2014 IEEE International Conference on High Performance Computing and Communications (HPCC), 2014 IEEE 6th International Symposium on Cyberspace Safety and Security (CSS) and 2014 IEEE 11th International Conference on Embedded Software and Systems (ICESS)

Service Deployment in Cloud

Amel Haji, Asma Ben Letaifa, Sami Tabbane
Higher School of Communication of Tunis, SUP'COM
Mediatron Laboratory
Tunis, Tunisia
{amel.haji|asma.benletaifa|sami.tabbane}@supcom.tn

***Abstract*— Among the many limitations that the clouds are facing today, the heterogeneity of IT resources and networks. The dynamic deployment and on-demand applications are also major challenges. To ensure this deployment, it is essential to rely on methods of managing applications and services in a cloud. This paper proposes to apply the concept of Virtual Appliance for cloud architectures to ensure the dynamic deployment, interoperability and reuse of services supported by these clouds. Proposed architecture aims to solve problems of users whose are blocked from sellers with very little flexibility to move their services to other providers. Using standards of DMTF could provide more flexibility and response to user's requirement under SLA constraints.**

Keywords—Cloud Computing; Cloud federation; OVF; Virtual Appliance; Service Deployment

I. INTRODUCTION

The Cloud Technology has changed how to deploy and use the services (pay-per-use, automatic scalability). Interoperability challenges arise: Amazon EC2, GoGrid, etc. are based on proprietary mechanisms of service definition. So users are vendor lock-in. Migration of service from a cloud to another is difficult. The most popular barriers in cloud: insure interoperability and portability between different Clouds, deploy Virtual Appliances in a cloud federation, role sharing between different actors (Cloud provider, cloud broker, Network Provider, user ...) along stages of service deployment. The main lock is deploying Virtual Appliance composed a graph of service in a cloud federation. The current works focus on provisioning Virtual Appliance in a single Cloud not within a federation.

The rest of this paper is organized as follows: First, a concise survey on the existing state-of-the-art in Cloud computing and its standards is presented. Next, the comprehensive description related to overall system architecture and its elements that forms the basis for constructing federated Cloud infrastructures is given. This is followed by discuss and proposed architecture.

II. CLOUD COMPUTING

A. Service Models

There are three big levels: SaaS (refers to providing on-demand applications over the internet), *PaaS* refers to providing platform layer resources, including operating system support and software development frameworks and IaaS refers to on-demand provisioning of infrastructural resources

B. DeploymentModels

Private cloud: also known as internal cloud, private clouds are designed for exclusive use by a single organization. *Community cloud*: The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns.

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 hybrid cloud is a combination of public and private cloud models

III. STANDARDS OF CLOUD COMPUTING

In previous work [1] we have studied the different cloud solutions. All of them has changed how to deploy and use the services (pay-per-use, automatic scalability) in cloud. But the main barrier remains Interoperability and Portability. Indeed, Amazon EC2, GoGrid and others are based on proprietary mechanisms of service definition. So users are vendor lock-in. Migration of service from a cloud to another is difficult. To provide flexibility and interoperability between clouds by eliminating vendor lock-in, we should using Standards.

A. OCCI: Open Cloud Clomputing

OCCI is developed by the Open Grid Forum (OGF). The development of OCCI follows the idea of the fast provision of a minimalistic interface, which is extended later and therefore offers high extensibility. (OCCI) is a RESTful Protocol and API for all kinds of management tasks. OCCI was originally initiated to create a remote management API for IaaS model-based services, allowing for the development of interoperable tools for common tasks including deployment, autonomic scaling and monitoring. It has since evolved into a flexible API with a strong focus on interoperability and high degree of extensibility [2].

B. OVF: Open Virtualization Format

OVF describes an open, secure, portable, efficient and extensible format for the packaging and distribution of

978-1-4799-6123-8/14 $31.00 © 2014 IEEE
DOI 10.1109/HPCC.2014.48

software to be run in virtual machines. Properties of OVF are:
Portable VM packaging: OVF is virtualization platform neutral, while also enabling platform-specific enhancements to be captured. It supports any open virtual hard disk formats. Virtual machine properties are captured concisely using OVF meta-data [3].
*Optimized for secure distribution***:** OVF supports content verification and integrity checking based on industry standard public key infrastructure, and provides a basic scheme for management of software licensing.
Simplified installation and deployment: OVF streamlines the installation process. During installation, metadata in the OVF file can be used to validate the entire package and automatically determine whether the virtual appliance can be installed.
Supports both single VM and multi-VM configurations: OVF supports both standard single VM virtual appliance and packages containing complex, multi-tier services consisting of multiple interdependent VMs.
*Vendor and platform independent***:** The OVF does not rely on the use of a specific host platform, virtualization platform, or guest operating system.
*Extensible***:** OVF is immediately useful – and extensible. It is designed to be extended as the industry moves forward with the virtual appliance technology. It also supports and permits the encoding of vendor specific meta-data to support specific vertical markets.
*Localizable***:** Supports user visible descriptions in multiple locales, and supports localization of the interactive processes during installation of a virtual appliance. This allows a single packaged virtual appliance to serve multiple market opportunities.

