Automation of Linux Installations

This document introduces software which can be used to automate Linux installs for executing mass installations. The software is explained and an overview of the installation and configuration process of each software is given.

1 Introduction

With the rapid adoption of Linux the need for easier and faster installation rises. But not only an easier installation process is requested. More important is the automation of the installation procedure. The desire for automated installation and configuration of Linux has different reasons:

- Simplification of the installation process
- Speed-up of installation process / economy of time
- Unattended installation
- Replication of (basis-) systems within companies / PAs
- Customisation of a distribution after installation
- Fast replacement of defect or compromised systems
- Creating clusters

For these tasks, many Open Source solutions already exist. But before starting to use one some general assumptions have to be considered. The fundamental part here is the distribution which is going to be used. Furthermore it is important to differentiate between the initial installation and the administration later on. It also has to be defined if the administration of all systems is to be distributed or from a single point.

To choose a tool it is crucial to know the used distribution. This is because of the dependency of the solutions towards the distributions. Most active available solutions on the market only support one distribution or have been developed and tested only for a specific one.

2 Software

2.1 Disk Cloning

Disk Cloning is not a special software solution. Disk Cloning uses UNIX-Tools like dd, dump or restore. With these commands a prototype from a base-system (golden client) is created which can

be restored and/or replicated onto other systems. After that the configuration has to be done. DHCP or BOOTBP can simplify the network configuration step.

2.2 JumpStart Enterprise Toolkit

JumpStart[1] Enterprise Toolkit is a solution from Sun for Solaris. JumpStart itself is not distributed as a Sun product but can be used without any charges or license fees. Support is not available. It is specifically developed for Solaris and helps system administrators by carrying out many installations or setting up of clusters.

A JumpStart installation process is as follows:

1. The client sends a RARP request for his IP address
2. A boot server answers and the client sends a TFTP request for the boot image (kernel)
3. The server sends the kernel boot image
4. The client sends a BOOTP request
5. The server answers and the client NFS mounts the root partition from the installation server
6. The client mounts the configuration server
7. The client mounts the install image and starts the installation procedure

2.3 KickStart

RedHat's KickStart[2] is another tool for automation of installation including the configuration of systems. It is only usable for RedHat distributions. At the beginning the administrator creates a file with all necessary information which would be asked during the manual installation. This file is afterwards used as basis for the answers which now are executed automatically. In addition it is possible to create pre-install and post-install scripts. This enables the customisation of the system after the installation. The KickStart configuration file is hosted on a server and can be requested by clients starting an installation.

KickStart installation process:

1. Preparation of a KickStart file
2. Creation of bootdisk or boot-cd with the KickStart file. The KickStart file may also be provided over the network (BOOTP/DHCP)
3. An installation tree is provided with a copy of binary RedHat Enterprise Linux CDs
4. Installation according to the KickStart configuration file

1 Homepage: http://www.sun.com/bigadmin/content/jet/
2 HOWTO: http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/sysadmin-guide/pt-install-info.html

For preparation of the KickStart file a graphical frontend is provided. That means the administrator does not need to know the specific syntax of RedHat's KickStart tool and may configure all options comfortable with a graphical user interface.

These options include:

- Basic configuration
- Installation method
- Bootloader (i.e.. lilo or grub)
- Partitioning
- Network
- Firewall
- X-Window Manager
- Packets

2.4 FAI (Fully Automatic Installation)

FAI[3] (from Thomas Lange, University of Cologne) is an automation tool for the Debian distribution released under the GNU 2 license. This tool offers already packaged Debian files (.deb) for the versions stable, testing and unstable. First attempts of porting FAI to Ubuntu are currently carried out by the community.

No interaction during installation is needed. All actions are planned before and may be configured. As of this, FAI is a scalable method for installing and updating a network of workstations and/or servers. It consists of a set of shell scripts and Perl scripts. FAI focusses on:

- Remote OS-Installations
- Linux roll-out
- Mass-Installations and/or Cluster Installations

The client which is going to be installed boots the kernel and NFS-mounts the root directory from the installation-server. Classes and variables are defined upon execution of different scripts and may be used to customize the FAI installation. The hardware is detected automatically and necessary drivers are loaded. The classes define different usage of clients (i.e. desktop, server). FAI delivers predefined classes for Beowulf[4] clusters, developer systems and others.

After booting and defining classes the hard-disk is partitioned. The partition table is read from a file and may be customized. After mounting the created filesystems the necessary software packages are installed. Existing dependencies are resolved by Debian. At the end of the installation process, self-created scripts may be executed depending on the class. This allows even more customization of the client. FAI therefore supports different languages like Perl, Cfengine, expect and shell.

