Multi-Level Kick Start Server

Ananthakrishnan Ravi and Roopak Venkatakrishnan

***Abstract*—To set up an operating system on hundreds of machines is a daunting task. To automate this process the concept of a kickstart server may be used. While this system has its merits it fails in a large scale environment where high performance and availability is required. This paper focuses on utilizing the available resources to optimize the efficiency of a kickstart server in a network. The current kick start server employs one system which acts as the master node and sets up an Operating System and applications in a network of systems. When the scalability increases the performance of the master node drops exponentially. A multi-level kick start server helps overcome these issues. Here one server creates more servers and controls them, thereby reducing the load on the main server and also improving speed and efficiency.**

***Index Terms*—Kick start server, multi-level, sub-server.**

I. Introduction

Kickstart is a system for automated installation of Red Hat Linux or Fedora Core Linux. Instead of answering all the installation questions manually, one can put configuration information and packages selection into a file which is read and executed by the installation program. This page provides general information, links and some tools for creating or manipulating kickstart configuration files and boot images.

The process of installing a Linux operating system in several machines is easily done using what is known as a Network install, by adopting Anaconda the installer for Red Hat Enterprise Linux. Kickstart servers provide automated installation of the OS as well setting up of applications and various other packages. It may also be used to perform updates of various packages on an entire network of systems. The Rescue mode can be used to troubleshoot a system that is no longer bootable. It loads the necessary rescue environment from the second stage unit [1] and gives the user a shell to access a system. There are two stages in a Linux boot loader. The first stage boots the system and performs initialization of the system it then loads the second stage from the specified installation site.

A Kickstart file contains a series of options, to be passed to the Anaconda installer that describes how to set up the system. It may also include custom scripts to be run before or after the installation.

A Kickstart installation in Linux comprises of a sequence of steps 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 [2] 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, 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

In the multi-level Kick Start server, the main server sets up (say) n nodes. Each of these further propagates and sets up n nodes each. This reduces the load on the master and improves the performance and also assists in case of a failure.

The kick start server system described in this document provides a more efficient method to set up an entire network of systems.

II. Related Work

Network-based Kickstart [3] installations are more difficult to implement than the floppy-based ones. However, network-based Kickstart installations offer advantages. Once configured, a network-based system offers better automation features than the floppy method. Rather than building different floppies for different installation options, the process is made easier when using a network.

A BOOTP or DHCP server is needed on a private network to perform a network-based Kickstart installation. The BOOTP or DHCP server provides the installation client with its IP address so that it can communicate on the network; network connectivity is, of course, essential if the computer is to receive the Kickstart configuration.

Kickstart servers are deployed when multiple machines are to be fitted with a homogenous operating system. The clients are booted via network boot. In a kickstart server a client node requests the root node for an IP address to communicate with it, the root node which runs a dhcp server allocates a unique IP address to each of the nodes that request for a kickstart build. Once an IP address is allocated a kickstart file which is located in the server is sent to all the nodes and the configuration [4],[5] of the system to be built is specified in the kickstart file. The node gets all the files required for the build and parses the kickstart file and configurations

Manuscript received September 14, 2012; revised November 30, 2012.

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

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

specified are applied. The operating system is installed along with the necessary packages. A centralized repository of the kernel and the packages is maintained in the root node which is passed on the requesting nodes.

Example Kickstart Configuration

```
#platform=x86, AMD64, or Intel EM64T
# System authorization information
auth  --useshadow  --enablemd5
# System bootloader configuration
bootloader --location=mbr
# Partition clearing information
clearpart --none
# Use text mode install
text
# Firewall configuration
firewall --enabled
# Run the Setup Agent on first boot
firstboot --disable
# System keyboard
keyboard us
# System language
lang en_US
# Installation logging level
logging --level=info
# Use NFS installation media
nfs --server=192.168.1.1 --dir=/var/ftp/pub/
# Network information
network --bootproto=dhcp --device=eth0 --onboot=on
# Reboot after installation
reboot
# SELinux configuration
selinux --disabled
# System timezone
timezone  Asia/Calcutta
# Install OS instead of upgrade
install
# X Window System configuration information
xconfig  --defaultdesktop=GNOME --depth=8
--resolution=800x600
# Disk partitioning information
part / --bytes-per-inode=4096 --fstype="ext3" --size=5000
part /boot --bytes-per-inode=4096 --fstype="ext3"
--size=1000
part /home --bytes-per-inode=4096 --fstype="ext3"
--size=10000
part swap --bytes-per-inode=4096 --fstype="swap"
--size=2000
%packages
@office
@development-libs
@editors
@gnome-software-development
@text-internet
@x-software-development
```

The existing kickstart method of serving systems uses a single kickstart server. Other methods of automated install in windows include Windows Automated Install Kit (AIK) which does not offer methods similar to what linux based installs offer as the number of configurations that can be specified in kickstart are on the higher side with greater compatibility in machines.

