Automating Linux installations

An introduction to Red Hat Kickstart and SUSE AutoYaST installation tools

Agenda

- The manual install process (very brief)
- Network booting in a PC environment (very brief)
  - PXE, DHCP, TFTP
- Installation servers for Linux systems (brief)
  - PXELinux, DHCP, TFTP, NFS
- Automating Red Hat/Fedora installs, Kickstart
- Automating SUSE installs, AutoYaST

Typical manual install

- Boot from CD/DVD
- Prompt user for information about the installation
- Read packages from the CD/DVD
- Tool completes basic installation and configuration of host

Issues with typical install

- Boot from CD/DVD
  - Requires Disks and Drives for each system
  - May be difficult to get physical access to host
  - Does not scale well
- Prompt user for information about the installation
  - Requires valuable admin time
  - Possible error path particularly with multiple, “identical” builds
  - Does not scale well
- Read packages from the CD/DVD
  - See item #1
- Tool completes basic installation and configuration of host
  - May require post-install configuration to complete the process

Resolving typical install issues

- Boot from local medium (CD/DVD)
  - Network (diskless) boot of install image
- Prompt user for information about the installation
  - Specify configuration in advance then read
    - from local media (still has problem of physical access)
    - over network network via TFTP, NFS, HTTP, et al.
- Local package repository (CD/DVD)
  - Network repository via NFS, TFTP, HTTP, CIFS etc.
- Post-install configuration
  - Include post-install scripts as part of system specification

Network booting in a PC environment

- At startup, system loads network boot code from ROM or Disk
  - PXE, BootROM, Etherboot, Netboot, OpenBoot
- Boot code queries network for IP config and location of boot file
  - DHCP, BOOTP, RARP
- Boot file is loaded over the network and executed
  - TFTP
  - Note: This may be the actual OS kernel or an intermediate step in the boot process
- There are many combinations however we will focus on
  - PXE (PXELinux) / DHCP / TFTP

Linux network based install services

- The following services are required to boot and install a Linux system from the network.
  - They do not need to be on the same server
- DHCP
  - Required to provide configuration information to host at boot
- PXELinux
  - Required to load install kernel and InitRD
- TFTP
  - Required to serve PXE binary, install kernel and InitRD
  - May be used to serve the install config file and RPMs
- NFS, HTTP, other file services
  - May be used to serve the install config file and RPMs

PXELinux

- **P**reboot e**X**ecution **E**nvironment for Linux
- Docs and binaries at http://syslinux.zytor.com/pxe.php
- Part of the *syslinux* RPM distributed with RH, Fedora and SUSE
- Booting with PXELinux
  - BIOS uses DHCP/TFTP to retrieve *pxelinux.0* boot binary
  - Control is then passed to *pxelinux.0* which
    - retrieves PXE configuration file via TFTP
    - retrieves and boots the actual install environment as specified in the config file which consists of
      - Linux Kernel
      - RAM disk image of the root file system

PXELinux Configuration file

- Basic text file
- We only need to specify two options
  - **kernel** which specifies where to find the Linux install kernel on the TFTP server
    - e.g. kernel /SUSE/9.3.linux
  - **append** which specifes the boot arguments to pass to the install kernel
    - e.g. append initrd=/SUSE/9.3.initrd

Example configuration file:

```
LABEL linux

  # Fedora
  kernel /Fedora/vmlinuz
  append initrd=/Fedora/initrd.img ramdisk_size=8192 ks=http://10.0.0.10/kickstart/ks.cfg
```

Location of PXELinux config files

- Request files from /<bootdir>/pxelinux.cfg/ where <bootdir> is location pxelinux.0 was served from
- Request file using the ARP type code and hardware address, all in lower case hexadecimal with dash separators.
- If not found, use own IP address in upper case hexadecimal
  - **gethostip** utility is included with syslinux package
- If not found, remove one hex digit and try again, repeat until found or no digits left
- If not found, look for "default"
- Note: You may want to consider using soft links here
  - Common config file with multiple MAC or Hex IP named links pointing to it.

