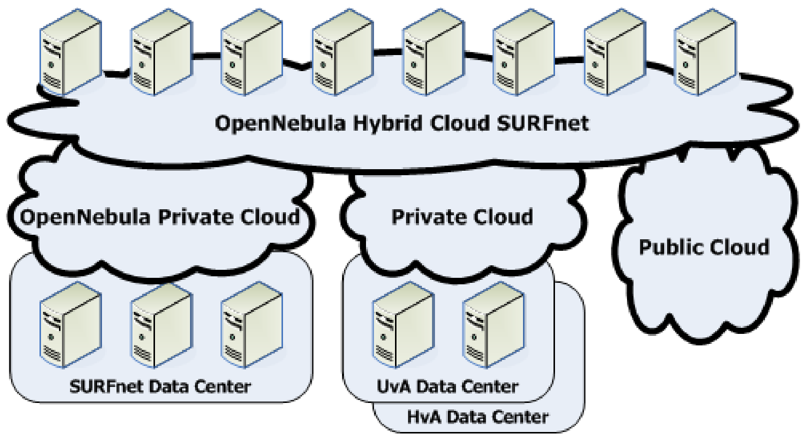

Figure 11: Cloud design SURFnet.

15 Use Cases

In this section, a few Use Cases are treated that can be used with the Cloud Computing solution. The Use Cases are divided in “Private Cloud”, “Hybrid Cloud”, “Community Cloud” and “Green IT”. Examples are given how cloud computing can optimize SURFnet datacenters, but also the datacenters of SURFnet’s relations in the case of the Hybrid and Community Clouds.

15.1 Green IT

With global warming and recent natural disasters people are starting to think more about the worlds environment we all live in. Many techniques are developed to be as energy efficient as possible as some of the natural resources of this planet are running out, like petrol and other biofuels[13]. Some examples are the hybrid car and energy harvest from wind or the sun. As computing has become a very large player in power consumption in this digital age, it can not stay behind.

Virtualization is one of these developments to save power (and thus money) by efficiently using the available resources. Another upcoming technique is cloud computing (what we have been talking about in this paper), it makes very good use of the just mentioned virtualization technique by sharing resources. Combining resources using computing equipment and power as efficient and effective as possible without or minimal impact to the environment which is called *Green computing* (Green IT)[45].

With clouds (both public and private) it is possible to save money by shutting down idle equipment. This would save power consumption and the overall live span of this equipment. When needed equipment could be turned on instantly. This way it is also possible to handle sudden peaks. Extra resources could be added within a few seconds to meet the demands.

By using cloud computing as described above you have implemented *green computing* and thus saving a lot of money on for example power consumption.

15.2 Private Cloud

All SURFnet’s relations have their own private datacenter, which now handle their resources only internally. These resources could be in the form of physical hardware, where services are running on. It could also be in a private cloud environment like VMware vSphere, which is a widely used datacenter platform, including SURFnet and many of their relations. Instead of using static physical machines, flexible and mobile virtual machines are used. (Private) Cloud Computing makes it possible to collaborate these resources within the company. All the resources can be used and reused very flexible. When there are enough

[13]The most important reason people are searching for alternative fuels is due to the fact that natural resources are becoming more and more expensive as they become rare. The world runs on money.

bundled resources, it is possible to use for example 50 percent for the production environment, 30 percent for the develop environment and 20 percent for growing. Adding and replacing extra servers can be done very flexible without changing the infrastructure. Just add the resources to the cloud and let the cloud handle the scaling and scheduling.

Optimize Resources A concrete example how companies can use a private cloud for their infrastructure is when they work with different departments, which all have their own hardware. If, lets say only 20 percent of the capacity for the mail servers are used, but the webservers are running on 80 percent of their capacity, a private cloud can easily extend the resources of the webservers, by using more resources from the cloud, because all the resources are collaborated. With physical machines, the 80 percent unused power from the mail servers, could have been a wast of resources (money). Now the purchase of extra hardware for the webservers is not needed.

Private cloud environments can change the behaviour of the investments of these companies. Instead of purchasing and depreciation of hardware on a department basis, this can now be done on a global level. This avoids overall over-investment and create a flexible pool of resources, which can be used company wide.

