International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:10, No:2, 2016

An Efficient Architecture for Dynamic Customization and Provisioning of Virtual Appliance in Cloud Environment

Rajendar Kandan, Mohammad Zakaria Alli, Hong Ong

***Abstract*—**Cloud computing is a business model which provides an easier management of computing resources. Cloud users can request virtual machine and install additional softwares and configure them if needed. However, user can also request virtual appliance which provides a better solution to deploy application in much faster time, as it is ready-built image of operating system with necessary softwares installed and configured. Large numbers of virtual appliances are available in different image format. User can download available appliances from public marketplace and start using it. However, information published about the virtual appliance differs from each providers leading to the difficulty in choosing required virtual appliance as it is composed of specific OS with standard software version. However, even if user choses the appliance from respective providers, user doesn't have any flexibility to choose their own set of softwares with required OS and application. In this paper, we propose a referenced architecture for dynamically customizing virtual appliance and provision them in an easier manner. We also add our experience in integrating our proposed architecture with public marketplace and Mi-Cloud, a cloud management software.

***Keywords*—**Cloud computing, marketplace, virtualization, virtual appliance.

I. Introduction

CLOUD technology offers a definite platform for dynamically managing the resources and provides wide range of services to users. Rapid provisioning of on-demand compute, network, storage, application and other services attract various organizations and industries to benefit cost saving. Definitions for cloud computing has been put forth by various industries, Vaquero [1] identifies essential characteristics of cloud computing and proposed new definition on cloud computing stating that:

> "*Clouds are a large pool of easily usable and accessible virtualized resources (such as hardware, development platforms and/or services). These resources can be dynamically reconfigured to adjust to a variable load (scale), allowing also for an optimum resource utilization. This pool of resources is typically exploited by a pay per-use model in which guarantees are offered by the Infrastructure Provider by means of customized SLA*".

They also identify the relationship among grid and cloud computing by analysing common features. More detailed description on the characteristics of both grid and cloud computing were discussed by [2] based on business model, architecture, resource management, programming model, application model and security model. NIST [3] also proposed five essential characteristics of cloud and identified three service and four deployment models. Although varied by different definitions, cloud computing focuses on the dynamic management of resources, providing better quality of service to users. In a typical cloud environment, cloud users request virtual machine by specifying operating system, CPU, memory and disk storage. Cloud services should therefore manage a wide range of images for provisioning them to users. However, if user wants to develop applications it becomes a big hurdle to user in terms of challenging installation and configuration of application with respect to version, software dependencies, OS, and related softwares. Virtual appliances solve these issues as it is a ready built application with all necessary softwares installed and configured. In this case, the user just chooses the appliance and deploys them. Once virtual appliance is deployed, respective application will be up and running, so that the user can readily use it without worrying about the installation and configuration.

Virtual appliance [4] eliminates the critical process of installing and configuration of softwares. Large number of public appliance providers are available in market; each provide a plenty of virtual appliance with wide range of image formats. With the increase in number of appliances, the complexity for users in choosing the required appliance also increases, as each appliance differs from other providers in terms of various parameters viz., Operating system and version. Meta-data information about each virtual appliance varies as each provider uploads their appliance with different set of appliance parameters.

Although current system allows user to choose the required appliance, it restricts the user with the packed versions of software. Therefore, it requires for a platform which could provide the flexibility of configuring software as required by the user and provision them in a much faster and easier manner. In this paper, we propose an architecture which primarily focus on the customization of existing public appliance, configure with the software required and provision the appliances. The collection of information about appliance is first and foremost important part of this process and since information retrieval has been addressed by various initiatives, we will not address it here. However, our main focus is

Rajendar Kandan is with the Advanced Computing Lab, MIMOS, Kuala Lumpur, Malaysia (e-mail: rajendar_bist@yahoo.com).

primarily on customization and ensures the respective softwares were available to user as requested.

The paper is organized as follows Section II describes the virtual appliance in detail followed by list of public marketplace available in Section III. Section IV discuss about the various configuration tools and proposed architecture in Section V. Implementation and case study are described in Section VI followed by conclusion and future work in Section VII.

II. Background and Motivation

Virtual appliance is a pre-configured image consisting of operation system and application. Virtual Appliance is available as an image which could run on specific hypervisor viz., Xen [5], KVM [6], VMWare [7], etc. The building and provisioning of these appliances vary with respect to each hypervisor.

