

Automated Installation and System Configuration with AutoYaST

Anas Nashif

Copyright © 2003 Anas Nashif

Table of Contents

Introduction	1
Availability	1
Motivation	1
Overview and Concept	2
Reasons why automated setup is needed?	2
Architecture	3
Features	4
Configuration management system	4
Installation sources	4
Hardware detection	4
System Customization	4
Rule Based Installation	4
The Control File	6
Introduction	6
Format	6

Introduction

AutoYaST is a system for installing one or more SuSE Linux systems automatically and without user intervention. AutoYaST installations are performed using a control file with installation and configuration data. The control file can be created using the configuration interface of AutoYaST and can be provided to YaST2 during installation in different ways.

Availability

AutoYaST is available with recent SuSE products starting from *SuSE Linux 8.0* and will be available with business products based on *SLES 8*.

Old products prior to SuSE Linux 8.0 and business products based on *SLES 7* have an auto-installation system relying on *YaST*. A configuration management system is provided by *Alice* for these products.

Motivation

The Linux Journal [<http://www.linuxjournal.com/>], in an article in issue 78 [<http://www.linuxjournal.com/categories.php?op=newincatid=178>] writes:

“ A standard Linux installation asks many questions about what to install, what hardware to configure, how to configure the network interface, etc. Answering these questions once is informative and maybe even fun. But imagine a system engineer who has to set up a new Linux network with a large number of machines. Now, the same issues need to be addressed and the same questions answered repeatedly. This makes the task very inefficient, not to mention a source of irritation and boredom. Hence, a need arises to automate this parameter and option selection.”

“The thought of simply copying the hard disks naturally crosses one’s mind. This can be done quickly, and all the necessary functions and software will be copied without option selection. However, the fact is that simple copying of hard disks causes the individual computers to become too similar. This, in turn, creates an altogether new mission of having to reconfigure the individual settings on each PC. For example, IP addresses for each machine will have to be reset. If this is not done properly, strange and inexplicable behavior results.”

Regular installation of SuSE Linux is semi-automated by default. The user is requested to select the necessary information at the beginning of the installation (Normally Language only) and YaST2 generates a proposal for the underlying system depending on different factors. In most cases, and especially for new systems, such a proposal needs not to be changed and can be used to install the system and provides a usable installation.

The steps following the proposal are fully automated and the user is only prompted at the end of the installation to provide more information about the system.

AutoYaST can be used where no user intervention is required or where customization is required. Using a control file, YaST2 prepares the system for a custom installation and avoids any interaction with the user, unless specified in the file controlling the installation.

AutoYaST is not an automated GUI system. This means that in most cases many screen will be skipped, i.e. you will never see the language selection interface. AutoYaST will simply pass the language parameter to the sub-system without displaying any language related interface.

Overview and Concept

Using AutoYaST multiple systems sharing the same environment and similar but not necessarily identical hardware performing similar tasks can easily be installed in parallel and in a short time. A configuration file (referred to as "control file") is created using existing configuration resources. The control file can be easily tailored for any specific environment.

Unlike autoinstallation systems available with older SuSE releases, AutoYaST is fully integrated and provides various options for installing and configuring a system. The main advantage over older systems and other auto-installation systems is the possibility to configure a computer by using existing modules and avoiding using custom scripts which are normally executed at the end of the installation.

Reasons why automated setup is needed?

Several situations can make non-interactive installations necessary:

- Installation of a large number of similar systems

A large number of client systems or servers for a special service, i.e. compute servers in a high performance cluster have to be installed in a similar manner. It would be too tedious to install every computer interactively. Automation is a necessary tool.

- Installation by less trained operators

Server layout and selection of software and configuration of the main services is a challenging task normally done by well trained administrators and system architects. On the other hand, time of this well paid personal is very limited and often installations have to take place at distant locations. With the use of automation it is possible to `\textit{design}` and configure certain systems in advance and delegate the actual installation to operators in the local subsidiary, for instance.