15.3 Hybrid Cloud

When all the players collaborate in the private cloud solution with offloading abilities, they can support each other when they need to handle peaks. Important to say is that with the current features of OpenNebula 1.4 and the fact that many of SURFnet's relations are currently running VMware datacenters, this Use Case can not be realized at this moment. In order to realize these situations, SURFnet and their relations have to migrate to OpenNebula 1.4. Another option is to wait for the release of OpenNebula which supports the vCloud API, this would make it possible to offload to VMware infrastructures. In this case, the OpenNebula implementation at SURFnet will be able to realize the offloading ability to the other datacenters. This offloading ability gives us great flexibility between different private cloud environments and also to public clouds.

"Unlimited Resources"

In a certain way a Hybrid Cloud solution offers "unlimited resources". Every moment extra resources can be added to the cloud. This can be resources from other private clouds, or resources from a public cloud.

Handle Resource Peaks A concrete example of these peaks are that websites of higher educations and universities have a big peak in the summer and during Christmas holidays (Figure 12) from people who are looking for another study. Of course these peaks could be handled internally, but if that is not possible and other institutes have some (temporary) overcapacity, they can help each other without doing unnecessary investments for temporary peaks. Of course there are a lot more examples of service peaks. Peaks like the previous example are predictable and easy to prepare for.

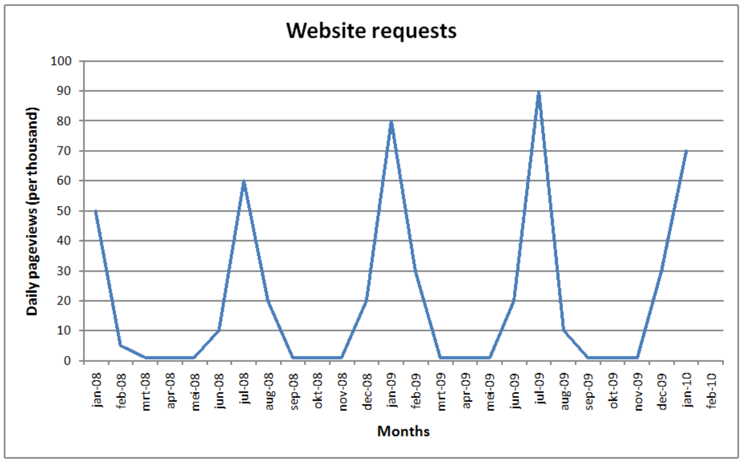

Figure 12: Example of request peaks (fictive data).

SURFnet Spam Filter Another example is the Spam Filter SURFnet is using. Sometimes there are peaks in the usage of these services. With Cloud Computing it is easy to participate on those peaks by adding extra resources (virtual machines) to the Spam Filter. The only thing that needs to be handled is that those virtual machines can coöperate to handle those peaks together. When the peak is over, virtual machines can be shutdown in order to reuse the resources for other purposes.

Private vs Public

We have been talking a lot about private en public clouds, but how could these be used by SURFnet. The advantages of private clouds have been described extensively in previous sections. Although their are some extra costs with this compared to a public cloud, we still advice a private cloud to SURFnet. With a private cloud SURFnet would have to manage their own datacenter which means: lots of power consumption, cooling, man power to manage everything, etc. This would be expensive. As with a public cloud you would just pay for the virtual machines you host within the public cloud. This would reduce the costs for servers/resources a lot compared to a private cloud.

A public cloud is a shared medium so other companies are also using the same resources although through availability zones customers are separated. One of the biggest disadvantages is the location of the public cloud.

As SURFnet already has a datacenter as well as their institutes we advice to only use a public cloud when resources within the private cloud can not meet the demand. SURFnet also had another research done to find out if it is possible to use a public cloud due to law and regulations.

Managing a Hybrid Cloud

In order to run a Hybrid Cloud solution together with other relations, some management decisions deed to be made in advance.

Offloading management Offloading services means that the services are moved outside the walls of SURFnet's datacenter. For security reasons it is important to keep the most critical and sensitive services within their own walls. In order to accomplish this the services (virtual machines) need to have a priority, so the system knows which services to offloaded first.

Borrowing or Billing These resources can be shared without charging for instances, if they all use them evenly. When this is not the case, overcapacity could be sold to other parties.