3 Homepage: http://www.informatik.uni-koeln.de/fai/
4 Beowulf is a design for high-performance parallel computing clusters on inexpensive personal computer hardware.

The log files produced by the installation are then saved on the hard-disk – as well as on the server if set. After finishing the installation the client boots with the freshly installed system. With this way the installation of a Linux system is extremely fast.

Such an installation takes about 10 minutes on a Athlon 500 MHz PC if at all. 640MB of software is going to be installed. With a software-packet of about 160MB (i.e. Beowulf cluster) it would only take about 2 minutes.

FAI's features are:

- A fully automated installation can be performed.
- Very quick unattended installation
- Update of running systems without re-installation
- Hosts can boot from floppy, network or CD.
- DHCP and BOOTP protocol and PXE boot method are supported.
- Remote login via ssh during installation process possible.
- Two additional virtual terminals available during installation
- All similar configurations are shared among all install clients.
- Log files for all installations can be saved to the installation server.
- Shell, Perl, expect and cfengine scripts are supported for the configuration setup.
- Access to a Debian mirror via NFS, FTP or HTTP
- Can be used as a rescue system.
- Flexible systems through an easy class concept
- Diskless client support
- Easily adds own functions via hooks.
- Automatic hardware detection

2.5 YaST AutoInstaller

AutoYaST[5] is the system for installing one or more SuSE Linux systems automatically and without user intervention. Using AutoYaST, one can create a configuration for a single system or a set of systems to control automated installations. The control file can be provided to YaST2 during installation in different ways.

5 Homepage: http://yast.suse.com/autoinstall/

Like Kickstart, AutoYaST uses a configuration file containing all necessary information. During installation AutoYaST uses existing YaST2 components. AutoYaST supports network-, floppy-, and cd- boots.

Procedure of an AutoYaST installation:

- creation and configuration of control-file (XML format)
- installation of clients (YaST2 components)
- post-install (YaST2 components)

2.6 System Installer Suite

System Installation Suite is an disk-image based installation tool for RPM-based Linux distributions. It is a collection of packages, including SystemInstaller, SystemImager, and SystemConfigurator which provide a total solution for Linux installation and maintenance.

SystemImager facilitates the execution of automated installs (clones), software , content or data distribution, configuration changes, and operating system updates to a network of Linux machines. Even updates from one Linux release version to another are possible.

It can also be used to ensure safe production deployments. By saving a current production image before updating to a new production image, users have a highly reliable contingency mechanism. If the new production environment is found to be flawed, simply roll-back to the last production image with a simple update command.

System Installer Suite consists of these projects:

SystemInstaller[6]

- tool for building Linux images
- for creating installable chroot images for SystemImager

SystemImager[7]

- allows retrieval of an entire system image from a client (manually installed, customized)

SystemConfigurator

- configuration engine for SystemImager

RescueCD

- rescue CD images for deploying without a network

6 Homepage: http://systeminstaller.sourceforge.net/
7 Homepage: http://www.systemimager.org/

2.7 Cfengine

Cfengine, or the configuration engine is an autonomous agent and a middle to high level policy language and agent for building expert systems to administrate and configure large computer networks. Cfengine is designed to be a part of a computer immune system. It is ideal for cluster management and has been adopted for use all over the world in small and huge organizations alike.

Cfengine is designed to let you use configuration files to describe the desired state of a system, rather than describing what should be done to a system to reach that state. You can think of Cfengine configuration files as being in a very high level language - much higher level than Perl or shell scripts. For example, a single Cfengine configuration file statement can result in hundreds of links being created, or the permissions of hundreds of files being set.

The idea of Cfengine is to create a single set of configuration files which describe the setup of all hosts on your network, i.e. a configuration policy is deployed over a network. In a typical setup, Cfengine runs on every host, retrieves configuration data from the master server, and applies changes as needed.

Cfengine focusses upon a few key areas of basic system administration and provides a declarative language designed to remove if/then/else constructs. In this way the configuration files are shorter and more transparent.

Here are some tasks Cfengine can automate:

- Checking and configuring network interfaces.
- Creating, changing and deleting files.
- Making and maintaining symbolic links.
- Checking and setting file permissions.
- Deleting 'junk' files lying around your systems.
- Automated mounting of file systems.
- Controlling execution of scripts and shell commands.
- Monitoring md5 file checksums against a database.
- Checking daemons and restarting them if needed.
- Configuring new installations according to standardized rules.
- Duplicating systems readily.
- Reverting unauthorized changes, whether malicious or not.