Example of PXELinux config file search

Example:

    Binary served from /tftpboot/pxelinux.0
    Ethernet NIC which means ARP Type 1
    MAC MAC Address = C0:FF:EE;00:00:01
    IP address = 192.0.2.91 = C000025B

Search:

```
/tftpboot/pxelinux.cfg/01-c0-ff-ee-00-00-01
/tftpboot/pxelinux.cfg/C000025B
/tftpboot/pxelinux.cfg/C000025
/tftpboot/pxelinux.cfg/C00002
/tftpboot/pxelinux.cfg/C0000
/tftpboot/pxelinux.cfg/C000
/tftpboot/pxelinux.cfg/C00
/tftpboot/pxelinux.cfg/C0
/tftpboot/pxelinux.cfg/C
/tftpboot/pxelinux.cfg/default
```

DHCP Server

- Red Hat / Fedora and SUSE include ISC's DHCP
  - http://www.isc.org/isc/dhcp.html
- Configuration
  - A an example configuration is included in the *Notes* section of this presentation
  - Items to note:
    - Definition of *PXE option space* and *pxeclients class* to limit responses to PXE boot requests
    - filename "pxelinux.0"; Load PXELinux binary
    - Next-server 10.0.0.10; Where to find it

dhcp.conf: PXE option space

```
# PXE specific options      -=-=-=-=-=         -=-=-=-=-=         -=-=-=-=-=

# Code 1: Multicast IP address of boot file server
# Code 2: UDP port that client should monitor for MTFTP responses
# Code 3: UDP port that MTFTP servers are using to listen for MTFTP requests
# Code 4: Number of seconds a client must listen for activity before trying
#            to start a new MTFTP transfer
# Code 5: Number of seconds a client must listen before trying to restart
#             a MTFTP transfer
option space PXE;
   option PXE.mtftp-ip                  code 1 = ip-address;
   option PXE.mtftp-cport               code 2 = unsigned integer 16;
   option PXE.mtftp-sport               code 3 = unsigned integer 16;
   option PXE.mtftp-tmout               code 4 = unsigned integer 8;
   option PXE.mtftp-delay               code 5 = unsigned integer 8;
   option PXE.discovery-control         code 6 = unsigned integer 8;
   option PXE.discovery-mcast-addr code 7 = ip-address;
```

dhcp.conf: pxeclients class

```
# Classes                    -=-=-=-=-=          -=-=-=-=-=          -=-=-=-=-=

# Match hosts presenting the PXEClient VCI
class "pxeclients" {
     match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
     option vendor-class-identifier "PXEClient";

     # Client has all the info and doesn't need to try for more at port 4011
     option dhcp-parameter-request-list 60,43;

     vendor-option-space PXE;

     # At least one of the vendor-specific PXE options must be set in
     # order for the client boot ROMs to realize that we are a PXE-compliant
     # server.  We set the MCAST IP address to 0.0.0.0 to tell the boot ROM
     # that we can't provide multicast TFTP (address 0.0.0.0 means no
     # address).
     option PXE.mtftp-ip   0.0.0.0;

     # Boot file name is present in initial DHCP offer
     option PXE.discovery-control 8;

     # This is the name of the file the boot ROMs should download.
     filename "pxelinux.0";

     # This is the name of the server they should get it from.
     next-server 10.0.0.10;
} #class pxeclients
```

TFTP server

- Red Hat / Fedora and SUSE all include TFTP packages
  - Configuration is trivial, defaults to serving from /tftpboot
  - Consider *atftp* with SUSE. It does not require (x)inetd
- Consider the following directory structure

/tftpboot/pxelinux.0	PXE binary
/tftpboot/pxelinux.cfg/	PXELinux host config files
/tftpboot/<distro>/	Distribution's install environment

- Where to find the Install kernels and initial RAM Disks

SUSE	: <CD/DVD>/boot/loader/ linux & initrd
RH/FC	: <CD/DVD>/images/pxeboot/ vmlinuz & initrd.img

Kickstart

Red Hat Kickstart

- Kickstart is the automated install mode of RH's Anaconda installer
- How it works:
  - The desired system configuration is entered in a simple text file.
  - The “ks” kernel argument alerts Anaconda to retrieve the file and perform an automated install
  - The system configuration file may be placed on local media or made available across the network via NFS or HTTP. (Sorry, not TFTP)

