Cloud Based Remote Infrastructure Management

Ananthakrishnan Ravi, Roopak Venkatakrishnan, and Baskar Kannan

***Abstract*—As more companies shift to computerization the dependence of an organization on its system administrators is huge. They are required to manage not only the simple mundane tasks but also any other problems the company may face. This paper describes a unique approach to solving the simple, mundane yet time consuming tasks. Setting up Operating systems in a new work/lab environment is a daunting task. Similarly, working on company project from a remote location has been a persisting problem due to data security and privacy. Operating Systems can be set up over the internet using a kickstart server over the cloud and FUSE data transfer mechanisms. Virtual Images of current scenario company projects are booted up for use on demand, saved after work is done and deleted automatically.**

***Index Terms*—Cloud storage, centralized repository kickstart, automation, FUSE, virtualization**

I. Introduction

In cloud computing the way data is looked upon is different, everything is stored and accessed from the cloud and the differences between the systems are not known. The systems that access the cloud do not know from where the data is being accessed rather it's like hiding the complexity the IT assets. In this approach the entire kickstart server is moved to the cloud and options are provided for regular updating and the systems are invoked once updates are available and they are updated in turn. The centralized repository is updated using create-repo package in Linux. A package manager is used to latest upgrade.

When using a cloud environment the location of the data is dynamic and varies constantly. Although this is true for a user of the system this remains hidden and he perceives the data being located in his account all the time. The movement of data between different locations and servers is invisible to the user.

The main reason why cloud storage is gaining popularity is because of the financial and security based advantages.

Since data is stored on many machines (i.e. multiple copies) the chances of losing data is very low. Further the fact that each user pays only for what he/she uses ensures to make it cost effective

A web infrastructure is developed to view the progress of the machines that participate in the kickstart installations. Using the web infrastructure the entire nodes that are connected or rather using the services can be monitored. By using the same web infrastructure the clients can request for packages by accessing the centralized repository where the entire store of packages for all the distributions are stored. This makes updating and upgrading of software more easily. We make use of a package manager and a service look up table that notifies the user as and when a package is registered with the repository.

Middleware is a software application that allows various processes and applications in the host and clients to be contacted via a common channel or medium. This provides centralized processing of jobs that are to be transferred between machines which are heterogeneous and have trouble communicating among them. It provides interoperability. It is used to support complex distributed systems. It comprises of web servers, application servers. Services such as application development are also provided. Middleware is being popularly used in modern day architectures like Cloud Computing, Grid Computing and in Service Oriented Architecture.

Middleware lies between the application software and the various operating systems. As stated above they are used to provide and handle interoperability issues.. It is analogous to the central layer of athree-tier system architecture, conversely that it is geographically distributed. Some currently used citations are message queuing systems.

In actuality there is no defined difference between the OS and middleware. Usually the main kernel functionality is managed and provided only by the OS, further certain functions which were unique to separately sold middleware is now integrated with the Operating Systems themselves. An example for this is TCP/IP stack for communication, which is now a part of almost any OS.

In practical applications and simulations, middleware is considered high level architecture. This layer is sandwiched between the application code and the run time infrastructure. Middleware has a huge number of library functions which allows the many different applications to use these functions from the library than have them coded again.

File system in User space (FUSE) is a module that is loaded on demand in the Unix [1] operating system environment. The distinction is that it lets non-privileged users create their own file systems without editing kernel code. This is done by running a duplicate file system in user space while the FUSE module provides only a "bridge" to the actual kernel interfaces.

In order to facilitate faster, more efficient and less intensive data access and transfer what is known as a CDN (Content Delivery Network) is used. In this method the client is directed to the closest server containing the data it requires. This way one central server is not overloaded with the enter set of clients. Also since each client is accessing a server closest to it the speeds are much higher

Manuscript received August 20, 2012; revised October 10, 2012.

