How To Remotely Build and Manage a Linux Solution

(How Open Source Software Keeps Me From Driving to the CoLo at 3 AM)

by

gregory j pryzby
VALiNUX Systems

ABSTRACT

Linux is growing in popularity. Reason differ, but commodity hardware and open source software (OSS) are two reasons. Commodity hardware and OSS help drive the total cost of ownership (TCO) down. If systems can be managed remotely and more efficiently, the TCO drops again.

Using two OSS projects, VACM (http://vacm.sourceforge.net/) and SystemImager (http://systemimager.sourceforge.net/), systems can be remotely installed, updated, and power cycled.

Once the systems are initially installed, there is no requirement for physical access of the systems. All the changes and management required can be done over the Internet, securely–– even BIOS changes.

1. OVERVIEW

Keeping machines up and running is the most visible part of a job. When a machine is down, everyone knows it. Management and clients are standing by waiting for the systems to come back on−line. Every minute the system is down is lost revenue. Below is a solution that should minimize down time.

I work for VA Linux Systems (VALS) (NASDAQ: LNUX). We are committed to Open Source Software. What does this mean? It means that all software we develop at VALS is OSS. We meet the definition as defined the Open Source Initiative (http://www.opensource.org/). Typically the license is the GNU Public License (GPL).

VALS is committed to open standards. All pieces of a solution should be available from multiple sources. VALS encourages that our customers look at what we have to offer. We believe that our service and understanding of the problem and software used to solve the problem will lead you to VALS.

1a Open Source

Open Source Software (OSS) is the future. The reason is simply: no software does it all and the ability to tweak the software to meet the requirements is essential to success. OSS makes that possible.

Using OSS allows multiple people to review and enhance the software. Instead of spending time trying to solve a problem that someone else has solved, the developer can enhance a product that almost meets the requirements. Without OSS the developer would have to write all the code from scratch. If a small change or feature needs to be added, this is much quicker when OSS is used.

1b Sourceforge.net

VALS is committed to OSS. Sourceforge (http://sourceforge.net/) can help overcome the common problems of developing OSS. As a developer, bandwidth, space, and tools (mailing lists, cvs, ftp, bug tracking) are not easy to come by and typically cost money. If this service is provided by a company as part of employment, when employment is terminated, the service will most likely go away.

Sourceforge.net to the rescue! All of the tools for developing and maintaining software are available at no charge. The only requirement is that the software meets the definition of OSS. Once that is determined, you are on your way.

1c Problem

Now you know a little about OSS and Sourceforge.net. So what? Why is that important to this talk? Because OSS and Sourceforge.net are how VALS solve problems. The software VALS uses is OSS. The software is hosted on Sourceforge.net.

The problem I want to solve is to allow remote management of systems. This means installing a new system or software, updating software and making BIOS changes must be available at a minimum. Using OSS software, this is possible.

We will talk about how to use SystemImager (http://systemimager.sourceforge.net/) and VACM (http://vacm.sourceforge.net/) to accomplish this.

2. TOOLS

The tools I am going to discuss are located on Sourceforge.net (http://sourceforge.net/).

2a SystemImager

SystemImager (http://systemimager.sourceforge.net/) is the first tool we will examine. SystemImager is OSS. The purpose of SystemImager is to allow machines to be built quickly. It "images" a machine. That image is duplicated to any other machine quickly. Networking is the only section that is different between images. SystemImager uses syslinux, rsync and perl to load the image. SystemImager uses a pull technology. The Image–Server pulls the image from a client. A client pulls the image from the Image–Server on an install or update.

To use SystemImager, the hardware of the machines should be the same. We will assume that it is for our purposes. Please refer to the documentation and source code if the hardware of the clients differ. Also refer to the documentation for the exact commands and syntax.

You need at least two machines for SystemImager–– the client and the Image–Server. Linux is loaded onto the client. The client is configured. At this point, a script is run on the client to prepare it to be imaged. The Image–Server runs a script that grabs the image from the client. This script also configures the Image–Server to allow it to serve the image to other clients.

Any client can pull the image from the Image–Server. To pull the image, a client requires either: a boot disk, a boot CD, an image on the root partition, or network interface card that supports PXE boot. After the image has been pulled, the machine is ready to use.

An image can be update by either updating an image and running a script on the Image–Server to sync the images or by chroot–ing the image on the Image–Server and making changes to the image on the Image–Server.

2b VA Cluster Manager (VACM)

VACM (http://vacm.sourceforge.net/) is the next tool we will examine. VACM is OSS. The purpose of VACM is to remotely control machines. You can turn off power, turn on power, identify (make the power light on the front blink) and read sensors from the motherboard. You can also access the BIOS to make changes.