The Kickstart configuration file

- Simple text file consisting of 4 sections:
  - Comand
  - %packages
  - %pre
  - %post
- Sections **must** be specified in order
  - %pre and %post may be swapped and are optional
- It is **not** necessary to include all sections. Not specifying a required section will result in the installer prompting for a manual input of the required information.
- Lines beginning with "#" are ignored (Comments)

Command section

- First section of config file. Not labled
- Basic install options and system configuration, e.g
  - Location of install media
  - How disks should be partitioned
  - How networking should be configured
  - Should X be configured
  - Should system reboot or power down at end of install
- Full listing of options in Section 1.4 of RHEL 4 System Administration Guide

%include option

- %include /path/to/local/file
- Include contents of local file at this point in the configuration file.
- This is parsed after the %pre script has executed (>= RHEL 3)
  - This allows some capability to modify the config file "on the fly".
  - Example 1.6.1 on page 20 of RHEL 4 System Administration Guide shows how this may be used to alter the partitioning specification based on the output of the %pre script.
  - YMMV (I've never tried this)

%packages section

- Specifies what packages to install
- Begins with “%packages [--options]”
- Must follow command section
- Packages may be specified in groups and/or as individual RPMs
  - 1 item per line
  - Groups specified by "@ " followed by group name or ID as listed in comps.xml. (example follows)
  - RPMs specified by name portion of RPM
    - e.g. “*ethereal*” not “*ethereal-0.10.9-4.i586.rpm*”
  - RPMs may be specifically omitted by preceding the name with a "-"
    - useful if you want to specify a group but omit some of the files withing the group

Specifying groups in %packages

- Groups “Core” and “Base” are always selected by default. It is not necessary to specify them.
- Group are defined in CD1 <RedHat | Fedora>/base/comps.xml

Example: Specifying group “Administration tools” in Fedora Core 4

FC4 CD1, Fedora/base/comps.xml

```
<group>
  <id>admin-tools</id>
  <name>Administration Tools</name>
  ...
 </group>
```

Kickstart configuration file:

```
@ Administration Tools
    - OR -
@ admin-tools
```

%packages options

- resolvedeps
  - Automatically include additional RPMs as required to resolve dependencies
- ignoredeps
  - Ignore unresolved dependencies and install specified RPMs
- ignoremissing
  - It's OK if a specified package is not available, skip it and continue. Do not abort the install.

%pre section

- Script to be run **before** the installation
- Begins with “%pre [--interpreter]”
- If used, must follow %packages section
- Uses bourne shell by default
- Run immediately after the config file has been parsed
- Target file system not yet mounted at /mnt/sysimage
- Limited environment, e.g. No DNS, minimal utilities
- Useful for tasks such as sanity checking before proceeding with the install, customizing partitioning based on hardware