- Installation over a large distance

Sometimes installation have to be made in locations without techical IT personal or over such large distances that travel expenses are too high. In these situations, it would be ideal if one could send a well prepared installation CD to the location and after booting the computer with this CD installed and a few minutes the computer is configured correctly and reachable through network services.

- Quality assurance for important server systems

In a productive environment, one wishes to have a consistent set of servers who are identical in the layout and basic configuration. It is very hard to ensure this by written installation guides, etc. Using automation this is done by design and additionally one archives a good documentation of the installed systems.

Architecture

The complete auto-installation process can be divided in three phases: Preparation, Installation and Configuration:



- Preparation: All relevant information about the target system is collected and turned into the appropriate directives of the control file. The control file is transferred onto the target system where its directives will be parsed and transformed to YaST2 conforming data.
- Installation: follows the instructions given in the control file and installs a base system.
- Configuration: YaST2 and some user-defined post-install scripts accomplish the system configuration

Features

AutoYaST provides a lot of new features, starting from a graphical configuration management system to user defined postinstall options.

Configuration management system

In order to create a control file or a certain class of control files, one can use a YaST module.

Installation sources

There are quite a number of different installation sources and control file sources. In the simplest scenario, one can use the standard SuSE Linux Enterprise Server CDs as installation sources and provide the control file on a DOS formatted floppy disk. In a more advanced situation, but without network connectivity, YaST can create a ISO image file which contains all packages and the control file. Using the network, installation can be done using NFS (Network File System). Then the control file can be provided in different places:

- on a floppy disk
- in an NFS repository
- via HTTP, SMB or FTP
- TFTP

Booting can be done via network (etherboot/PXE) or via CDROM.

Hardware detection

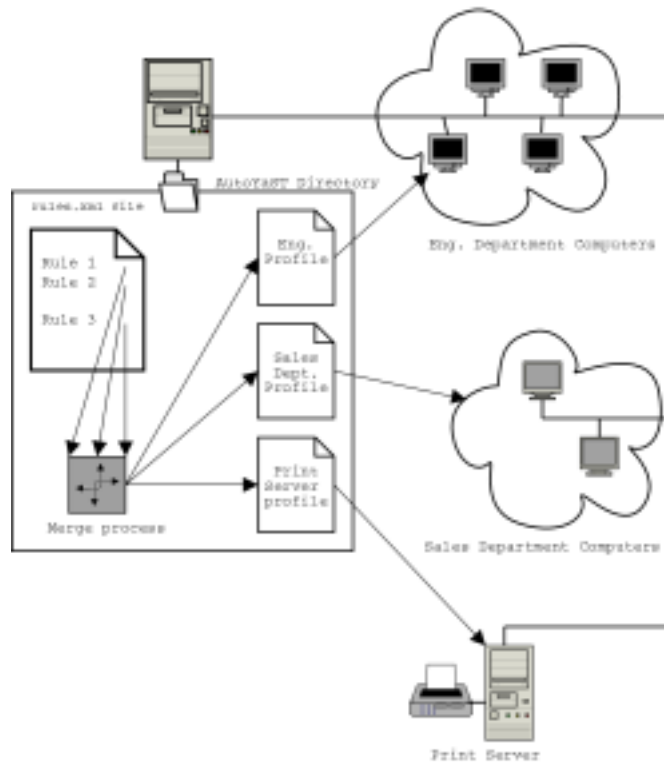
During autoinstall, the same convenient hardware detection is used as for interactive install using YaST. This is achieved through the complete integration of AutoYaST into the installation and configuration system.

System Customization

Most of the system customization is done in the second stage of the installation. YaST provides most of the important resources needed to bring up a system to a usable, general state. However, you may have other requirements for the installed system. If the required customizations can't be done using YaST resources, then the post-install scripts can be used to accomplish this task. You can define an unlimited number of custom scripts in the control file either by editing the control file or by using the configuration system.

Rule Based Installation

Rules offer the possibility to configure a system depending on system attributes by merging multiple control file during installation. The rules based installation is controlled by a rules file.