OVF Environment

The OVF environment defines the interaction between the guest software and the deployment platform. This environment allows the guest software to access information about the deployment platform, for example, to access the user specified values for the properties defined in the OVF descriptor. The environment also allows the guest software to provide feedback to the deployment platform (for example, during the software installation phase). The environment specification is split into a protocol part and a transport part. The protocol defines the format and semantics of the documents and messages that can be exchanged between the guest software and deployment platform. The transport defines how the information is communicated between the guest software and deployment platform.

Environment Document Protocol: The environment document is an extensible XML document that is provided to the guest software about the environment in which it is being executed [3].

C. CDMI

The *Cloud Data Management Interface* (CDMI) is a standard of the Storage Networking Industry Association (SNIA), and its purpose is the founding of a basis for rich cloud storage management interfaces [4]. CDMI is intended to be implemented above or besides other storage interfaces even if they are not applicable to clouds. The Cloud Data Management Interface defines the functional interface that applications will use to create, retrieve, update and delete data elements from the Cloud. As part of this interface the client will be able to discover the capabilities of the cloud storage offering and use this interface to manage containers and the data that is placed in them.

IV. RELATED WORKS

Several studies have been developed to find the best manner to satisfy customer's requirement. Indeed, Virtual Appliance as a complete package (OS preinstalled preconfigured delivered with a software solution in a virtual machine) allows customers to download and deploy their applications with minimal effort. VA avoid user to do tasks of installation and configuration. The use of Virtual Appliances for the deployment of cloud services has generated OVF.

Changua Sun et al. [5] study the process of service deployment with and without virtual appliance. They compare results of deployment process with traditional deployment and with virtual appliances. The results show virtual appliances offer significant advantages for service deployment by making the deployment process much simpler and easier, even for the deployment of advanced enterprise services.

Fermín Galá et al. [6] propose a service specification language for cloud computing platforms OVF based on the DMTF's standard. They extend OVF to address the specific requirements of cloud's environments. To assess the feasibility of their proposal, they have implemented a prototype system able to deploy and scale service specifications using their proposed extensions. Authors present results based on practical industrial case study that demonstrates that using the software prototype can automatically deploy and flexibly scale the Sun Grid Engine application.

Diego Kreutz and Andrea Char [7] propose a software system to manage virtual appliances in heterogeneous virtualized infrastructure. This system, named FlexVAPs, provide to costumers a flexible and adaptable virtual appliance management system. This system allow to instantiate virtual machines over different types of virtual machine monitors. Authors establish a comparison among FlexVAPs and two major solutions targeted to virtual appliance management, so as to highlight their benefits and drawbacks.

Amir Vahid Dastjerdi et al. [8] propose an effective architecture using ontology-based discovery to provide required QoS deployment of appliances on Cloud service providers. They offer an approach which gives enough flexibility to end users to discover their needed appliance from range of providers and dynamically deploy it on different IaaS providers.

Maicon Stihler et al [9] describe a new architecture that provides a full federated identity management to assist multi-domain clients in a multi-vendors environment. Their proposition eases identity management without losing flexibility, offers better user tracking through the whole cloud

computing layers, and enables the implementation of multi-provider environments through account data replication.

Tobias Kurse et al [10] define the federation in the cloud as a concept of service aggregation characterized by interoperability features, dealing with economic problems vendor lock-in and vendor integration. In addition, the authors discuss a reference architecture that enables new service models using the horizontal and vertical integration to create value-added software solutions.

V. ARCHITECTURE FOR SERVICE DEPLOYMENT IN FEDERATED CLOUD

The concept of cloud federation that two or more independent cloud providers can work together to create a federated cloud. Participants in the federation that have excess capacity can share their resources for an agreed price, with participants who need additional resources.

There are two basic dimensions of Cloud Federation: horizontal, and vertical. While horizontal federation takes place on one level of the Cloud Stack, the application stack, vertical federation spans multiple levels. In the following we focus on horizontal federation; aspects of vertical federation are out of the scope of this publication.