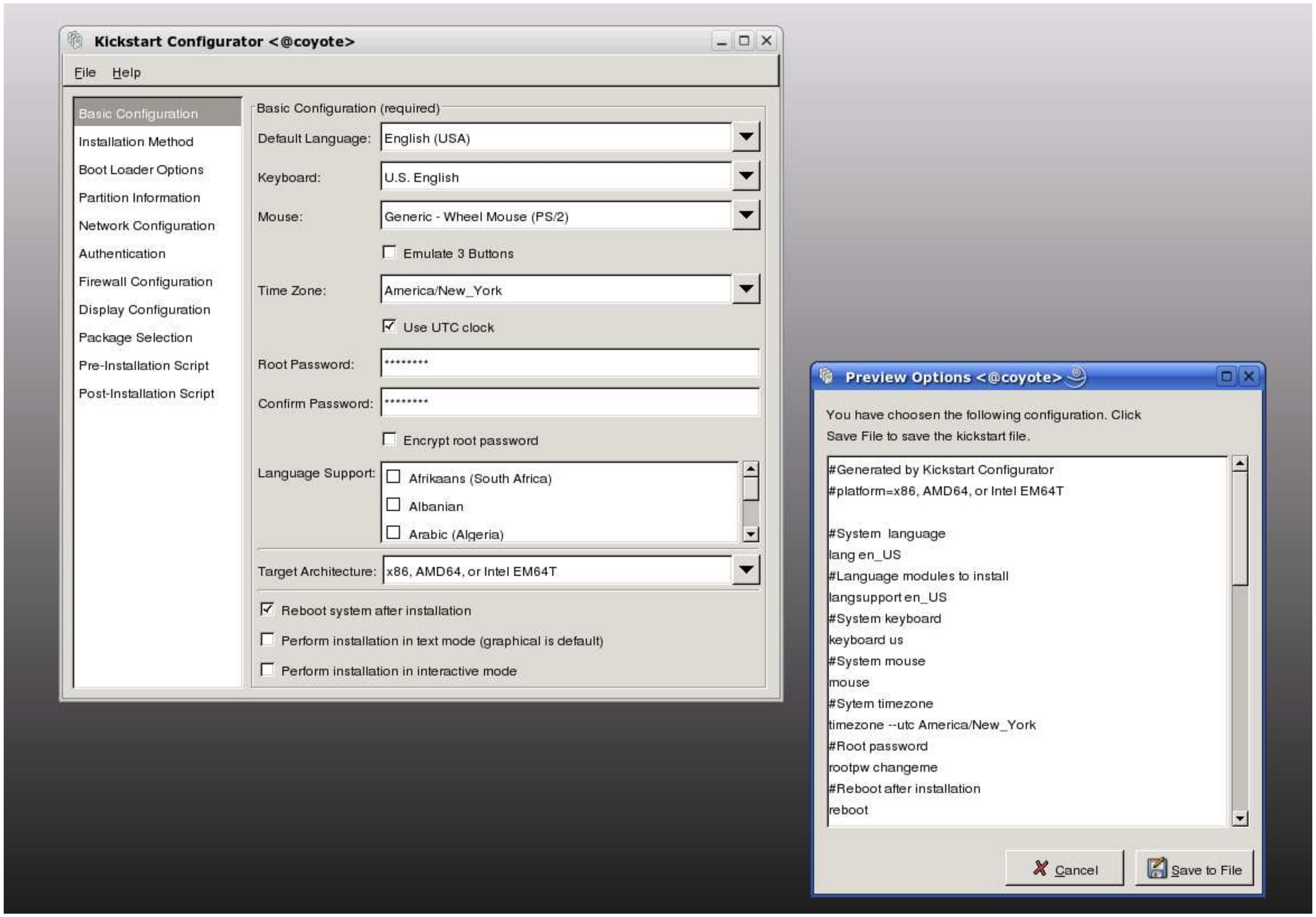
%post section

- Script to be run **after** the installation
- Begins with “%post [--interpreter] [--nochroot]”
- If used, must follow %packages section
- Uses bourne shell by default
- Run in the chroot /mnt/sysimage environment
  - Installation media is mounted at old root so it is no longer available
  - Access to full system with all the installed utilities
- Useful for site specific custom configuration
  - Turn default services on/off
  - Customize daemon config files

Creating the Kickstart configuration file

- Manually
  - Useful for tweaking an existing config file but probably not something you want to do from scratch.
- *anaconda-ks.cfg* from an existing system
  - After completing an install anaconda generates a file reflecting the install configuration in */root/anaconda-ks.cfg*
  - This file will need to be edited
    - some options commented out, e.g. partition information
    - %pre & %post sections not included
- /usr/sbin/system-config-kickstart
  - X based tool to create and edit configuration files
  - Very similar to screens of GUI guided install
  - system-config-kickstart-<version>.noarch.rpm

Kickstart Configurator



Basic Configuration (required)

Default Language: English (USA)
Keyboard: U.S. English
Mouse: Generic - Wheel Mouse (PS/2)
☐ Emulate 3 Buttons
Time Zone: America/New_York
☑ Use UTC clock
Root Password: ********
Confirm Password: ********
☐ Encrypt root password
Language Support: ☐ Afrikaans (South Africa) ☐ Albanian ☐ Arabic (Algeria)
Target Architecture: x86, AMD64, or Intel EM64T
☑ Reboot system after installation
☐ Perform installation in text mode (graphical is default)
☐ Perform installation in interactive mode

You have choosen the following configuration. Click Save File to save the kickstart file.

```
#Generated by Kickstart Configurator
#platform=x86, AMD64, or Intel EM64T

#System  language
lang en_US
#Language modules to install
langsupport en_US
#System keyboard
keyboard us
#System mouse
mouse
#Sytem timezone
timezone --utc America/New_York
#Root password
rootpw changeme
#Reboot after installation
reboot
```