The rules file is an XML based file that contains rules for each group of systems (or single systems) that you want to automatically install. A set of rules distinguish a group of systems based on one or more system attributes, after passing all rules, it links each group of rules to a profile. Both the rules file and the profiles must be located in a pre-defined and accessible location.

If more than one rule apply, the final profile for each group is generated on the fly using a merge script. The merging process is based on the order of the rules and later rules override configuration data in earlier rules.

The use of a rules file is optional. If the rules file is not found, system installation proceeds in the classic way by just using the supplied profile or by searching for the profile depending on the *MAC* or the *IP* address of the system.

The following simple example illustrates how the rules file is used to retrieve the configuration for a client with known hardware.

```
<?xml version="1.0"?>
<!DOCTYPE autoinstall SYSTEM "/usr/share/autoinstall/dtd/rules.dtd">
<autoinstall xmlns="http://www.suse.com/1.0/yast2ns"
xmlns:config="http://www.suse.com/1.0/configs">
  <rules config:type="list">
    <rule>
      <disksize>
        <match>/dev/hdc 1000</match>
        <match_type>greater</match_type>
      </disksize>
      <result>
        <profile>machine1.xml</profile>
        <continue config:type="boolean">>false</continue>
      </result>
    </rule>
    <rule>
      <disksize>
        <match>/dev/hda 1000</match>
        <match_type>greater</match_type>
```

```

        </disksize>
        <result>
            <profile>machine2.xml</profile>
            <continue config:type="boolean">false</continue>
        </result>
    </rule>
</rules>
</autoinstall>

```

The last example defines 2 rules and provides a different profile for every rule. The rule used in this case is *disksize*. After parsing the rules file, YaST2 attempts to match the system being installed to the rules in the `rules.xml` file in order: first rule through the last rule. A rule match occurs when the system being installed matches all of the system attributes defined in the rule. As soon as a system matches a rule, the result resource is added to the stack of profiles AutoYaST will be using to create the final profile. The *continue* property tells AutoYaST if it should continue with other rules or not.

If the first rule does not match, next rule in the list is examined until a match is found.

Using the *disksize* attribute, you can provide different configurations for different hard drives with different size. First rule checks if the device `/dev/hdc` is available and if it is greater than 1 GB in size using the *match* property.

The Control File

Introduction

The control file is in most cases a configuration description for a single system. It consists of sets of resources with properties including support for complex structures representations such as lists, records, trees and large embedded or referenced objects.

Format

The XML configuration format provides a consistent file structure, which is easier to learn and remember when attempting to configure a new system.

Using XML, you can eliminate (nearly) all of the configuration file parsing and error handling - an external XML parser can do that instead - (especially if it is a validating parser). To make sure the control file is well-formatted and that the syntax is valid, you can run the control file through a validating parser before it is actually used for automatic installation. This is especially required if you prefer to edit the profile manually.

The following example shows a control file in XML format:

```

<?xml version="1.0"?>
<!DOCTYPE profile SYSTEM
"/usr/share/autoinstall/dtd/profile.dtd">
<profile
xmlns="http://www.suse.com/1.0/yast2ns"
xmlns:cfg="http://www.suse.com/1.0/configs">
<install>
    <partitioning config:type="list">
        <drive>
            <device>/dev/hda</device>
            <partition>
                <filesystem>ext2</filesystem>
                <size>520Mb</size>
                <mount>/</mount>
            </partition>

```

```

        <partition>
            <filesystem>reiser</filesystem>
            <size>1200Mb</size>
            <mount>/data</mount>
        </partition>
    </drive>
</partitioning>
</install>
<configure>
    <scripts>
        <pre-scripts>
            <script>
                <interpreter>shell</interpreter>
                <filename>start.sh</filename>
                <source>

                    <![CDATA[
#!/bin/sh
echo "Starting installation"
exit 0
]]>

            </source>
        </script>
    </pre-scripts>
</scripts>
</configure>
</profile>

```