In our case, we consider a particular complex service that's called virtual Appliance.

A. Virtual Appliance

A virtual appliance is a pre-integrated software solution, consisting of one or more virtual machines that are packaged, maintained, updated and managed as a unit as illustrated below in figure 1 [11]. Deploying a software solution in a virtual appliance is to build a complete package (an operating system OS preinstalled, preconfigured with a software solution delivered in a virtual machine) that allows software developers to ship solutions preinstalled, preconfigured enabling customers to immediately download and deploy their applications in their environments with minimal effort, avoiding tasks of installation and configuration.

Thanks to hardware virtualization, a new option is emerging that leverages the advantages of hardware appliances without the hardware. It lets ISVs package a fully configured and optimized software stack in a virtual server format that can be quickly deployed at the customer site on top of VMware ESX, Xen, or another hypervisor.

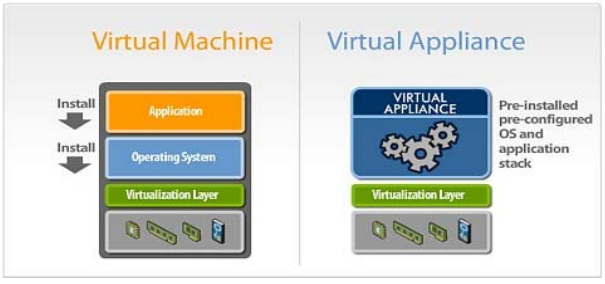

Figure 1. Virtual Appliance

B. Proposed Architecture and Discussion

Based on the state of the art [4], we opted to use the Virtual Appliance for the deployment of services in the cloud. We will also use OVF as a mechanism for representing the cloud's service. We intend to use OVF to express the needs of the user and the interface. We also consider complex needs based on services deployed on Cloud federation. We will try to include OVF extensions to meet the requirement of the cloud environment as shown in figure 2. We will add specifications to assume cloud's federation with QoS constraints predefined in advance like (network parameters).

Indeed we assume a scenario, where the user expresses his needs in resources through a web interface. Their need will be translated into an OVF package and will be passed to the cloud broker who will analyze the OVF package, identify the characteristics and the rules and do the matching between this information, the network QoS constraints and available resources for IaaS . Matching should be between user's requirement and IaaS provider maintaining a certain QoS (predefined in advance).

Our architecture shown in the figure 4 is able to deploy virtual appliance in the federated cloud. The lock is to develop an algorithm which does matching between requirement resources by VA deployment and available resources from different cloud providers. The goal to find a transparent environment to user like illustrated in figure 3.When user deploys its VA, he isn't bothered by tasks of installation and configuration. These tasks are transparent for end user.

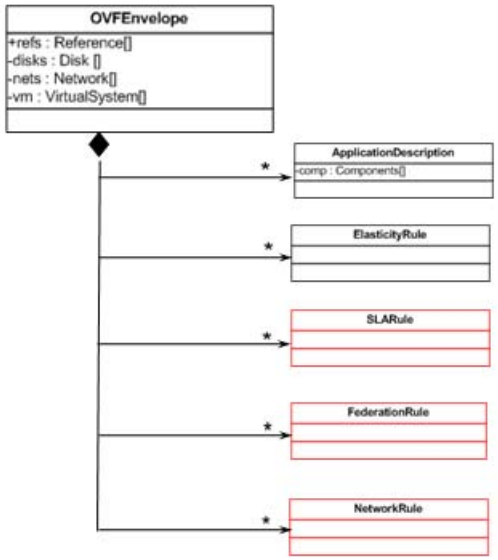

Figure 2. OVF's extension

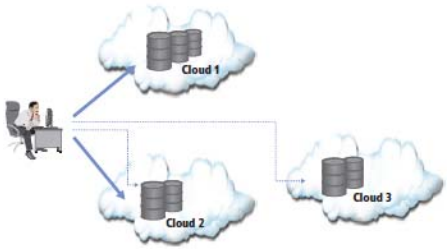

Figure 3. Federated cloud

Our architecture is composed by these key elements:

Broker Management: manages requests of service and resources. It is responsible to do matching between requests made by the user in the manifest and information reported by the information collector of each IaaS provider.

Information Collector: Collects information about the deployment environment (VM states, service status, ...). This component exists in every IaaS.

OVF processing engine: process the Service Manifest, save rules in rule engine then it passes it to broker management for processing (requirements for deployment, resources needed,).