Making the Kickstart file available

- The “ks” kernel argument specifies an automated install and the location of the configuration file
- Locally via floppy, CD/DVD, Other local device
  - ks=floppy[:/<path>]   path defaults to /ks.cfg
  - ks=cdrom[:/<path>]
  - ks=hd:<device>:/<file>
- Over the network via NFS or HTTP
  - ks=nfs:<server>:/<path>
  - ks=http://<server>/<path>
  - ks, via NFS, filename from DHCP or based on IP address

Making the packages available

- Locally via CD or DVD (Default)
- Over the network via NFS, FTP or HTTP
  - Common installation tree made by copying contents of all RedHat | Fedora directories to a common location
  - ISO images of the CDs in a common directory (NFS)
    - Only one release per directory
  - loopback mounted copies of each CD (FTP, HTTP)
    - Mount each image at the same level
    - Use the names “disc1” ... “disc4” for the mount points

Some useful kernel arguments

- Serial console for headless installs
  - *console=device,options*, e.g. console=ttyS0,115200n
- VNC for remote access to the anaconda GUI
  - *vnc [vncpassword=<password>] [vncconnect=<client>[:<port>]]*
    - vncconnect permits server to connect out to a viewer which has been started with the "*-listen*" option
  - Useful for remote access to a manual install or to monitor a Kickstart automated installation
- *ks* to initiate a kickstart install
- Note: All of these may be passed to the boot kernel through the ***append*** line of a PXELinux configuration file

Putting it all together

Install Server:

- DHCP server to provide boot information for PXELinux
- TFTP server to provide
  - PXELinux binary and configuration files
  - Install environment kernel and RAM Disk
- NFS, FTP or HTTP to provide release RPMs and Kickstart file

Host configuration:

- Create the Kickstart configuration file
- Configure PXELinux to append "*ks*" argument to kernel
  - Other arguments as desired, e.g. console, vnc
- Configure the host to boot via PXE

AutoYaST

SUSE AutoYaST

- SUSE's automated install tool
- How it works:
  - The desired system profile is stored in XML Rule and Control files
  - The “autoyast” kernel argument instructs the installer to retrieve a specified control file or select one based on the rules.xml file.
    - XML files on local media or made available across the network via NFS, HTTP, TFTP or FTP
  - Control file is parsed, information passed to respective yast modules, pre-install scripts are executed
  - Yast completes initial install based on the retrieved profile.

The Autoyast control file

- XML text file which defines resources and their properties
  - Simple property such as the size of a hard drive partition
  - Complex property such as a list or complete script
- Limited documentation regarding which resources are defined and valid values for a resource
- Most documentation on official site is a variation on that which is included with the autoyast package
  - file:///usr/share/doc/packages/autoyast2/html/index.html
- XML DTD
  - /usr/share/autoinstall/dtd/profile.dtd
- SUSE Autoinstall list
  - suse-autoinstall-subscribe@suse.com

Rules

- Rules allow you to generate a profile at the time of install based on system attributes.
- This is done by merging one or more control files based on conditions specified in a rules file.
- The rules file is retrieved only if no specific control is supplied using the autoyast keyword
- The use of a rule file is optional.
- If a rules file is used it must:
  - Be valid XML
  - Have at least one rule
  - Have at least one match with a system attribute
  - Be called “*rules.xml*” and be located in the directory “*rules*” of the profile repositroy

System Attributes for rules

Attribute	Values	Description
hostaddress	IP address of host	exact match
domain	Domain name of host	exact match
network	Network address of host	exact match
mac	MAC address of host /C0FFEE001122/	exact match
linux	Number installed Linux partitions	>=0
others	Number installed non-Linux partitions	>=0
xserver	X Server needed for graphic adapter	exact match
memsize	Memory available on host in (Mbytes)	All match types are available
totaldisk	Total disk space available (Mbytes)	All match types are available
haspcmica	System has PCMCIA (i.e Laptops)	exact match, 1 for PCMCIA or 0 for none
hostid	Hex representation of IP address	exact match
arch	Architecture of host	exact match
karch	Kernel Architecture (e.g. SMP, Athlon)	exact match
disksize	Drive device and size	All match types are available
product	Hardware product name as specified in SMBIOS	exact match
product_vendor	Hardware vendor as specified in SMBIOS	exact match
board	System board name as specified in SMBIOS	exact match
board_vendor	System board vendor as specified in SMBIOS	exact match
custom1-5	Custom rules using shell scripts	All match types are available