A. Ravi is with the Madras Institute of Technology, Chennai, India (e-mail: Ananthakrishnan.ravi@gmail.com).

R. Venkathennai is with the India akrishnan is with Sri Sairam Engineering College (affiliated to Anna University), Chennai, India (e-mail: roopak.v@gmail.com).

B. Kannan is with the Nixbase Technology Chennai, India (e-mail: baskar910@gmail.com).

We get into Cross platform Kickstart [2], [3] where a kickstart for a set of Linux distributions are available say Red Hat, Fedora, Debian and Ubuntu. By having a cross platform kickstart the users demand can be satisfied based on what distribution they are satisfied and which distribution would satisfy their purpose of using the particular operating system.

Hardware virtualization is similar to creating virtual machines in a system that act or simulate an operating system. Each virtual machine simulates various components present in the machine Software submitted to these virtual machines are distinct from the hardware resources utilized.

Hardware virtualization includes terminologies, the host machine is the machine in which virtual machines are created and these virtual machines are called guests or guest machines. The terms guests and hosts are comparative in nature when we refer to the virtual machines. The software or firmware that builds a virtual machine on the host is called Hypervisor or Virtual Machine Monitor. Some examples include Xen, VMware.

II. Related Work

The job of system and cluster administrators is getting difficult day by day having to manage a large number of machines simultaneously. To reduce the stress on them we go for the concept of kickstart in which the time required to deploy several machines in production as well as in administration is largely reduced. Cloud computing helps on a large extent to serve remote requests. Apart from serving administrators this paper concentrates on the need of automation for setting up labs and offices in a jiffy without human intervention. The process of remote management of computers as well as virtual machines is concentrated in the forthcoming divisions. Apart from performing kickstart installations the virtual machines for work from home feature are also discussed where the concept of data security and privacy comes into the picture.

III. Proposed System

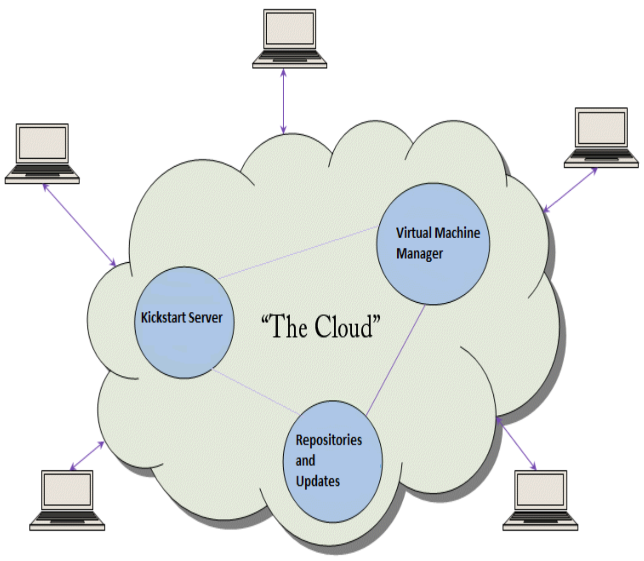

Fig. 1. Sample of proposed cloud layout.

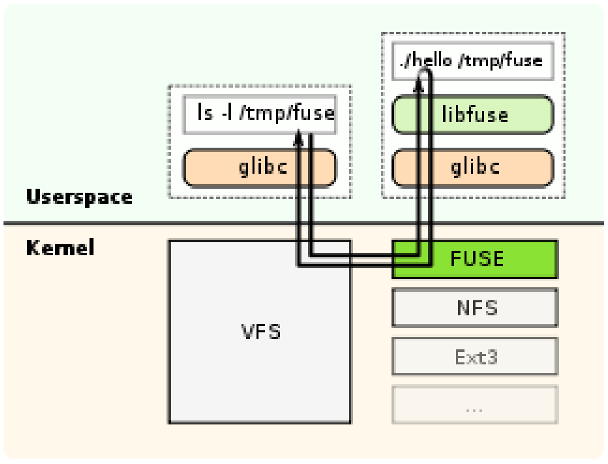

Fig. 2. FUSE userspace – kernel interaction.

