Available at: www.researchpublications.org

# Auto Provisioning Portal

Neha Chapke[#1]

[#] *Shri Sant Gajanan Maharaj College of Engineering , Shegaon, India*

[1]hi.tinachapke123@gmail.com

***Abstract*— A solution to automate the infrastructure of a private cloud in the form of a tool implemented by us is presented in this article. We define and describe in this document our project which is Auto Provisioning Portal where we are provisioning resources like Operating System and memory in a manner specified and requested by the clients of the system. These resources are provided in the form of a Virtual Machine. This virtual machine can be accesses by the clients using the user credentials offered to them via their accounts on portal. Clients can specially benefit from services presented to them in the guise of extending the virtual machine with extra time or memory.**

***Keywords*— Private Cloud, Auto Provisioning, Virtualization, Resource Pooling**

## I. INTRODUCTION

We live in dynamic times where the success of any enterprise hugely depends on its ability to adapt to the changing environment and extract benefits. An organization has different departments handling diverse functionalities. The requirements of these units may dwindle as the market circumstances change. An organization can expertly tackle such conditions using the concept of private cloud. Private cloud is an infrastructure operated solely for a single organization, whether managed internally or by a third-party and hosted internally or externally. Private cloud offers the greatest level of security and control, but it may require the company to purchase and maintain all the software and infrastructure and so the existing resources of companies can also be utilized.

Virtualization is a prime technology in Cloud Computing that makes the major benefits of cloud possible as the resources are consolidated and allocated to requesting entities as per their needs. Thus the hardware is efficiently used giving reductions in cost. There are multiple tools that implement virtualization on resources and provide virtual machines for use. A better means is to implement a cloud management tool that automates creation and supervision of virtual machines. The actual designing and application of an automation strategy is a complex task needing something more than this.

Xen Cloud Platform is one cloud management tool that provides facilities to configure the cloud architecture using application program interfaces. This configuration needs to be done manually by a person having proper prior knowledge about the working of the tool. This person- the administrator will have to interact with XCP through the command line interface or graphic interface tool and execute each and every command for the features provided and ascertain that the stages are executed in order, completely and successfully. The intervention of man in each process thus prohibits complete automation of process. Manual involvement can also introduce potential bugs and require more time for task completion than actually desired. The progress of processes will depend on the administrator which can be a bottleneck.

The figure shows a typical scene of provisioning with the administrator.

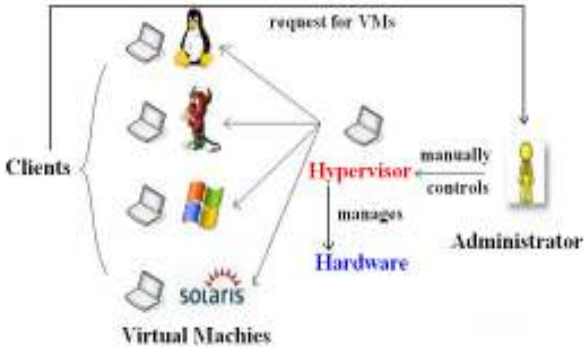


Fig. 1 Provisioning scenario before automation

A still better proposition to completely automate the virtual machine provisioning is the tool-"VAMP" Auto Provisioning Portal, offered by us.

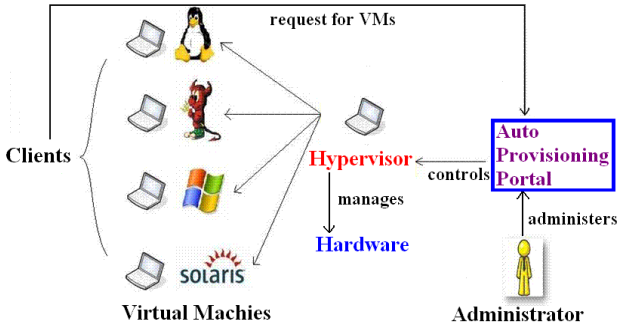


Fig. 2 Provisioning scenario after automation

Available at: www.researchpublications.org

VAMP Auto Provisioning Portal introduces computerization and reduces the workload on the administrator. APP via the portal offers the clients an instance of the "self-service" model where they specify their requirements for virtual machine. The requests are then realised automatically by the system but scheduling and executing all the steps in a timely manner. Requests can be executed efficiently in bulk requiring minimal assistance from administrator who can solely concentrate on task of accepting/ rejecting the requests.