Most of the cloud providers have their own marketplace, which contains list of virtual appliance managed by their community users. Marketplace contains the list of appliances and their associated informations including name, size, format, etc. User can refer marketplace, search for desired appliance and provision them accordingly. One of popular marketplace is turnkey linux[8], which contains wide range of debian based pre-packaged virtual appliances. It manages nearly 100 appliances in various image formats and suitable for deploying across different hypervisors including Xen, VMWare and KVM. It also contains various image builds that could be deployed across virtualization softwares and cloud based environments including Openstack, Amazon, LXC, OpenVZ and Docker.

Other popular marketplace includes Bitnami [9], which contain virtual appliances in different builds as native installer, virtual machines and cloud images. The appliances were ready to deploy over wide ranges of cloud platform including Amazon, Microsoft Azure, Google cloud platform, VMware vCloud, Openstack and Cloudstack. OpenNebula marketplace [10] is another marketplace providing customizable virtual appliances which are ready to deploy on OpenNebula cloud environment [11]. It contains appliances supporting hypervisors KVM, XEN and VMware. Users also have the provision to either download images directly from the marketplace or register as images directly to the OpenNebula cloud environment. Amazon marketplace [12] also provides wide range of virtual appliances facilitating cloud users to deploy appliances in their own cloud environment in much easier way. All appliances available are of Amazon image (AMI) format which are ready to deploy on Amazon cloud environment. Similarly, VMware marketplace [13] also provides various appliances which can only be deployed across VMWare cloud environment. Although different marketplaces are available, our main motivation is that there is no single architecture which can address the customization of existing appliance against the user request.

Customizable virtual appliance provides facility for cloud user to upgrade or install new softwares. However, there are many open source configuration tools available in market which manages the configuration of application in a much easier way. Some of these configuration management tools are described in section III.

III. Configuration management tools

Appliances have pre-built set of softwares installed and configured, however if the user want to install or update any new softwares over the appliance, there requires a configuration management (CM) tools which could provide the automation of configuration of any software applications. These tools were designed to eliminate complexity in governing multiple applications and provide an efficient way of installing or upgrading any applications without any errors.

Popular configuration tool includes Puppet [14], a model-based configuration management tool which can run on all major Linux and UNIX platforms, Mac OS X and windows. Puppet is written in ruby and available in both open source and enterprise versions. It automates all steps of software delivery process ensuring consistent, reliability and stability. User can provide their requirements using file called “manifest” which contains the syntax in JSON like format. It follows the client/server architecture which can compile these instructions listed at “manifest” files and ensure the installation and configuration of application at the desired client machines. It also provides the complete list of operation and their status on the client machines. Chef [15] is another popular configuration management tool developed using ruby. Free and enterprise versions of chef are also available in market similar to Puppet. It also follows client/server architecture and contains components Chef server and Chef nodes respectively. Chef server stores the configuration data and manages chef nodes for installation and configuration of desired software applications. There were more configuration management tools available in market including Ansible, Salt, Vagrant, etc.

Although the configuration tools are available for installation of additional new softwares, there is no such architecture for removal of existing application from the appliance and manage them efficiently for re-configuring appliance as required by the user. Hence we propose architecture for managing the configuration of application on public appliance and customize as required by the user. We have also used puppet configuration management tool which is based on ruby implementation and customized it for managing the entire operation of re-configuring the appliances. Following section describe our proposed architecture.

IV. Proposed Configuration Management Architecture

We proposed architecture a Dynamic Configuration management (DCP) for managing on demand configuration of softwares required by the user. The architecture is as in Fig. 1.

