Future Generation Computer Systems 29 (2013) 323–329

Contents lists available at SciVerse ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Automatic software deployment using user-level virtualization for cloud-computing

Youhui Zhang *, Yanhua Li, Weimin Zheng

Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

ARTICLE INFO

Article history:
Received 1 November 2010
Received in revised form
26 March 2011
Accepted 5 August 2011
Available online 5 September 2011

Keywords:
Cloud computing
User-level virtualization
Virtual machine
Deployment

ABSTRACT

Cloud Computing offers a flexible and relatively cheap solution to deploy IT infrastructure in an elastic way. An emerging cloud service allows customers to order virtual machines to be delivered virtually in the cloud; and in most cases, besides the virtual hardware and system software, it is necessary to deploy application software in a similar way to provide a fully-functional work environment. Most existing systems use virtual appliances to provide this function, which couples application software with virtual machine (VM) image(s) closely.

This paper proposes a new method based on the user-level virtualization technology to decouple application software from VM to improve the deployment flexibility. User-level virtualization isolates applications from the OS (and then the lower-level VM); so that a user can choose which software will be used after setting the virtual machines' configuration. Moreover, the chosen software is not pre-installed (or pre-stored) in the VM image; instead, it can be streamed from the application depository on demand when the user launches it in a running VM to save the storage overhead. During the whole process, no software installation is needed. Further, the enormous existing desktop software can be converted into such on-demand versions without any modification of source code.

We present the whole framework, including the application preparation, the runtime system design, the detailed deployment and usage workflow, and some optimizations. At last, test results show that this solution can be efficient in performance and storage.

© 2011 Elsevier B.V. All rights reserved.

1. Introduction

Infrastructure cloud service providers (e.g., [1,2]) deliver virtual hardware and software in their datacenters, based on the demand from customers. Then, customers avoid capital expenditure by renting usage from the provider and they consume resources as a service.

Usually, besides virtual hardware and system software, it is necessary to deploy application software in a similar way; therefore customers can get a fully-functional work environment conveniently with the required application software.

Most existing solutions allow cloud customers to order Virtual Appliances (VAs) [2–4] to be delivered virtually on the cloud. For example, VA marketplaces [2,5,6] provide lots of categories of appliances, and each is a pre-built software solution, comprised of one or more Virtual Machines that are packaged. VA-based methods can reduce time and expenses remarkably associated with application deployment.

However, because VA couples the application software and VMs closely, it also has some drawbacks:

(1) *Lack of flexibility*. For example, a customer needs software A and B to work together in a virtual machine, while the provider only has two separate VAs containing A and B respectively. Then, the provider has to create a new VM template to combine A and B together. In theory, such combinations are countless.

(2) *Inefficiency of storage*.

Each VA comprises one VM image at least, which means the OS has to be combined in the image. Therefore, the storage overhead is larger, although some technologies (e.g., Just enough OS [7], De-duplication [8,9]) have been employed to reduce the overhead.

The essential reason of these drawbacks lies in that the VA solution heavily depends on the virtual machine technology and the latter only isolates system software from hardware. Therefore application software has to be packaged in the whole system for deployment.

To solve this problem, this paper introduces a double-isolation mechanism that uses the user-level virtualization technology to further isolate application software from OS, while the VM-level isolation is still kept. Therefore, application software can be deployed in a fine granularity to increase the flexibility and decrease

* Corresponding author. Tel.: +86 10 62783505x3.
E-mail address: zyh02@tsinghua.edu.cn (Y. Zhang).

0167-739X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2011.08.012

the storage overhead. In this paper, we call such application software as *on-demand software*.

Based on this design philosophy, we make the following contributions:

(1) *The whole deployment framework based on the double-isolation mechanism*.

The deployment of application software on user-level virtualization is the focus. It includes the on-demand software preparation, deployment, runtime system, customization and usage accounting.

(2) *User-level virtualization of on-demand software*.

Some essential technologies, like converting legacy software into the on-demand style or the runtime system of user-level virtualization, are implemented. Especially, our methods can support existing application software without any modification of source code.

(3) *A central distribution system for on-demand software*.

One or more central data servers are used to provide software on demand for the deployed virtual machines, rather than place software within VMs in advance. Because of the commonality of frequently-used applications in the Cloud Computing environment, this technology can decrease the storage consumption significantly.

Moreover, some access optimizations, including the content-addressable storage and local cache, are presented, too.