## II. XEN CLOUD PLATFORM

VAMP Auto Provisioning Portal makes use of Xen Cloud Platform for managing virtual machines. The Xen Cloud Platform raises the bar of a cloud platform by going beyond the hypervisor to deliver a complete run-time virtual infrastructure platform product that virtualizes storage, server and network resources. Developed as part of the Xen Cloud Project introduced in 2009, XCP provides a solution for small and medium size enterprises looking to build private clouds, as well as open source enthusiasts, universities and researchers wanting to experiment with cloud computing. The Xen Cloud Platform (XCP) manages storage, VMs and the network in a cloud. XCP does not provide the overall cloud architecture, but rather focuses on configuration and maintenance of clouds.

Figure8. shows where XCP is stationed in the system. XCP is installed on a machine which is connected to a web server. They communicate with each other via XML-RPC. On requests of APP clients appropriate commands for VM creation, VM updation or VM deletion will be sent to XCP via web server. XCP now handles these requests through its own APIs. Using Xen Management APIs, we can remotely configuring and controlling virtualized guests running on a Xen-enabled host. The API is presented as a set of Remote Procedure Calls (RPCs), with a wire format based upon XML-RPC. The API reference uses the terminology classes and objects.

Various RPCs are available for attributes, classes. The return value of an RPC call is an XML-RPC 'Struct' (structure). The first element of the struct is named Status; it contains a string value indicating whether the result of the call was a "Success" or a "Failure". If Status was set to Success then the Struct contains a second element named Value. Value contains the function's return value. In the case where Status is set to Failure then the struct contains a second element named 'ErrorDescription'. This contains an array of string values. The first element of the array is an error code; the remainder of the array is strings representing error parameters relating to that code. XCP is managed through various classes like session, VM, VM_metrics, etc which provides its users with perfect platform for creating and handling virtual machines.

The important classes of XCP are arranged hierarchically in the following diagram succeeded by their short explanation.

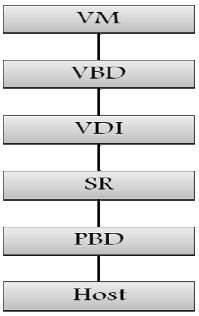


Fig. 3 Hierarchy of XCP classes

VM: A VM object represents a particular virtual machine instance on a Xen Cloud Platform Host.

VBD: A VBD (Virtual Block Device) object represents an attachment between a VM and a VDI. When a VM is booted its VBD objects are queried to determine which disk images (VDIs) should be attached.

VDI: A VDI object represents a Virtual Disk Image. Virtual Disk Images can be attached to VMs, in which case a block device appears inside the VM through which the bits encapsulated by the Virtual Disk Image can be read and written

SR: An SR (Storage Repository) aggregates a collection of VDIs & encapsulates the properties of physical storage on which the VDIs' bits reside.

PBD: A PBD (Physical Block Device) object represents an attachment between a Host and a SR (Storage Repository) object.

Host: A host object represents a physical host in a Xen Cloud Platform pool.

The open source community is actively developing solutions to manage clouds, especially those employed in academic research around the world. Some of the major open source cloud computing solutions are XCP, Eucalyptus, OpenNebula. Building a cloud environment for research purposes often involves choosing initially a solution for cloud management. This is a very difficult decision since each solution has specific characteristics adapted towards given scenarios. Suppose we have a homogeneous pool of Xen or KVM hypervisors and want to offer a public cloud service to a community, more specifically offering a simple IaaS. In this case, the free version of Eucalyptus will fit very well, though it is not able to provide some important services like VM migration. Now, consider that we have a pool of resources, possibly with heterogeneous virtualization platforms, and want to create a private/hybrid cloud over it. OpenNebula can easily support this. Also, if we want to provide IaaS with more diversified and powerful functionalities, as virtual machine migration and a more useful resource allocation mechanism then OpenNebula can also provide this. Overall, OpenNebula seems to be more suited for research and experimental studies. Now consider another scenario where we just want an easy way to manage a virtualized pool of resources and we do not have any concern regarding providing an external service to a

third party. In this case, a best approach would be to use XCP. In "Auto Provisioning Portal" we are required to set up a small scale cloud platform that is easy to manage and so XCP is efficiently used.

To summarize, XCP is an open source infrastructure manager tool for clouds that does not provide the overall architecture for cloud computing, since it does not provide interfaces to end users to interact with the cloud. However XCP provides a useful environment for administrators and an API for developers of cloud management systems.

## III. AUTO PROVISIONING PORTAL

In VAMP, Auto Provisioning, the portal running on web server can be accessed by clients anytime from anywhere. It not only provides the GUI but also simplifies and speeds up the process of interacting with the system.

The overall scenario of the project can be depicted easily through the following picture.