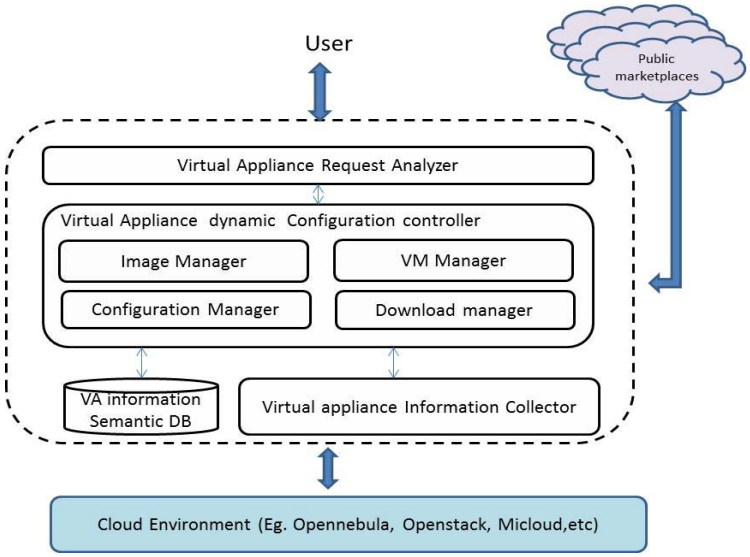

Fig. 1 DCP architecture

Proposed architecture DCP consists of three main components, namely Virtual appliance request analyser (VARA), Virtual appliance dynamic configuration controller (VADCC) and virtual appliance information collector (VAIC).

VARA handles the user request regarding virtual appliance and softwares. It ensures that the proper syntax is provided by the user. It transforms and store requested information before proceeding to the next component VADCC.

VADCC is the main component that manages the dynamic configuration of softwares across virtual appliance. VADCC includes sub-components Download manager, VM manager, Image manager and configuration manager. Download manager is responsible for downloading images from public marketplaces. Image manager handles the management of appliance images across the local site. Configuration manger is responsible for generation of contextualization script for re-configuration on desired virtual appliances.VM Manager is responsible for the sending request regarding VM creation and monitoring the status of configurations.

VAIC communicates to the external marketplace and contains the required information about all public appliances available across different marketplaces. This component contains required API for collecting the information across Turnkey linux and Bitnami site. We have used Jsoup, open source Java API for extracting information about the appliance information published across the marketplaces. Information about each appliance is then transformed and stored in semantic database which aid in searching the appliance much faster and helps to identify the relationship among each applications and appliances.

V. Implementation and Case Study

In our implementation, we have integrated our proposed architecture with Mi-Cloud, open source cloud infrastructure management software [16]. Mi-Cloud consists of two major components *front-end*, which manages the cloud operations and *node*, where virtual machines are deployed. The architecture of integrated Mi-cloud and DCP are as follows

DCP semantic database consists of appliance information available across turnkey linux and bitnami marketplaces. It consists of all meta-data information about applications available across each appliance. User request appliance and softwares, DCP manages the image selection and contextualization of images. Provisioning of virtual appliance is managed by Mi-Cloud system.

User requests the list of software and base operating system. The algorithm for processing and provisioning of appliance is as:

Algorithm:

I. *User request the softwares S_R (where R=1,…N Integer)*
II. *Select appliance A from marketplace MP_i based on $S_A >= S_R$, where A is the appliance and S_A is the list of softwares available at appliance A. Procced to step V.*
III. *If no match found at Step II. Select appliance B from marketplace MP_i based on $Max(S_B) <= S_R$, where $Max(S_B)$ is the maximum list of requested softwares available at appliance B. Procced to step V.*
IV. *If no match found at step III, select appliance C based on $OS_c = OS_R$, where OSc is the operating system of appliance C and OS_R is requested operating sytem. Proceed to step V*
V. *Prepare puppet manifest configuration file depending on virtual appliance and requested software.*
VI. *Prepare configuration file as VM context and request for VM provision*
VII. *Monitor status of virtual machine and application configuration*
VIII. *Update status of application configuration*

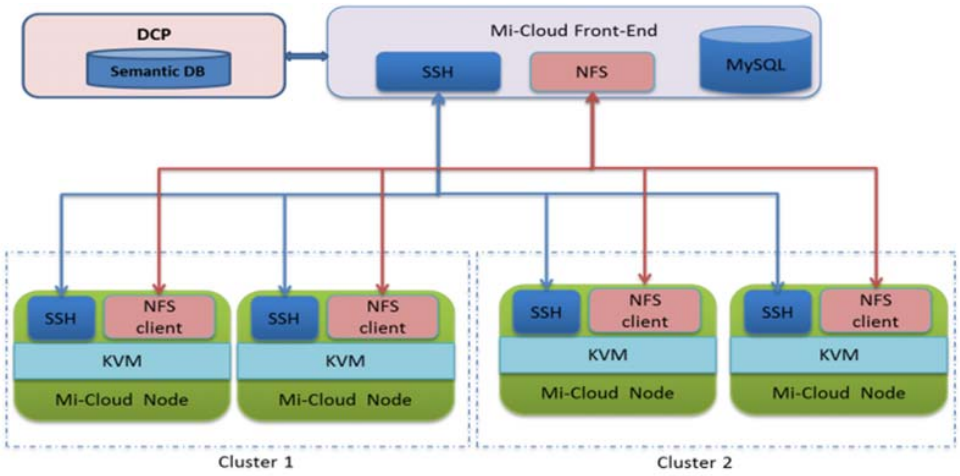

Fig. 2 DCP integration with Mi-Cloud

Table I shows the sample list of appliance, description, major softwares installed and operating system of appliance. These appliances are available at turnkey linux marketplace.