Example rules.xml

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/configns">
  <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>
```

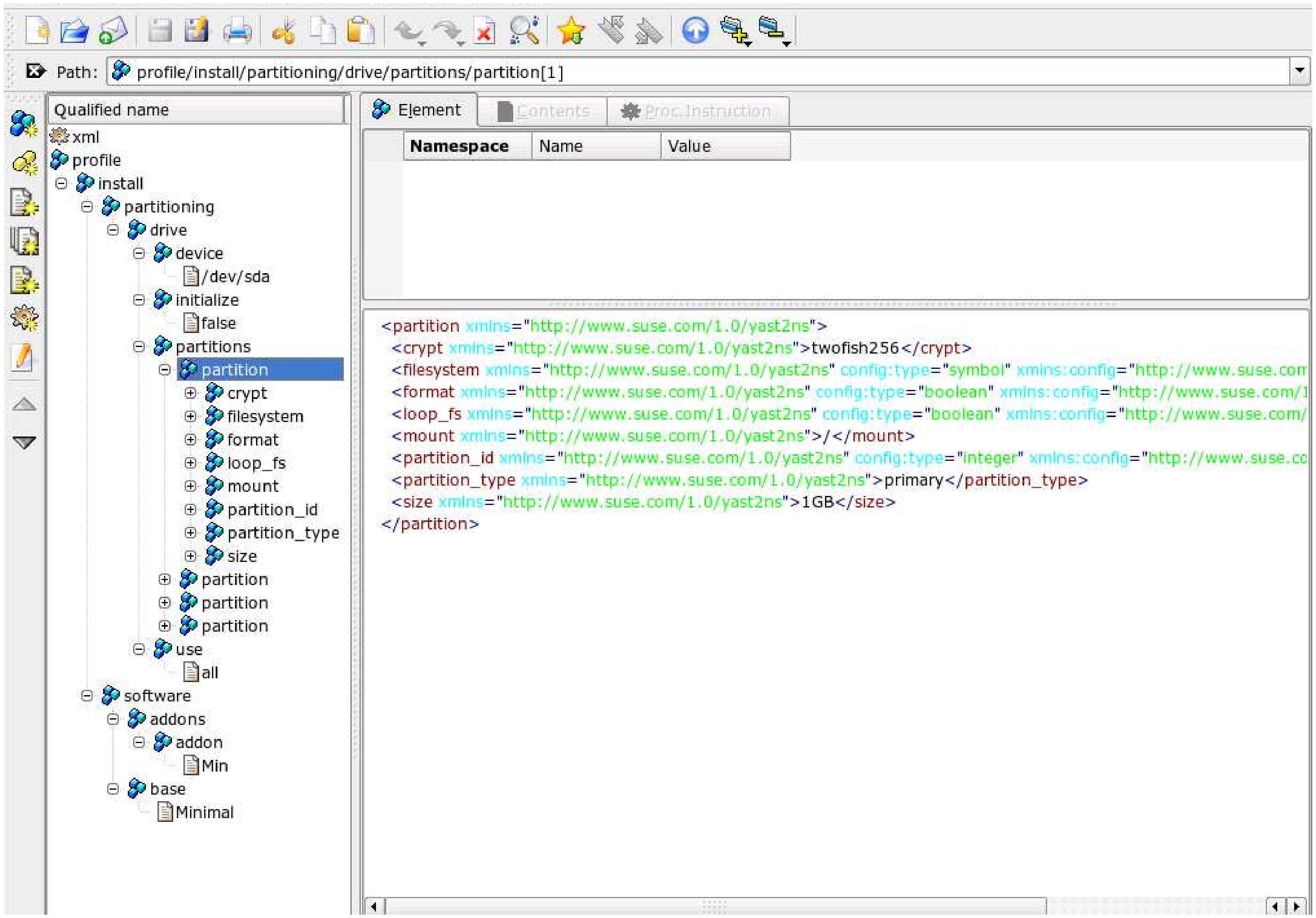
Classes

- Common configuration that can be merged into the final profile
- Defined using the same syntax and format as a complete control file
- Multiple classes may be merged into a final profle
- Merging may be done automatically at install or in advance using the YaST2 autoyast module.
- May care to think of a class as an include file.