Setup security policies SURFnet has their own security policies. SURFnet and the other parties need to make arrangements of the security level the cloud provides, in order to take away compatibility problems

Cloud Management Tools like Haizea are able to schedule and reserve the available resources. Haizea is able to reserve resources for specific moments. It is important that all the resources for internal usage are reserved in time, so no other parties are offloading their virtual machines to resources of SURFnet, that want to use them internally. This way, parties do not get in each others way.

15.4 Community Cloud

Community Clouds can be created by SURFnet and interested relations by donating dedicated hardware to build a cloud together. The idea here is to donate some physical hardware with a group of companies, which are connected as a system wide cloud. Together they can build a shared environment with all the donated resources. This pool of resources can be used for all different kind of purposes. All the involved parties should install OpenNebula Node Controllers on their hardware. One of the parties will run the OpenNebula Front-End

Backup Environment

You can also use this cloud as a kind of insurance for disasters. Together parties can set-up a Community Cloud that can be used as a backup site. It is not very likely that more datacenters run into disasters simultaniously. This way they can create an private cloud environment that can be used by the party for disaster recovery. Depending on the size of the cloud the datacenter can migrate fully or partially to the community cloud, until the datacenter is back online. It is important to use this Community Cloud efficiënt, which means that this whole Cloud should not be idle until there is a disaster. This will destroy the efficiënt and *green* properties of Cloud Computing.

Managing a Community Cloud

Who is responsible? One of the parties should be responsible for the Community Cloud, because there is only one Front-End. Access to this Front-End should be arranged between the parties.

Cloud Management Because the Cloud is used for some global purpose, the management of the Cloud is centralized. The party that is responsible for the development that is within the cloud should manage this.

Security Because the Cloud is used for one purpose, all the security policies should be compatible between the different parties.

16 Future Research

For this research we had to make a lot of assumptions, because there was not enough time to really test it. Most of the information can only be found on the developers website and everyone claims to have abilities which are unique. Especially abiCloud has very limited information, so we see this as a big drawback. Maybe if there was more time to build a test environment our opinion about this platform (and others) will change. We mostly focussed us on Eucalyptus and OpenNebula, as these were the most interesting ones, most reviewed and explicitly designed for cloud computing.

Most important information we are missing now is testing results on how the different platforms offload to each other, the stability and more about the security. As developers hope to "sell" their product negative points are left out.

We hope that soon there will be more information, testing results and reviews about the platforms available from other (third party/objective) sources. Further research (like several test environments) on this topic can offer this kind of information.

As there are multiple tools to manage OpenNebula and other platforms there has to be looked into which is best for managing one or more clouds. When SURFnet would combine its cloud with other clouds to create a hybrid cloud some party has to be responsible to manage the connections between these clouds. This has to make sure clouds are not overloaded when multiple clouds want to offload. Also an important aspect is sensitive data. When using multiple clouds no sensitive data should be offloaded to another cloud.

In future research we suggest performance, security, coöperation, and stability are further researched as we were not able to include this in our research due to the limited time.

17 Conclusion

The result from our research is that out of the five researched platforms OpenNebula is best for SURFnet. This does not mean this is the best platform in any case as others offer different features which could be critical in different scenarios. Currently cloud computing is relatively new as everything is still in development. There are several solutions which are starting to mature but until now all platforms still have many limitations for the requirements of SURFnet.

To answer our research question; *Which Cloud Computing platform meets the requirements best, to combine multiple clouds to create a hybrid cloud for SURFnet?*. OpenNebula has all capabilities to share resources between different clouds plus some other useful features which makes it the best candidate. OpenNebula is able to create a private cloud and is able communicate with several other (private) cloud platforms to share data between clouds. Also the ability to offload workload to a public cloud is possible with the combination of OpenNebula and the Amazon EC2 API. These were the most important features which needed to be supported. Other platforms (like OpenQRM) were also able to support both these features, although in a limited way compared to OpenNebula. Besides the offloading abilities other interesting features were offered by OpenNebula, like live migration and fault-tolerance, but not by the others (for all features and abilities see section 13).

Although OpenNebula supports all features SURFnet requires, before they are able to use their cloud optimal, first one or more of their institutes need to implement a private cloud also. Since OpenNebula is compatible with Eucalyptus it is not necessary for SURFnets institutes to also implement OpenNebula. As OpenNebula is also developing support for the vCloud API in the near future it will be possible to combine it with vSphere.