A. *Permanent Kickstart Installation*

The permanent kickstart installation is similar to the regular installation of operating systems in computers. The cloud is used as a mechanism to enhance this feature. The kickstart is performed via HTTP which makes it even faster. It is conventional that HTTP transfer is quicker than FTP and NFS as it capable of withstanding greater load than the former two protocols.

A Kickstart installation follows the sequence as stated below:

1) The BIOS configuration is changed to enable Network Boot (generally F12 hotkey is used).
2) The nodes to be serviced by the kickstart server are booted over a network using PXE [4] and the network protocols Dynamic Host Configuration Protocol and Trivial File Transfer Protocol are configured correspondingly to suit the need.
3) The Kickstart file is downloaded from the kickstart server.
4) An Anaconda installation is automatically launched and prompts the user to select the desired installation which can be automated though by configuring the kickstart file to select default installation then reads the Kickstart file for the location of the Installation Tree. The tree generally resides on the kickstart server.
5) After accessing the Installation Tree [5], the installer attempts an unattended installation. If all the required information is unavailable from the Kickstart file or the file is configured incorrectly, the installer may prompt the user for additional information.

Another enhancement is the ability of the kickstart server to perform cross installations of operating systems like Ubuntu, Debian, Fedora, and Red Hat according to the request of the clients.

The kickstart installation in the cloud varies from the traditional kickstart install. A middleware is used to communicate with the hardware at boot up. Once the URL is accessed the installation starts and completes in accordance to the configuration requested by the user/client.

Other mechanisms are the same as the traditional kickstart. Only the inclusion of the middleware and the use of a CDN for repositories vary.

In order to provide faster updating and redundant presence of packages we implement a Content Delivery Network which greatly enhances the installation of packages and updates. In the absence of the kickstart server the CDN can act as a redundant measure which helps in the scenario of a fail over in the cloud.

Further FUSE with its own extensions can further improve the performance in installations which would be assessed in the forthcoming sections.

To capitalise on performance measures we can implement a multi-level kickstart server [6], [7] after one of the servers in the network have been configured as a kickstart server by using the services present on the cloud.

B. Temporary Kickstart Installation for Virtual Machine Management

The companies provide work from feature for many employees but that aspect proves to be a dangerous issue, sensitive data of the company may be leaked as the access rights of the person is not curbed unlike in office environment. In this case we go for a mechanism through which virtual machines are loaded by the kickstart server which provides on demand frameworks and upon completion data is saved on to the cloud and the data present in the employee's machine is deleted. The provision of on-demand frameworks is provided with the help of a cloud of service in which a remote plugin is used to load the framework on to the employee's machine from the browser used by the person. A person can even work with multiple frameworks if he wants as some other tools may be wanted and networking features between the virtual machines is possible.

This provides data privacy and security of the data handled an further companies can rely on the work given to their employee's by using this concept.

IV. Implementation

A. Centralized Repository Using Content Delivery Network

For implementation of a centralized repository we use the concept of content delivery network through which the repository nearest to the location of request is serviced thus enhancing the efficiency and performance.

Content types include web objects, downloadable objects which can be in the form of text documents as well as applications, real time media streams, and other components such as networking information. Domain naming service and routing information are examples. .Web caches store popular content on servers that have the greatest demand for the content requested. These shared network appliances reduce bandwidth requirements, reduce server load, and improve the client response times for content stored in the cache.

Server-load balancing can be implemented using software and hardware measures. Software load balancing is called global load balancing. Hardware load balancing includes the use of higher level switches i.e. layer 4–7 switches which is used to share traffic among a number of servers or web caches. Switch is assigned with a unique virtual IP address. Traffic arriving at the switch is then directed to one of the real web servers attached to the switch. This has the advantage of balancing load, increasing total capacity, improving scalability, and providing increased reliability by redistributing the load of a failed web server and providing server health checks.