- Classes and Rules may be used in combination

Creating the Autoyast control file

- Manually
  - Remember, this must be a valid XML file
    - Validate with *xmllint*
    - Use an XML editor, e.g. *kxmleditor*
- Use the autoyast YaST2 module
  - /sbin/yast2 autoyast
    - GUI similar to installer
    - Ability to create a “Reference Profile” from current system
    - Ability to import a KickStart config file

kxmleditor

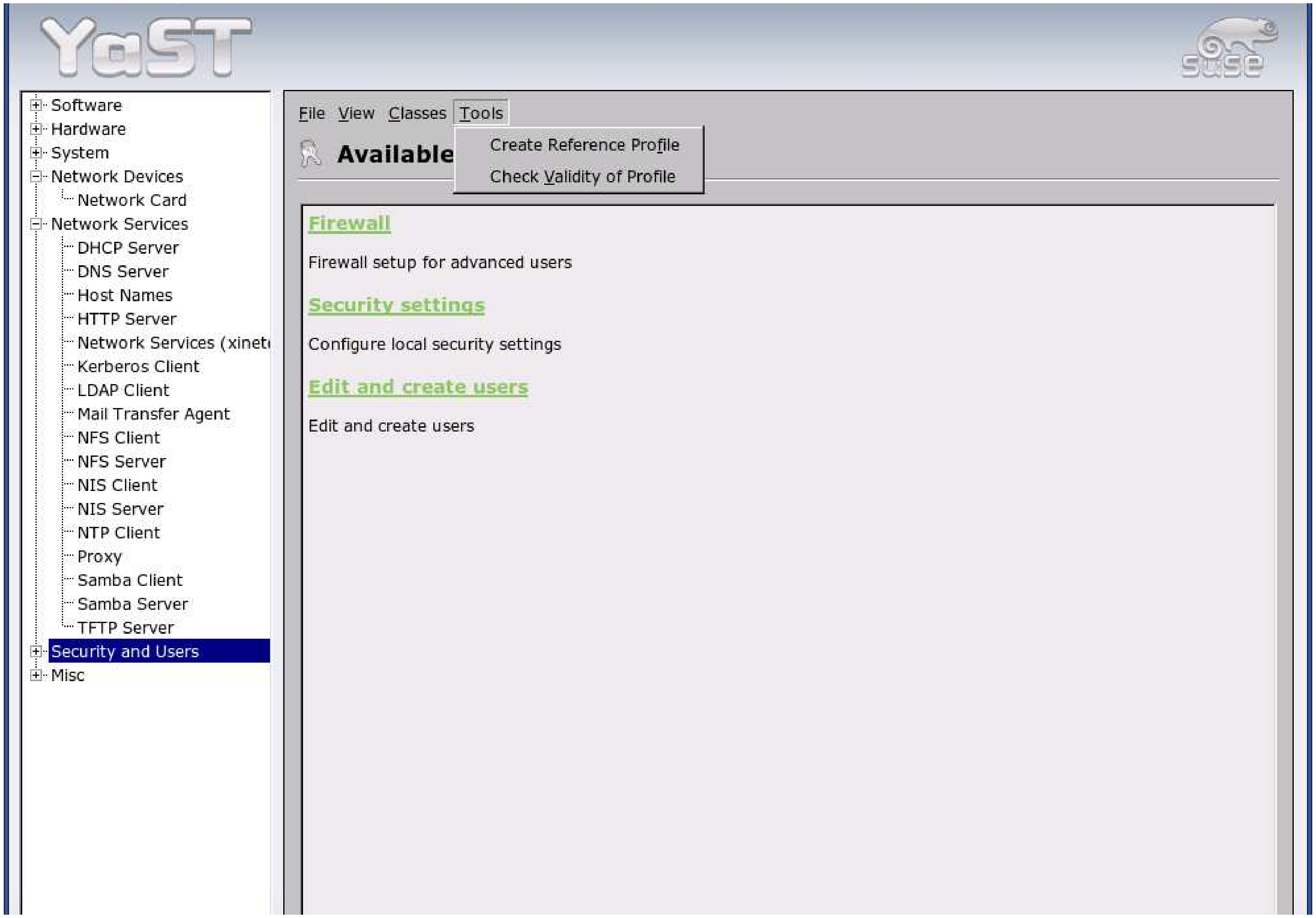


Path: profile/install/partitioning/drive/partitions/partition[1]