TABLE I
TURNKEY LINUX VIRTUAL APPLIANCES AND SOFTWARES

Virtual Appliance	Description	Softwares	Operating system
Drupal	Drupal is an open source content management platform licensed under the GPL	turnkey-drupal7-13.0 turnkey-core-13.0 Apache2 Mysql-server-5.5 Openssl php5-common Phpmyadmin python2.7	Distributor ID: Debian Description: Debian GNU/Linux 7.2 (wheezy) Release: 7.2 Codename: wheezy
Joomla	Joomla! is an award-winning Content Management System (CMS) for building websites as well as a Model-view-controller (MVC) Web Application Development framework.	turnkey-joomla25-13.0 turnkey-core-13.0 Apache2 Git Mysql-server-5.5 Openssl php5-common Phpmyadmin python2.7	
Apache tomcat	Tomcat is a servlet container that implements the Java Servlet and the JavaServer Pages (JSP) specifications, and provides a "pure Java" HTTP web server environment for Java code to run in.	Apache2 tomcat6 mysql-server mysql-server-5.5 openjdk-6-jdk:amd64 openjdk-6-jre:amd64 openssh-server python2.7	
Mysql	MySQL is a relational database management system (RDBMS) which has more than 11 million installations, and is owned by Oracle.	turnkey-core-13.0 turnkey-mysql-13.0 mysql-server-5.5 Openssl openssh-server php5-common Phpmyadmin python2.7	

Table I describes the sample list of turnkey linux appliances and major softwares available with respect to each appliance. DCP works on the stated algorithm. Whenever the user request for list of softwares, DCP first checks with the existing appliance and their list of softwares and matches the best suited appliance which contains all the software required by the user and provision them using Mi-cloud infrastructure. If required appliance is not available, it checks for the appliance which contains the maximum list of softwares available as requested by the appliance. After the selection of appliance, it generates the puppet configuration script for installation/ updation of remaining softwares, if needed. If all softwares were already available at appliance, it generates the configuration script for cleaning up the appliance. It then finally transforms configuration scripts as contextualization and send the VM provisioning request to Mi-cloud. If no softwares were available at appliance, DCP then selects the base OS as requested by user and prepares the configuration file and this condition is more similar to any configuration management tool. DCP mainly focus on the analysing the appliance software against the list of softwares requested by user, where by minimal effort of dynamic configuration of application is employed and adds advantage over the normal configuration softwares. We also have estimated the configuration time required for processing the installation of applications against the re-configuration of appliances. In this example, we calculated the standard installation time required for software package “openjdk-6-jdk” on Ubuntu 14.04.1 LTS (Trusty). Further we calculated the time required for re-configuring the appliance “Apache tomcat” described in Table I.

Table II, compares the configuration time required for installation of new software against the re-configuration of ready built appliance. From our sample results, application can be provisioned in much reduced time instead of following the standard installation of softwares. Also the installation of software and re-configuration varies with respect to application and hence DCP architecture manages the provisioning of softwares based on following conditions

i. Re-Configuration of softwares is carried if only the standard installation of softwares requires more amount of time.

ii. Only those softwares which are independent of required softwares list are processed for removal.

TABLE II
COMPARISON OF SOFTWARES INSTALLATION AND RECONFIGURATION

Type	Application	Time (Sec)
Installation	openjdk-6-jdk	464
	E.g.: Apache tomcat appliance is selected for re-configuration	
Re-Configuration	1. Identification of softwares to remove	15
	2. Removal of Apache2	20
	3. Removal of tomcat	9
	4. Removal of Mysql-server	10
	5. Verification of softwares	20
	Total Re-Configuration	74

Thus DCP helps in configuring the softwares using existing appliance and configure them as requested by user in a much reduced time. Further, as illustrated in Table I, if user request for any combination of application, DCP can manage installation/re-configuration of applications by using only three images. Traditional approach requires four images for processing the user requested applications. Hence by leveraging meta-data information about each appliance, the image can be well handled at the local site and configure them in a much reduced time.