A content cluster [8] can be assembled using a higher level switches which aid in balancing the load across a number of servers or a number of web caches within the network.

B. Httpfs (HTTP Using FUSE)

Httpfs is used for the following reasons:

- The server determines the file size with the help of a HEAD-Request, and in turn of downloading the whole file to get to know the entire size.
- loading arbitrary parts of the file with a Range-Header in the GET-Request

All other tasks are done by FUSE. This yields a read only file system.

For this to work, the server must only comply with the HTTP/1.1-standard. Therefore httpfs can be used with (nearly) arbitrary web servers.

C. P2Pfs

p2p-fs [9] is read-only peer-to-peer file-system. What this means is that peers can perform all read-only actions on files that are exported by their peers to them. The peer-to-peer part means that all nodes in the system are equal, and none has a higher preference/authority than any other. That apart, there are some other features like redundancy, fault tolerance and load balancing that come as a part of being a peer-to-peer system p2p-fs was developed to be used in a LAN, where all systems are connected to each other, and users want to be able to share files with each other without having to copy them to their disks; thus saving a lot of disk space. We have successfully shared over 40GB of data in our LAN very comfortably. Suppose this 40GB was to be copied by each client, then even 10 clients would be wasting about 400GB of disk space. p2p-fs is also able to stream video and audio files comfortably.

A kickstart environment involves the transfer of large files which can be greatly enhanced by using p2p fuse, thereby ensuring that machines with lower computation power can efficiently use the files thereby increasing installation speed. In a distributed environment we can use p2p efficiently by using the concept of CDN.

D. Virtual Image Distribution

In a work environment where data security and privacy is of utmost importance, employees are rarely allowed to work outside the office campus. This is to prevent data leakage etc. We propose a system where a virtual image is stored on a deployable cloud server. When an employee wants to work from a remote location, the image is loaded onto the employee's system and all changes are stored in a settings file. On closing the virtual machine, the image is destroyed and the settings file containing all the changes made is stored back on the server.

The client system has an image manipulation software which can load the image files deployed from the server (Loading on demand). This software generates a settings file which stores the changes and work done on the image. When done working the settings file is saved back on the server. The image is then deleted.

V. Analysis

Consider a network with n requests of a kickstart install. Let op be the optimum machines that are to be serviced by the root kickstart server and let k be the time to perform a kickstart installation.

Time Taken = $n/op*k$

In the context of a cloud kickstart server time taken to set up a kickstart server is just slightly greater than installation on a node. So the time for n nodes can be computed by using a recursive function as follows:

Algorithm:

```
timefn (nodes_left)
{
    convert (nodes left/op) number of finished nodes to subserver
    service left over nodes
    nodes_left=new number of nodes left
    timefn(nodes_left)
}
```

By using the above algorithm the number of server conversions is lesser when compared to individual installs via independent requests and hence providing a kickstart install via HTTP serves it purpose.

VI. Advantages

A. Performance

The basic intuition for going for a cloud level kickstart server is performance. It can service a larger number of nodes with less latency, faster installation and better execution of tasks.

B. Installation of Applications and a Centralised Repository

The kickstart server has all its applications and packages installed in all the kickstart servers. Hence any required packages are obtained without any human intervention.

C. Use of Resources

The entire kickstart environment balances the load among itself by designating n machines to each server hence there is efficient use of resources and network bandwidth.

D. Reduced Stress on System Administrators

By providing kickstart as a cloud service we can reduce the stress on administrators as well as provide remote setting up of labs with ease.