```
xml
profile
  install
    partitioning
      drive
        device
          /dev/sda
        initialize
          false
        partitions
          partition
            crypt
            filesystem
            format
            loop_fs
            mount
            partition_id
            partition_type
            size
          partition
          partition
          partition
        use
          all
    software
      addons
        addon
          Min
      base
        Minimal
```

```
<partition xmlns="http://www.suse.com/1.0/yast2ns">
  <crypt xmlns="http://www.suse.com/1.0/yast2ns">twofish256</crypt>
  <filesystem xmlns="http://www.suse.com/1.0/yast2ns" config:type="symbol" xmlns:config="http://www.suse.com
  <format xmlns="http://www.suse.com/1.0/yast2ns" config:type="boolean" xmlns:config="http://www.suse.com/1
  <loop_fs xmlns="http://www.suse.com/1.0/yast2ns" config:type="boolean" xmlns:config="http://www.suse.com/
  <mount xmlns="http://www.suse.com/1.0/yast2ns">/</mount>
  <partition_id xmlns="http://www.suse.com/1.0/yast2ns" config:type="integer" xmlns:config="http://www.suse.co
  <partition_type xmlns="http://www.suse.com/1.0/yast2ns">primary</partition_type>
  <size xmlns="http://www.suse.com/1.0/yast2ns">1GB</size>
</partition>
```

YaST2 autoyast module



Firewall

Firewall setup for advanced users

Security settings

Configure local security settings

Edit and create users

Edit and create users

Making the control file available

- The “autoyast” kernel argument specifies an automated install and which control file to use
- Locally via floppy, CD/DVD, Other local device
  - autoyast=file://<path>
  - autoyast=device://<device>/<file>
  - autoyast=floppy://<path>
- Over the network via NFS, HTTP, TFTP or FTP
  - autoyast=<nfs|http|ftp|tftp>://<server>/<path>
- See docs for algorithm to determine filename if only a directory path is specified

Making the packages available

- Locally via CD or DVD (Default)
- Over the network via NFS, CIFS, HTTP, FTP or TFTP
  - loopback mounted copies of each CD (FTP, HTTP)
    - Mount each image at the same level
    - Use the names “CD1” ... “CD5” for the mount points
  - Use “install” kernel argument to specify location of packages
    - e.g. install=nfs://10.16.72.128/export/SuSE/9.3/CD1

Some useful kernel arguments

- Serial console for headless installs
  - *console=device,options*, e.g. console=ttyS0,115200n
- VNC for remote access to the anaconda GUI
  - *vnc=1 [vncpassword=<password>]*
    - Unlike RH, no VNC Connect option
  - Useful for remote access to a manual install or to monitor an automated installation
- *autoyast* to initiate an automated install
- Note: All of these may be passed to the boot kernel through the ***append*** line of a PXELinux configuration file

Debugging

- YaST2 writes log information to files below /var/log/YaST2
- The kernel argument “Y2DEBUG=1” will provides more verbose logging
- The *save_y2logs* utility will create a gzipped tar of the log files which may then be pushed to another system
  - e.g. save_y2logs /tmp/y2logs.tgz

Putting it all together

Install Server:

- DHCP server to provide boot information for PXELinux
- TFTP server to provide
  - PXELinux binary and configuration files
  - Install environment kernel and RAM Disk
- NFS, (CIFS), HTTP, FTP or TFTP to provide release RPMs and control file

Host configuration:

- Create the rules and profile XML files
- Configure PXELinux to append "*autoyast*" argument to kernel
  - Other arguments as desired, e.g. console, vnc
- Configure the host to boot via PXE