(4) *The system prototype*.

In addition, tests show that this solution is efficient in performance and storage.

In the following sections, we first present the whole framework and the user-level virtualization technology for on-demand software. The central distribution system for Cloud Computing and the related optimizations are given in Section 3. The prototype is introduced in Section 4, as well as the performance tests. Section 5 gives related work; the conclusion and future work are presented finally.

2. The framework

2.1. Software deployment overview

To deploy on-demand software in the Cloud Computing environment, it is necessary to provide a system to own the following functions:

(1) *Software preparation*.

Most existing software needs to be installed before it can run normally. However, in our design, the on-demand software requested by a customer can be used instantly without any installation process. Thus, we should convert software into the on-demand mode in advance, and all on-demand software is stored in the software depository for users' selection.

The details are presented in Section 2.2.

(2) *Software selection*.

For most existing cloud service providers, a customer usually chooses one or more VAs before deployment, which means that the required software and its lower-level VM(s) are selected at once.

In contrast, we provide a more flexible selection procedure: a customer can choose the wanted OS, as well as any number of software in separated stages. For example, Lisa orders a Windows VM as her remote work environment on the cloud; and then she can select any on-demand software (only if it can run in the Windows OS) that she will use in the VM. It means that we can provide any combination of VM and software, rather than depend on the limited number of existing VM templates.

(3) *On-demand deployment and usage accounting*.

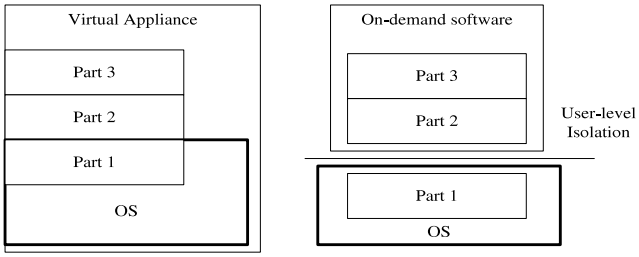

Fig. 1. On-demand software and virtual appliance.

After the preparation and selection, software is not to be stored in the VM image (like the Virtual Appliance does). In contrast, one or more central data servers are used to provide software on demand for the deployed virtual machines. It means only when the customer actually uses the chosen software, will it be streamed from the data server and run locally without installation. In other words, on-demand software is stored remotely and run locally; a local cache is also used to improve the access performance.

Inherently, this deployment mode enables a fine-grained billing mechanism: the accurate running time of any on-demand software can be gotten and used as the accounting basis.

The technical details are presented in Section 2.3 about the runtime design.

(4) *Software customization*.

Another problem of the VA-based solution is how to save the user's customization. When Lisa finishes her work, she wants to terminate the rent agreement, but reserve her customization of application software, like the default homepage, browser bookmarks/history, cookies and even toolbars' positions, etc.; then it is possible for her to restore these favorites when she rents the same virtual environment again.

For the VA-based solution, it is difficult to implement this function efficiently. One way is to use application-specific tools to draw the customized configurations [10]. Another is to save the difference between the current VM image and the original one, which will contain too much unrelated data.

We solve this problem through the runtime environment based on user-level virtualization, which is independent of the concrete software and achieves higher storage efficiency.

The details are presented with the runtime design in Section 2.3.

2.2. Preparation of on-demand software

According to the on-demand software model we presented in [11], any software can be regarded as containing three parts: Part 1 includes all resources provided by the OS; Part 2 contains what are created/modified/deleted by the installation process; and Part 3 is the data created/modified/deleted during the run time. The resources here mainly refer to files/folders, environment variables and/or the related system registry keys/values (for Windows OS).

Because the traditional solution only depends on the VM, it has to carry the OS image in order to take Part 1, as well as Part 2, to construct the whole virtual appliance.

For the new solution, user-level virtualization isolates application software from the OS and our solution only makes software run on compatible hosts (which implies that all resources of Part 1 are available on the local system), so that only Part 2 is needed to be drawn to build the on-demand software. The difference between these two solutions is illustrated in Fig. 1.

An installation snapshot is taken to build the on-demand software: we start with a machine in a known state (for example, immediately after the OS was installed); then, we install software and finally identify everything that was added to the system by that process of the installation. Typical additions mainly consist

Download English Version:

https://daneshyari.com/en/article/10330616

Download Persian Version:

https://daneshyari.com/article/10330616

Daneshyari.com