VI. CONCLUSION

In this paper, an architecture for dynamic configuration and provisioning of virtual appliance is presented. The proposed architecture addresses the utilization of existing appliance from marketplaces to match against user requested software and provide a basic approach for dynamically configuration the requested softwares using puppet configuration management software. This approach benefits user in requesting the softwares in addition to the appliance in a dynamically manner and provision them much faster. It also addresses how similar type of virtual appliance can be utilized for optimal usage, which could be very useful to data centres, where it requires the internal storage of appliance for future provisioning. In future, we plan to extend for multiple marketplaces, softwares and operating systems to analyse the efficient way of re-configuring the appliance. In addition, we also plan to investigate on multi-tier appliance provisioning using SLA for better QoS to cloud users.

ACKNOWLEDGMENT

The authors sincerely thank MIMOS BERHAD, for financially supporting the Advanced Computing Laboratory, MIMOS BERHAD, Kuala Lumpur, Malaysia for carrying out research.

REFERENCES

[1] L. Vaquero, L. Rodero-Merino, J. Cáceres, and M. Lindner, “A Break in the Clouds: Towards a Cloud Definition”, SIGCOMM Comput. Commun. Rev.,2009, Vol. 39, No. 1, pp. 50-55
[2] I. Foster, Y. Zhao, I. Raicu, and S. Lu, “Cloud Computing and Grid Computing 360-Degree Compared”, IEEE Grid Computing Environments Workshop, 2008, pp. 1-10.
[3] http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf
[4] *Scannell, Ed (March 10, 2009).* "TurnKey Linux Delivers Open Source Appliances" .InformationWeek. Retrieved March 23, 2009
[5] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, Andrew Warfield, “Xen and the art of virtualization”, Proceedings of the nineteenth ACM symposium on Operating systems principles, October 19-22, 2003, Bolton Landing, NY, USA.
[6] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori. “KVM: The Linux virtual machine monitor”., OLS '07: The 2007 Ottawa Linux Symposium, pages 225--230, July 2007
[7] Jeremy Sugerman, Ganesh Venkitachalam, Beng-Hong Lim, “Virtualizing I/O Devices on VMware Workstation's Hosted Virtual Machine Monitor”, Proceedings of the General Track: 2002 USENIX Annual Technical Conference, p.1-14, June 25-30, 2001
[8] "Introducing TurnKey Linux Appliance Library". *OpenNode Cld Platform. 18 October 2013*. Retrieved 8 February 2014
[9] Jacob (2007-09-27). "Get it done with BitNami Stacks". FOSSwire. Retrieved 2008-07-12.
[10] J. Tordsson, R.S. Montero, R. Moreno-Vozmediano, I.M. Llorente. Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. Future Generation Computer Systems, 28(2):358-367, 2012.
[11] Rafael Moreno-Vozmediano, Rubén S. Montero, Ignacio M. Llorente, "IaaS Cloud Architecture: From Virtualized Datacenters to Federated Cloud Infrastructures", Computer, vol.45, no. 12, pp. 65-72, Dec. 2012, doi:10.1109/MC.2012.76
[12] "Amazon Web Services". *AWS Products. Amazon Web Services*. Retrieved 23 April2015.
[13] "Open Virtualization Format (OVF) -Virtual Machines - Virtualization". Vmware.com. Retrieved 2011-12-09.
[14] *Dehaan, MP.* "Deploying Apache Tomcat Applications with Puppet". *tomcatexpert.*
[15] Gregory Katsaros, Michael Menzel, Alexander Lenk, Jannis Rake Revelant, Ryan Skipp, Jacob Eberhardt, “Cloud Application Portability with TOSCA, Chef and Openstack”, IC2E’ 14 Proceedings of the 2014 IEEE International Conference on Cloud Engineering, pages 295-302.
[16] Kandan, Rajendar, Mohammad Zakaria Alli, and Hong Ong. "DiAF: A Dynamic Virtual Appliance provision and management Framework for cloud Computing." Proceedings of the International Conference on Grid Computing and Applications (GCA). The Steering Committee of the World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2014, Las Vegas, USA.