VII. Conclusion

To further improve the provisions of the cloud it is possible to enhance the performance of virtual machines by enhancing the corresponding hypervisors. Using multiple servers further reduces the load on the kickstart server present in the cloud and this can be overcome by using simple load balancing servers and the idle servers can be converted into web servers which can serve a different purpose. Again an analysis of the protocols can be done and implemented as which ever may suit its application. We can further work on the web interface provided by the cloud to monitor the installation in each machine.

VIII. Future Work

One idea is that all the network parameters are not the same hence we can go for a new environment to implement this say a private cloud which can provide a kickstart service as a cloud service through a web interface and providing resource management and progress in each ,machine. Another idea is to work with each and every protocol the kickstart supports i.e. HTTP, FTP, NFS. Surely there would me a marginal difference in the working as well as the performance metrics.

References

[1] D. Liu and P. Bai, “Using the user space file system to protect file,” *Presented at the Apperceiving Computing and Intelligence Analysis (ICACIA), International Conference*, 2010.

[2] S. M. Diesburg, P. A. Gray, and D. Joiner, “High performance computing environments without the fuss: the Bootable Cluster CD,” *Presented at IPDPSI*, 2005.

[3] F. Yi, Y. Luo, X. Wang, Z. Wang, B. Zhang, H. Chen, and X. Li, “Fast live cloning of virtual machine based on Xen,” *Presented at High Performance Computing and Communications*, 2009

[4] T. Cruz, P. Simoes, F. Bastos, and E. Monteiro, “Integration of PXE-based desktop solutions into broadband access networks,” *Presented at the 6th International Conference on Network and Services Management (CNSM)*, October 2010.

[5] W. Jin, L. Teng, and L. Feng, “Real-time net-booting System in large-scale DSP network,” *Presented at the IEEE Radar Conference*, 2006.

[6] H. Zhang, K. Wu, J. Li, B. Zhang, Z. Xue, and B. Yan, “Parallel file system-surpported server virtual environment in data center,” *Presented at the ICDNC Conference*. 2010

[7] D. Ramalingam, “Practicing computer hardware configuration and network installation in a virtual laboratory environment: A case study presented at the frontiers in education conference–global,” *Engineering: Knowledge without Borders, Opportunities without Passports*, 2007.

[8] J. Schiffman, T. Moyer, T. Jaeger, and P. McDaniel, “Network-based root of trust for installation,” *IEEE Security and Privacy*. vol. 9, no. 1, pp. 40-48, Feb. 2011.

[9] B. Amann, B. Elser, Y. Houri, and T. Fuhrmann, “IgorFS: A distributed P2P file system,” *Presented at the Peer to Peer Computing*, 2008.

Ananthakrishnan Ravi has completed his Bachelors in Technology specializing in Information Technology at the Madras Institute of Technology, Anna University, Chennai, India. He has worked on Cross user level de-duplication that aims at storing data efficiently on the cloud by using distributed soft links as a part of his undergraduate thesis. The project aims at storing data across various servers and referencing those using links so that data can be accessed from various parts based on CDN. Further he is a Red Hat Certified Engineer (RHCE). He will be pursuing his Masters in Computer Science (Networked Systems) at the University of California, Irvine starting September 2012.

Roopak Venkatakrishnan has completed his bachelors in engineering specializing in computer science at Sri Sairam Engineering College, affiliated to Anna University Chennai, India. He worked on a project on data analysis at Alpha Cloud labs as Intern in Chennai during the period of December 2011 to march 2012. This project aimed at data extraction and analysis for the purpose of data summarization. He will be pursuing his masters in computer science at North Carolina State University beginning August 2012.

Baskar Kannan is currently the Technical Director at Nixbase Technology based in Chennai, India. He is engaged in providing open source solutions to Small and Medium Enterprises (SME). He also conducts workshops and sessions based on open source products and technologies. He is a part of the Indian Linux Users Group (ILUGC). He specialized in Single Sign-On technologies. He is a Zend Certified PHP Programmer an d a Red Hat Certified Engineer (RHCE).