The various phases in Windows AIK include:

Phase 1: Pre-installation Planning
Phase 2: Pre-installation Preparation
Phase 3: Pre-installation Customization
Phase 4: Image Deployment
Phase 5: Image Maintenance

Even though some tools may evolve that provide similar interfaces to the kickstart install via PXE .The traditional kickstart install still proves to be a great competitor as many optimizations are possible which can be configured depending on the environment.

The current trends in installation of operating systems in multiple machines apart from kickstart include cloning of virtual machines. Tools such as VMware Clone can be used to clone virtual machines on many nodes in a network hence this may provide simulation of similar machines in the network but has the disadvantage that all the machines used are only virtual machines and does not comprise of any physical installation.

A clone is a copy of an existing virtual machine. The existing virtual machine is called the parent of the clone. When the cloning operation is complete, the clone is a separate virtual machine — though it may share virtual disks with the parent virtual machine:

- Changes made to a clone do not affect the parent virtual machine. Changes made to the parent virtual machine do not appear in a clone.
- A clone's MAC address and UUID are different from those of the parent virtual machine.

III. Proposed System

In this environment the root kick start server sets up n nodes and also converts some of the nodes into kick start servers themselves. These nodes acquire a copy of the configuration files and settings to be implemented on the various other nodes. The process repeats iteratively till all the nodes are completely set up. This idea utilizes one master node which controls the various sub servers (a kick start server created by the root [6] kickstart server from an existing node) which are set up and assigns them with responsibilities.

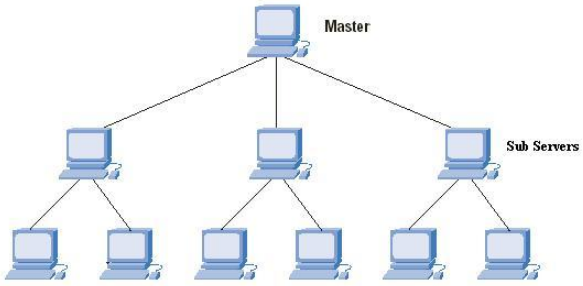

Fig. 1. Structure of network.

A. Management of Kickstart Servers

The root server maintains a status_table which it uses to monitor the scenario and progress of the entire process. This table is populated with information from the various sub servers and the nodes themselves.

When a node is set up depending upon the need it may be

converted to a kick start server. It is assigned nodes the sub server has to service. Each sub server maintains another table with its specified nodes and the progress. The table at the root is constantly in sync with the tables on all sub servers.

IV. Implementation

The process starts with a single root server which is configured by the administrator as the kickstart server by using packages dhcpd, xinetd, nfs and portmap along with ftp for file transfer. After the root is configured the process of multilevel kickstart begins where the root server selects nodes to be created as sub servers and once this process takes place certain numbers of sub servers are created based on the number of nodes that have requested for a kickstart build. Once adequate numbers of sub servers are built the nodes are serviced. Operating System and packages along with post installation scripts are installed in all the nodes.

A. Status_Table

The root server maintains a table with 3 attributes – IP address, type, and progress. The IP address maintains a list of the address of all nodes which the root node allocates at the beginning of the process. Type indicates whether the node is a sub kick start server or a regular node and progress maintains the level of completion on each node. Progress is set to -2 if the node is a sub server in use, -1 if it is a node yet to be touched and 0 to 100 for nodes allocated and in progress.

IP Addr.	Type	Progress
10.0.0.11	N	100
10.0.0.12	KS	-2
10.0.0.13	N	35
…		
10.0.0.83	N	0
…		
10.0.0.121	N	-1

Fig. 2. Sample of status_table.

B. Selection of Sub Servers

Sub servers are created only when there is a necessity and the number of sub servers is calculated on the basis of the nodes still unattended on the network. Consider a scenario with 100 machines to be set up. If a root server can service 10 systems at a time, when these 10 are completed 90 systems remain to be serviced. Including the root we require 9 servers. So 8 out of the 10 are converted into a sub server and are assigned 10 systems by the root. Conversion to a sub server involves copying all required resources and configuration files from the root or any other sub server. This methodology is used in all cases to find the required number of sub servers.

V. Analysis

Consider a network with n nodes. Let α be the optimum no of machines that can be serviced by the root kickstart server and let k be the time to perform a kickstart installation

Time Taken in case of a standalone kick start server

$$n / \alpha \times \tilde{k} \quad (1)$$

The analysis of the multi-level kickstart is to be done a large number of machines say (n>100) then only its application and the algorithm seem to be optimum and greatly efficient. The number of machines under each subserver may or may not be equal. Optimality of the number of machines is totally protocol dependent.