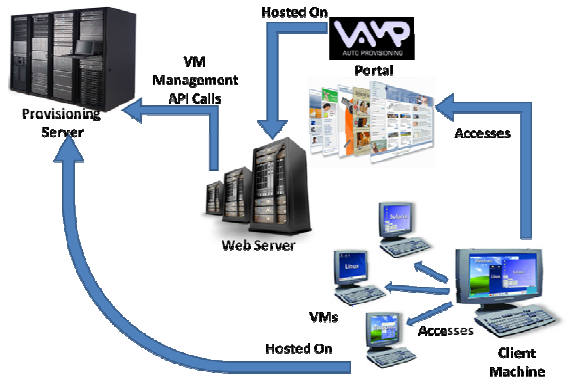


Fig. 4 High-Level representation of the system

VAMP APP provides various services to clients such as requesting for VM with configurations given in the specification catalog, view the request status, view bills, and terminate/extend VM. Administrator approves the requests and upon approval, provisioning of the requested resources is done.

When the clients want a virtual machine, they submit a request indicating the amount of memory, operating system and start and end date of the virtual machine. As in every other processing environment, the request passes through various stages which are visible to the clients. The first stage is to validate the request, i.e. to make sure the resources requested by clients are available for the duration and can be allocated to them. Once this is established, the request later goes for administrator's approval where he can either accept or decline request. Clients now make payments for the request.

### A. *Availability*

The requests have to be validated against the resources available with the system which briefly defines availability. The records of all requests granted by the system are maintained for checking availability. These records are kept in the form of a linked list where the date and amount of memory at hand are stored. Every time a request comes, the dates are mapped and memory is checked for validity. This availability is checked every time a new request comes, and whenever there is any unprecedented termination of requests or virtual machines. This strategy makes sure that the moment resources are free, they can be allocated accordingly. Every time a new request is accepted (original or extension) or deleted the availability list and chart is accordingly updated.

The availability list is used to obtain an availability chart where memory and operating system instances available with system are shown against a timeline. This feature of availability chart is shown to clients and assists them in submitting a request by referring to the capacity of system and ensuring that the request will be granted. A brief depiction of availability link and chart is shown in following diagram.

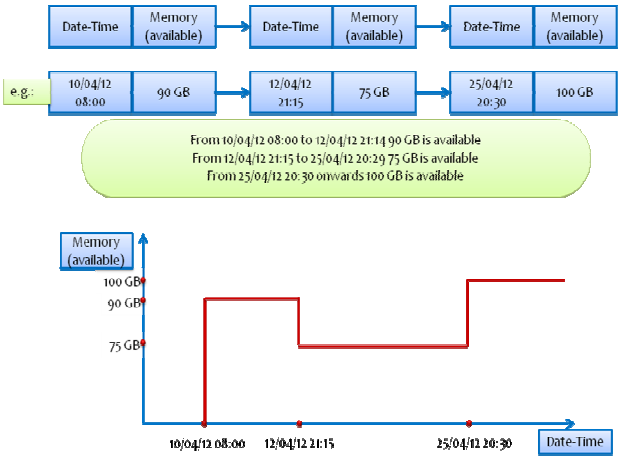


Fig. 5 Availability List and Availability Chart

### B. *Virtual Machine Creation*

The XCP provides in built templates for the operating system that it supports using which we create skeletons of virtual machines. The operating system now has to be installed using a media like a bootable device or a network repository. Certain settings have to be tweaked to allow clients to use these virtual machines remotely. The image of virtual machines with minimal pre-attached memory is then available with system. These steps need to be done manually by the administrator. For any operating system that APP wishes to offer, these steps need to be carried out first.

When a fresh request for virtual machines arrives, the image of the operating system requested is cloned and the disk volume is resized to that demanded by clients. An IP Address is allocated to this machine and a complete virtual machine is ready to be given to the clients.

### C. *Other Services of VAMP APP*

Functionalities like extending the virtual machine by stretching the end time or asking for additional memory is supported. When asking for time extensions, the availability list is referred to check if the same resources as that of the

original virtual machine can be extended till the new end date. If resources are available, the time extension request will be granted and availability list updated accordingly. When clients ask for memory extension, verification is done whether specified chunk of memory is available for given duration. If the system shows scope for granting extra memory, then a memory piece of size applied for is attached to the original memory as the start time of extension comes and that piece is removed after the end time on memory extension.

The diagrams show how processing happens during memory extension and deletion.