VM monitor: is associated at each VM queue. It performs the dynamic creation and destruction of VM and their queue for each VA (for all instances of services).

Rules engine: contains all the rules of elasticity, federation and networks.

The process is detailed on 6 steps.

1/OVF processing engine analyzes the service Manifest and produces an internal representation of the service.

2/Broker management receives the result of the analysis, develops and installs rules elasticity, federation and networks specified in the manifest.

3/ Broker management gets images required for VM which compose service. Reference to these images and personalization data are retrieved from the service Manifest.

4/Broker management sends its request to the deployment platform to create a new VM.

5/Cloud provider gets the basic disk of VM then VM monitor creates and handles the deployment.

6/customization disk is attached to the VM to permit VM monitor to customize and configure correctly VMs.

I. CONCLUSION AND FUTURE WORKS

In this paper we summarized some concepts related to virtualization and we introduce our proposed architecture which allows deploying VA into federated cloud. In future works we plan to propose an algorithm for matching between available resources from cloud provider and required resource by VA respecting the existing rules in rule engine.

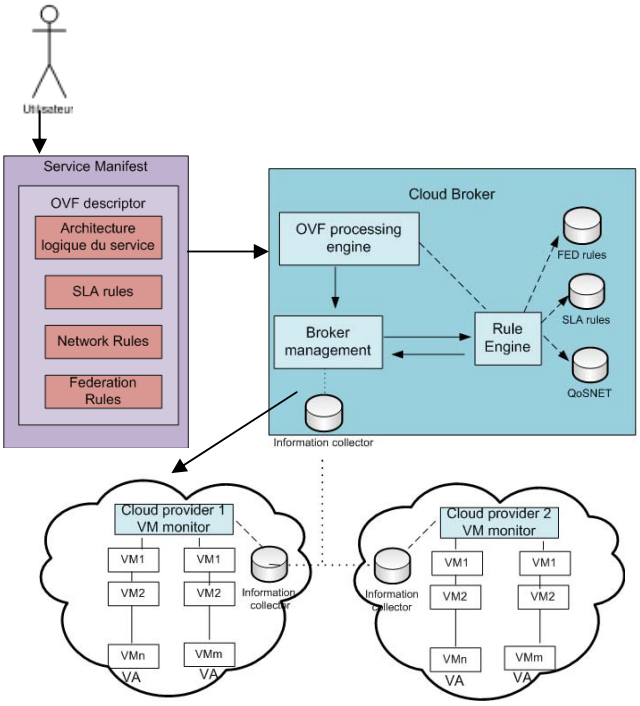

Figure 4. Proposed Architecture

REFERENCES

[1] Amel Haji, Asma Ben Letaifa, Sami Tabbane, “Implementation of a virtualization solution”, IEEE WMNC 2011.

[2] Thijs Metsch, Andy Edmonds, “Open Cloud Computing Interface – Infrastructure”, 2011.

[3] The Open Virtualozation Format Specification, DSP0243, DMTF,2009

[4] Storage Networking Industry Association, “Cloud data management Interface,” , Tech. Rep., 2010.

[5] Changhua Sun, Le He, Qingbo Wang, Ruth Willenborg, “Simplifying Service Deployment with Virtual Appliances”, IEEE International Conference on Services Computing, 2008.

[6] Fermín Galán Americo Sampaio Luis Rodero-Merino, Irit Loy Victor Gil Luis M. Vaquero, “Service Specification in Cloud Environments Based on Extensions to Open Standards”, ACM COMSWARE 09 June 16-19, Dublin, Ireland, 2009.

[7] Diego Kreutz, Andrea Char, "FlexVAPs: a system for managing virtual appliances in heterogeneous virtualized environments”, IEEE 2009.

[8] Amir Vahid Dastjerdi, Sayed Gholam Hassan Tabatabaei, Rajkumar Buyya, “An Effective Architecture for Automated Appliance Management System Applying Ontology-Based Cloud Discovery”, 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, 2010

[9] M.Stihler, A.O Santin, A.L Marcon, Jd.S Fraga, “Integral Federated Identity Management of Cloud Computing”, 5th IEEE International Conference on New Technologies, Mobility and Security (NTMS), 2012.

[10] Tobias Kurze, Markus Klemsy, David Bermbachy, Alexander Lenkz, Stefan Taiy and Marcel Kunze, “Cloud Federation”, In CLOUD COMPUTING, 2011, pp. 32-38, 2011.

[11] http://www.imagicle.com/tabid/648/Default.aspx , fevrier 2014.