To use VACM you need a motherboard that has a Baseboard Manager Controller (BMC) and supports the Intelligent Power Management Interface (IPMI). This allows turning power off and on, blinking the power light on the front and getting sensor data from the motherboard. In the BIOS, the Emergency Management Port is turned on and the data is sent on COM2. This is streaming data that sends the information about the sensor data. This is also used to control the BMC and control power and identify.

To have serial access to the machine, console needs to be redirected to COM1. This allows a user to logon to the machine over the serial port. It also allows a user access to the BIOS.

You can use flim, a graphical user interface, to control the nodes. You can highlight a node and "click" on a button turn power off/on, start/stop the power light from blinking, read sensor data or open a window to logon to the node.

You can use vash, a command line interface, to access and control the nodes. The functionality is a superset of flim.

3. BRINGING IT TOGETHER

Now you know about the tools available. Let’s look at a problem and how to use the tools, SystemImager and VACM, to solve them.

3a Real World Example

You are responsible for a group of boxes. There are two types of boxes that you are managing. The first are web servers. They are configured in a cluster that is load balanced using a layer 4 switching solution (UltraMonkey, F5’s BigIP, etc.). The boxes are all the same. The second are back–end boxes. They might access a common data–store via NFS. Or they might keep a copy of data locally. We are only concerned with the web servers in this example. Handling the back–end boxes is similar to the web servers, just slightly more involved. However if you understand how to manage the web servers, the same methods apply to the back–end boxes.

3b What You Need

You need the machines that are the web servers. You will also need the load balancing machines. A switch and infrastructure is also required.
You also need one box that is used by SystemImager and VACM. This can be the same box. You will need a serial concentrator. This can be a separate device or an add on to the machine that VACM runs on (ie, RocketPort).

3c Physical Install

You will need to install the hardware. The assumption is that you are using a rack to hold your machines. If not, cable management will be more difficult, but everything discussed applies. You will rack the machines. You will wire the ethernet to a switch. You will wire the serial ports to a serial concentrator. Now you are ready to create your image.

3d Grab Initial Image

Pick a box, but NOT the box that will hold the image. Connect a keyboard, monitor and mouse. Turn on the system and login. Download the SystemImager RPMS and install the client RPMs. Download and install the client RPMs for VACM. Configure VACM (see documentation; don’t forget serial redirect). Check the software installed on the system. Make all changes required. Make sure the machine is configured properly. When you are satisfied, run the command to prepare the client. The client is now ready.

Now we are ready to work on the Image−Server. Connect a keyboard, monitor and mouse. Turn on the system and login. Setup the network to allow internet access. Download the SystemImager RPMS and install the server RPMs. Download and install the server RPMs for VACM. Configure VACM (see VACM documentation for specifics). Run the command to grab the image from the client.

Check the BIOS on each node. Console redirection and EMP needs to be turned on. Refer to the directions with VACM on how to do this.

The Image−Server is now ready. Before you leave, I recommend that you try VACM to make sure you don't need to make a trip to your servers. Reboot the client. Start VACM on the Master−Node. You should be able to see the client with the GUI or CLI. Try to connect to the client using the sercon module. If all works, you are done. You can go back to your home or office and work from there.

3e Install Image

Once you are comfortable at your work location, logon the Image−Server. Connect to VACM using the CLI or GUI. Open a console on the machine you want to install. I recommend using the hard disk install. Run the update client script with the −autoinstall option. Watch it go!

3f Recovering a Node

A node has died or become corrupted. It is time to use SystemImager to rebuild the node. Use VACM to connect to the machine over the serial port, sercon module. If the machine is not running, power cycle the machine with VACM. Watch the machine boot in the serial window. If the boot succeeds, check to see if the image is OK. If so, you are done. If not, you need to put a boot image in root (/). This is the image from SystemImager that is run on boot. Reboot the machine and it will be reinstalled.

If the machine does not boot, then using the sercon module to enter BIOS. Change the boot order to use PXE. Save the changes and reboot. The image should be grabbed and the machine will be reinstalled.

3g Updating an Image

At some point, you will need to make a change to your image. I find the easiest way to do this is to work on the image on the Image−Server.

Find where the image is on your Image−Server. Change to that directory. 'chroot' to that directory. This allows you to issue commands and the commands change the files of the image. So changes to /etc/lilo.conf and running /sbin/lilo update lilo for the image and not the Image−Server. If you use a package management tool (RPM or apt) the packages in the image are changed, not on the Image−Server.

After the changes are made, the client runs a script to sync with the Image−Server.

4. SUMMARY

That is all there is to it! Once you setup the systems and build the infrastructure you can hand it off. You can give it to the junior administrator and work on what you enjoy!

4a Highlights

Free software can make you life easier. And if you discover that it doesn't do everything, you can modify it!

4b Questions

Any questions?