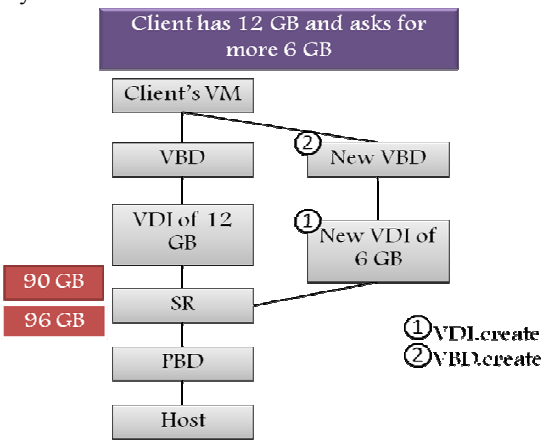


Fig. 6 Memory Extension

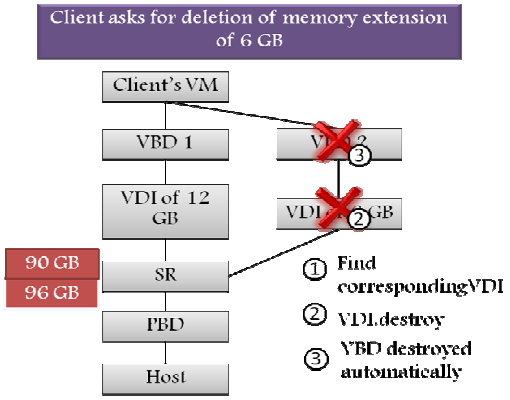


Fig. 7 Memory Extension Deletion

The clients are sent notifications about all highlight stages. Clients can query administrator with any failures encountered and view the replies of problems by admin.

## IV. TECHNOLOGY

This various technologies used in the system are XCP, XML-RPC, Apache Tomcat, Struts 2, Quartz Scheduler & Hibernate. The following figure gives an idea about the way technologies are distributed across the system.


Fig. 8 Distribution of Technologies

### *A. Apache XML-RPC*

Apache XML-RPC is software that enables us to create a server that uses Java to process XML-RPC requests and produce XML-RPC responses as well as to create a client that uses Java to call procedures on that server which are ultimately converted into XML-RPC requests. Apache XML-RPC consists of three basic parts: Apache XML-RPC data model, Apache XML-RPC server, Apache XML-RPC client.

Our system basically consists of a Cloud Platform provided by Xen and a portal developed using Java and Struts. The Xen Cloud Platform enables virtualization and provides various cloud related features whereas the portal acts as an interface where the client can avail to the cloud services provided by the cloud platform. As a result, there needs to be some mechanism by which the Xen cloud platform and the portal can communicate.

The Xen Cloud Platform does provide a Xen Management API which is an interface for remotely configuring and controlling virtualised guests running on a Xen-enabled host. The Xen Cloud Platform further provides language bindings which exposes the individual API calls as first-class functions in the target language. These functions can directly be used in the third party applications (that wants the services of the Cloud Platform) thus greatly simplifying the task of developing those applications. The Xen Cloud Platform provides language bindings for C, C# and python programming languages but the portal (i.e. the third party application in our case) is to be developed in Java. As a result, using these bindings is not an option for the system.

The Xen Management API can also be presented as a set of Remote Procedure Calls with XML-RPC as the wire protocol. In other words, the various functions and methods included in the Xen Management API can be remotely called from the portal using the XML-RPC protocol. The portal being in Java, the system requires a java implementation of the XML-RPC protocol for the Xen Cloud Platform and the portal to communicate. Here comes the use of Apache XML-RPC which has an additional advantage of being open source.

The Xen Cloud Platform will act as a Apache XML-RPC server and the portal will act as a Apache XML-RPC client. The client uses it's execute method to remotely call any of the procedures included under the Xen Management API. The execute method has two arguments namely the procedure to be called and the parameters required by the corresponding procedure.

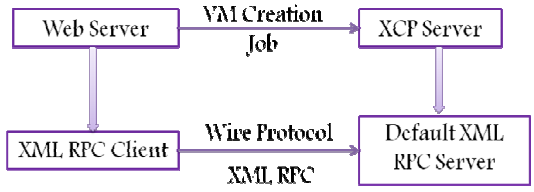


Fig. 9 Communication using XML-RPC

### B. *Apache Tomcat*

Apache Tomcat is a web container which allows running servlet and Java Server Pages based web applications. Most of the modern Java web frameworks are based on servlets and Java Server Pages and can run on Apache Tomcat, e.g. Java Server Faces, Struts and Spring. Apache Tomcat also provides by default a HTTP connector on port 8080 where it can listens for incoming requests. Installing Apache Tomcat on a personal computer turns it into a web server and can do any task that a server does. Our portal is deployed in the container of Tomcat, receiving, processing and replying the http requests with proper responses.