Another important factor for a successful implementation of a Hybrid or Community Cloud is the management and scheduling of those resources, when those resources cross the borders of the company. There should be one global cloud administrator for administrating the shared part of the cloud between those parties. At least for the shared resources in order to plan the usage of the resources smoothly.

17.1 Advice

We were not able to test some of the features (like security and stability) as described in section 16 due to the limited time. This is why we suggest this is done in future research as this is an important aspect in enterprise solutions. Our advice is to build a demo model of a Hybrid Cloud together with interested parties to first of all get familiar with the platform. They still need to look into the security threats of offloading to other private and public clouds. Performance of offloaded services can be tested and the results of these tests can be taken into account. Another important part of the cloud is how to handle the Front-End that is now a single point of failure. This is a big disadvantage of the platform right now. The whole management for the shared part of the cloud needs to be arranged in advance between the different parties in order to use and reuse all resources optimal.

As SURFnet wants to implement our solutions, we suggest they first set up a demo environment to fully test all features and performance.

Bibliography

[1] **Website: Wikipedia - Utility computing**
As seen on Januari 24 2009
`http://en.wikipedia.org/wiki/Utility_computing`

[2] **Website: Cloud computing: Middleware & Virtualization**
As seen on Januari 24 2009
`http://natishalom.typepad.com/nati_shaloms_blog/2007/12/`
`middleware-virt.html`

[3] **Website: Cloud layers**
As seen on Januari 24 2009
`http://blog.gogrid.com/2009/03/26/navigating-the-layers-of-the-cloud-computing-pyramid/`

[4] **Website: Wikipedia - Communication as a service**
As seen on Januari 24 2009
`http://en.wikipedia.org/wiki/Communication_as_a_service`

[5] **Website: What cloud computing really means**
As seen on Januari 24 2009
`http://www.infoworld.com/d/cloud-computing/`
`what-cloud-computing-really-means-031`

[6] **Article: The NIST Definition of Cloud Computing**
Authors: Peter Mell and Tim Grance
Date: Juli 10 2009 `http://csrc.nist.gov/groups/SNS/`
`cloud-computing/cloud-def-v15.doc`

[7] **Website: Cloud standards**
As seen on Januari 29 2009
`http://cloud-standards.org/wiki/index.php?title=Main_Page`

[8] **Website: SURFnet**
As seen on Januari 27 2009
`http://www.surfnet.nl`

[9] **Website: Wikipedia - NREN**
As seen on Januari 27 2009

`http://en.wikipedia.org/wiki/National_research_and_education_network`

[10] **Website: Wikipedia - Abicloud**
As seen on Januari 22 2009
`http://en.wikipedia.org/wiki/AbiCloud`

[11] **Website: Abiquo - Community Abicloud**
As seen on Januari 22 2009
`http://community.abiquo.com/`

[12] **Website: Cloud hosting**
As seen on Januari 22 2009
`http://www.b10wh.com/tag/cloud-hosting/`

[13] **Website: Open Virtualization Format**
As seen on Januari 22 2009
`http://www.vmware.com/appliances/getting-started/learn/ovf.html`

[14] **Website: Abiquo - OVF**
As seen on Januari 25 2009
`http://blog.abiquo.com/category/uncategorized/`

[15] **Website: AbiCloud test servers**
As seen on Januari 25 2009
`http://www.abicloud.org/display/abiCloud/Abicloud+Server+Live+Installers`

[16] **Website: Eucalyptus Open source**
As seen on Januari 16 2009
`http://open.eucalyptus.com`

[17] **Article: The Eucalyptus Open-source Cloud-computing System**
Authors: Daniel Nurmi, Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, and Dmitrii Zagorodnov
`http://open.eucalyptus.com/documents/ccgrid2009.pdf`

[18] **Whitepaper: Eucalyptus Open-Source Cloud Computing Infrastructure - An Overview**
`http://eucalyptus.com/`

[19] **Website: Eucalyptus Enterprise Edition**
As seen on Januari 16 2009
`http://eucalyptus.com`

[20] **Website: Wikipedia - Amazon Web Services**
As seen on Januari 17 2009
`http://en.wikipedia.org/wiki/Amazon_Web_Services`

[21] **Website: Eucalyptus Press Release - November 5, 2009**
As seen on Januari 18 2009
`http://www.eucalyptus.com/news/11-05-2009`