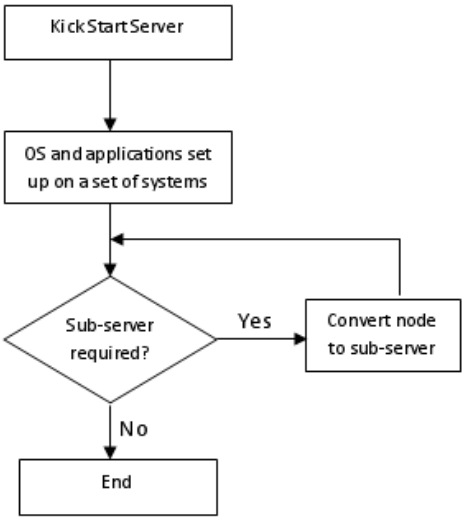

Fig. 3. Flow diagram of sub server creation.

In the context of a multilevel kickstart server time taken to convert a node to a kickstart server is much lesser than installation of 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 / α) number of finished nodes to
   subserver
  Service left over nodes using all subservers
  nodes_left=new number of nodes left
  timefn(nodes_left)
}
```

So the total time taken to complete the entire process is just time can be given by

$$k + timefn\ (n\text{-}\alpha) \quad (2)$$

This is based on the fact that n is very much greater than α and hence the multilevel kickstart server turns out to be more efficient.

VI. Advantages

A. Failover

The major advantage of a multilevel kickstart server is that can perform when a kickstart server fails. The system that was servicing comes under a kickstart server that has fewer nodes attached to it. In event of failure that kickstart server is removed out of the network and repaired until its return.

B. Performance

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

C. Installation of Applications and a Centralized 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.

D. 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.

E. Advantage of a HTTP Install

Kickstart Installations can be configured via an FTP, NFS or Apache server. Apache Server (HTTP) proves to be the better of the former two protocols due to its ability withstand heavier loads than NFS and FTP along with greater transfer rates while downloading files from the server.

Kickstart Server configured via HTTP protocol has the following merits:

- Often kickstart server has to be located on a remote network, often passing through a firewall. Firewall configurations for HTTP are easier to configure than for FTP and NFS.
- The http:// nomenclature is used by kickstart for accessing files is more familiar to users than that used for NFS and FTP. Configurations hence through HTTP seem to be easier as many find it easier to work with Apache.

F. Adjustment of Sub Servers on the Fly

According to the algorithm devised the number of sub servers is maintained to an optimum limit and hence there is no wastage of resources and the kickstart servers can be reverted to regular nodes easily.

G. Each Node can be Configured Based on Its Requirements

By using subnetting and network configurations the nodes can be configured based on the requirements and the corresponding kickstart configurator file is accessed and the build is performed.

VII. Conclusions

The multilevel kickstart server uses multiple servers to service nodes. This is similar to a DDOS attack in a hacking environment in which a number of servers are multiplied to increase the performance and also acts a mechanism to reduce failure of nodes. An analysis of the protocols used for kickstart process can be done to maximize the throughput of the entire environment. Apart from these parameters the entire process is thoroughly dependent on the hardware specifications used. The switches and the operating systems used for implementations also play a major role. The critical resources in this environment are the number nodes participating in the process. A post installation script can be used to perform a set of operations say create users and assign access and ACL lists without any intervention by the users of the systems. The pre installation script can be used to take back up in machines before the installation is started and this can be used to revert to the old settings. The post and pre installation tasks are invaluable parts that can be utilized in a kickstart environment which makes automation to greater heights. Total automation is possible using kickstart right from installation package dependencies to administrative tasks.

VIII. Future Work

One idea is to move the entire kickstart server to the cloud by doing which we could improvise on the resource management and obtain better performance metrics .Using a single web interface the machines can be monitored and managed .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.

Acknowledgment

A. Ravi thanks Mr. Baskar, a member of the Indian Linux users group who gave support while working on the paper.

References

[1] S. M. Diesburg, P. A. Gray, and D. Joiner, "High performance computing environments without the fuss: The bootable cluster CD," Presented at IPDPSI, 2005.

[2] 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.

[3] J. Wei, T. Long, and F. Liu, "Real-time net-booting system in large-scale DSP network," Presented at the IEEE Radar Conference, 2006.

[4] H. M. Zhang, K. C. Wu, J. H. Li, B. Zhang, Z. H. Xue, and B. P. Yan, "Parallel file system-surpported server virtual environment in data center," Presented at the ICDNC 2010 Conference. 2010.

[5] 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.

[6] 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, Jan.-Feb. 2011.

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.