### C. *Struts 2*

The Struts-2 framework is designed for the compilation of the entire web application development cycle including building, developing and maintaining the whole application. Struts is mainly a presentation layer framework like redirecting to a particular page, doing client side validations etc which is otherwise very tedious using just JSPs and servlets. Struts 2 is an action framework where in essence it gives the ability to map URLs to activities and code on the back end. Our system uses Struts 2 platform to generate the user interaction pages, define linkages across pages and action classes calling methods containing our business logic.

### D. *Hibernate*

Hibernate is an object/relational mapping tool for Java. Hibernate is used to develop persistent classes following common Java idiom - including association, inheritance, polymorphism, composition and the Java collections framework. Hibernate not only takes care of the mapping from Java classes to database tables (and from Java data types to SQL data types), but also provides data query and retrieval facilities and can significantly reduce development time otherwise spent with manual data handling in SQL and JDBC.

### E. *Quartz Schedular*

Quartz is a full-featured Enterprise Job Scheduler that can be incorporated in any Java EE or Java SE application. Quartz can be used to create simple or complex schedules for executing tens-of-thousands of jobs. Jobs are tasks that are defined as standard Java components & can execute almost anything they are programmed to do. Quartz is an ideal solution for an application that has tasks that need to occur at given moments in time, or in a recurring fashion.

Our project has number of tasks that have to be done in the background at a scheduled time like creating a VM at a given point in time, deleting a VM at the estimated time, getting and updating the usage statistics regularly after specific interval of time for doing which the Quartz scheduler would be used. It provides various types of triggers such as simple trigger and cron trigger which are capable of scheduling tasks at any given time.

## V. STRENGTHS AND WEAKNESS

### A. *Strengths*

*1)* The clients are shown an availability chart from where they can decide upon the specifications of the request.
*2)* The clients can request in advance for VM.
*3)* The client can ask for time extension and up/down memory scaling.
*4)* The clients can even cancel any kind of requests or contracts for which he would be refunded money.

### B. *Weakness*

1) Windows and Linux OS are currently offered though other OSs supported by Xen can also be offered.
2) Resource pooling is not currently supported.

## VI. CONCLUSION

In this paper we have outlined all the facets of the system “Auto Provisioning Portal” and given a step by step description of the concepts, technologies and strategies that build and make the system functional.

## VII. FUTURE SCOPE

The virtual machines that are provided to the clients are bare minimal with no softwares installed on it, so if the clients want to run any specific program they will have to configure the environment accordingly. We propose to install the softwares that clients ask for and configure the VMs according to their specifications and provide them. We now provide static up and down memory scaling but in future we wish scale the memory up/ down dynamically. We will monitor the memory usages of the clients and if it appears that they might require more memory it will be made available to them or if they are not using a large part of memory allocated to them, it will be taken away and used elsewhere. At the moment we provision only the memory and operating system but network bandwidth and CPU can also be provisioned in the coming days.

## ACKNOWLEDGEMENT

We express our gratitude to our mentors who have kindly been directing our wandering footsteps in the right direction. The references included provided a foundation of knowledge on account of which we were able to achieve this target today and present a paper.

REFERENCES

[1] Chunye Gong, Jie Liu, Qiang Zhang, Haitao Chen, Zhenghu Gong. "The Characteristics of Cloud Computing". In 39th International Conference on Parallel Processing Workshops (2010).

[2] Johannes Kirschnick, Jose M. Alcaraz Calero, Lawrence Wilcock and Nigel Edwards, Hewlett Packard Laboratories. "Toward an Architecture for the Automated Provisioning of Cloud Services". In IEEE Communications Magazine (December 2010).

[3] Thiago Cordeiro, Douglas Damalio, Nadilma Pereira, Patricia Endo, André Palhares, Glauco Gonçalves, Djamel Sadok, Judith Kelner, Bob Melander, Victor Souza, Jan-Erik Mångs. "Open Source Cloud Computing Platforms". In Ninth International Conference on Grid and Cloud Computing (2010).

[4] Vmware. "Virtualization overview".

[5] Vmware. "Understanding Full virtualization, paravirtualization and Hardware Assist"

[6] http://www.xen.org/

[7] http://ws.apache.org/xmlrpc/

[8] http://www.tutorialspoint.com/xml-rpc/xml_rpc_intro.htm

[9]http://www.allapplabs.com/hibernate/introduction_to_hibernate.htm

[10] http://www.quartz-scheduler.org/