[22] **Website: Ubuntu & Eucalyptus**
As seen on Januari 18 2009
`http://www.ubuntu.com/partners/Eucalyptus`

[23] **Website: Private Cloud to Public Cloud - Presto Eucalyptus**
As seen on Januari 19 2009
`http://www.cxotoday.com/India/News/Private_Cloud_to_Public_Cloud_-_Presto_Eucalyptus/551-101785-908.html`

[24] **Website: Azurejournal - weekly-cloud-application-eucalyptus**
As seen on Januari 19 2009
`http://www.azurejournal.com/2009/01/weekly-cloud-application-eucalyptus/`

[25] **Website: IBM - Cloud virtual infrastructure**
As seen on Januari 19 2009
`http://www.ibm.com/developerworks/opensource/library/os-cloud-virtual1/index.html`

[26] **Website: OpenNebula**
As seen on Januari 19 2009
`http://www.opennebula.org/doku.php?id=about`

[27] **Presentation: OpenNebula infrastructure**
As seen on Januari 19 2009
`http://www.linux-world.nl/nl-NL/Bezoeker/Activiteiten/~/media/linuxworld/Images/Algemeen/Ignacio%20M%20Llorente%20-%20The%20OpenNebula%20Open%20Source%20Toolkit%20to%20Build%20Cloud%20Infrastructures.ashx`

[28] **Website: OpenNebula - planning installation**
As seen on Januari 19 2009
`http://www.opennebula.org/doku.php?id=documentation:rel1.4:plan`

[29] **Website: OpenNebula - Architecture**
As seen on Januari 20 2009
`http://www.opennebula.org/doku.php?id=documentation:rel1.4:architecture`

[30] **Paper: Capacity Leasing in Cloud Systems using the OpenNebula Engine**
Authors: Borja Sotomayor , Ruben Santiago Montero, Ignacio Martin Llorente, and Ian Foster
`http://haizea.cs.uchicago.edu/pubs/Haizea_CCA08.pdf`

[31] **Website: Ubuntu - OpenNebula**
As seen on Januari 20 2009
`https://help.ubuntu.com/community/OpenNebula`

[32] **Presentation: OpenNebula - build cloud infrastructures**
As seen on Januari 28 2009
`http://www.opennebula.org/lib/exe/fetch.php?id=outreach&cache=cache&media=the_opennebula_standard-based_open-source_toolkit_to_build_cloud_infrastructures.pdf`

[33] **Website: OpenNebula - templates**
As seen on Januari 28 2009
`http://opennebula.org/doku.php?id=documentation:rel1.4:template`

[34] **Website: OpenNebula - cli**
As seen on Januari 28 2009
`http://opennebula.org/doku.php?id=documentation:rel1.4:cli`

[35] **Website: haizea**
As seen on Januari 28 2009
`http://haizea.cs.uchicago.edu/index.html`

[36] **Website: haizea - manual**
As seen on Januari 28 2009
`http://haizea.cs.uchicago.edu/doc_multiple.html`

[37] **Website: openQRM**
As seen on Januari 22 2009
`https://www.openqrm.com`

[38] **Website: openQRM - enterprise**
As seen on Januari 22 2009
`http://www.openqrm-enterprise.com/company.html`

[39] **Website: VMWare - vSphere**
As seen on Januari 23 2009
`http://www.vmware.com/products/vsphere/`

[40] **Whitepaper: vSphere features**
As seen on Januari 23 2009
`http://www.vmware.com/files/pdf/key_features_vsphere.pdf`

[41] **Video: vSphere features**
As seen on Januari 23 2009
`http://download3.vmware.com/demos/smb/vmw_smb_demo.html`

[42] **Whitepaper: vSphere - supported software**
As seen on Januari 23 2009
`http://www.vmware.com/resources/compatibility/search.php?deviceCategory=software`

[43] **Whitepaper: vSphere - limitations**
As seen on Januari 23 2009
`http://www.vmware.com/pdf/vsphere4/r40/vsp_40_config_max.pdf`

[44] **Website: Wikipedia - linux vServer**
As seen on Januari 28 2009
`http://en.wikipedia.org/wiki/Linux-VServer`

[45] **Website: Wikipedia - Green computing**
As seen on Januari 29 2009
`http://en.wikipedia.org/wiki/Green_computing`