HOW TO MAINTAIN HUNDREDS OF COMPUTERS OFFERING DIFFERENT FUNCTIONALITIES WITH ONLY TWO SYSTEM ADMINISTRATORS

R. Krempaska, A. Bertrand, C. Higgs, R. Kapeller, H. Lutz, M. Provenzano
Paul Scherrer Institute, 5232 Villigen PSI, Switzerland.

Abstract

At the Paul Scherrer Institute, the control systems of our large research facilities are maintained by the Controls section. These facilities include the two proton accelerators, (HIPA and PROSCAN), the two electron accelerators, (SLS and the Injector Test Facility of the future SwissFEL) as well as the control systems of all their related beamlines and test facilities.

This paper describes methods and tools which are used to develop and maintain the challenging computing infrastructure deployed by the Controls section.

CONTROLS COMPUTING INFRASTRUCTURE

The Control system is basically composed of Input Output Controllers (IOCs) running VxWorks operating system on VME hardware using the EPICS (Experimental Physics and Industrial Control System) software. Additionally, there is an increasing number of non VME based IOCs, (so called EPICS soft-IOCs) running mainly on Linux hosts. In total the we are responsible for the installation, configuration and maintenance of up to 400 VME-based IOCs and more than 200 soft-IOCs. There is a remote console interaction with these systems in order to support their development, booting and debugging. Additionally, the control system includes special EPICS servers such as the CCD camera systems that often need dedicated configurations.

Finally, the client part of control system applications requires operator and expert consoles, login clusters and status displays. The Control system configuration and applications software for each facility is stored on independent NFS file servers. The total number of Linux computers and servers is about five hundred. Since only two system administrators are responsible for their installation, configuration and maintenance, we have adopted a well defined solution to face this challenging task:

- Virtualization
- Unified operating system installation and update mechanism
- Automatic configuration by a common tool (puppet)

VIRTUALIZATION

Virtualization allows the system administrators to configure a multitude of "one-task" simple machines such that administrative tasks can run on their own virtual computer. These virtual computers do not require a lot of CPU or memory. This is particularly useful for software upgrades and maintanence. Such systems include boot servers, soft-IOC servers, port server hosts and configuration servers (also called auto-save and restore servers). Computers providing access to private machine networks, such as secure shell (ssh) gateways and Channel Access gateways, are also installed as virtual machines.

Virtualization also enables a rationalization of hardware, operating and energy costs. Our virtual computers run on a VMware cluster (two servers, each with 72Gbytes RAM and SAN storage). Of the 500 Linux computers, about 200 are virtual machines running on the VMware cluster. The remaining systems, (NFS file servers, archiver or development servers) are installed on dedicated hardware blades.

UNIFIED OPERATING SYSTEM INSTALLATION

At PSI the popular Scientific Linux (SL) distribution is used [1]. The PSI Central Computing Division is in charge of distributing and maintaining the SL core and rpm packages [2]. SL is supported by various labs and universities around the world and is based on Red Hat Enterprise, which is recompiled from source and repackaged.

Using the so called "kickstart" mechanism of Red Hat Enterprise, we deploy the base operating system installation which is "tailored" by a software distribution mechanism, called puppet (see next section). The goal is to use the same system management framework for all hosts through their life cycle.

Configuration changes and software updates, (e.g. kernel upgrades) are steered from a central location. Operator consoles in the control room are installed on desktop computers with a standard installation in order to guarantee redundancy and a fast and easy exchange in the case of failure or upgrade.

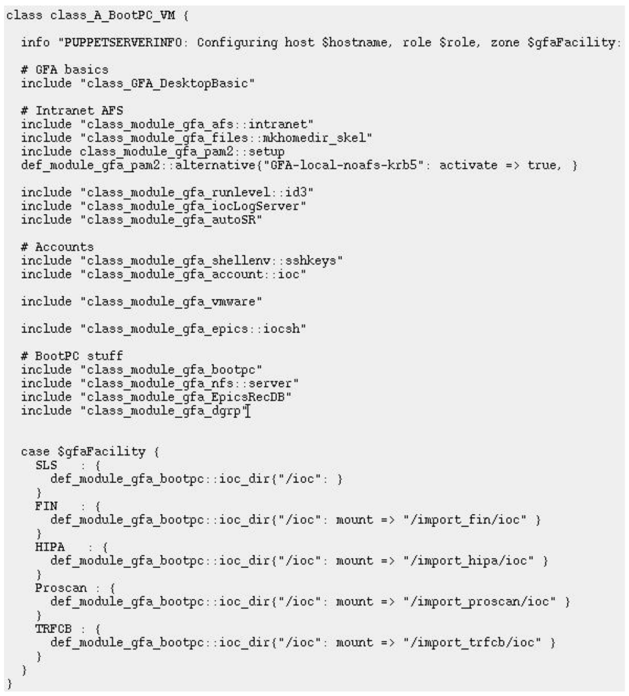
Thus we are able to achieve an easy and uniform installation on all our systems.

AUTOMATIC CONFIGURATION BY PUPPET

Puppet is an automated administrative software engine for system configuration. It performs administrative tasks, such as adding users, installing packages, and updating server configurations, based on a centralized specification [3].

The Controls system administrators receive daily requests from users for new consoles, servers or office computers. Our existing 500 computers also need to be maintained. Without automated scripts, this would be almost impossible. The SL kickstart and subsequent puppet configuration is central to our automated installation process.

Each facility needs several types of computers, which we call classes. There are about 35 computer classes defined in puppet. The reason is that different classes of computer need different NFS mounts, environmental variables setup, services and cron tasks or additional local user administration and user accounts. A set of puppet configuration files stored in the central PSI repository handle these particular installations issues. Each computer configured by puppet is assigned a class and each class has a hierarchy of configuration files. An example of a configuration file for an IOC boot server configuration can be seen on Fig.1.



```
class class_A_BootPC_VM {

  info "PUPPETSERVERINFO: Configuring host $hostname, role $role, zone $gfaFacility:

  # GFA basics
  include "class_GFA_DesktopBasic"

  # Intranet AFS
  include "class_module_gfa_afs::intranet"
  include "class_module_gfa_files::mkhomedir_skel"
  include class_module_gfa_pam2::setup
  def_module_gfa_pam2::alternative{"GFA-local-noafs-krb5": activate => true, }

  include "class_module_gfa_runlevel::id3"
  include "class_module_gfa_iocLogServer"
  include "class_module_gfa_autoSR"

  # Accounts
  include "class_module_gfa_shellenv::sshkeys"
  include "class_module_gfa_account::ioc"

  include "class_module_gfa_vmware"

  include "class_module_gfa_epics::iocsh"

  # BootPC stuff
  include "class_module_gfa_bootpc"
  include "class_module_gfa_nfs::server"
  include "class_module_gfa_EpicsRecDB"
  include "class_module_gfa_dgrp"


  case $gfaFacility {
    SLS    : {
      def_module_gfa_bootpc::ioc_dir{"/ioc": }
    }
    FIN    : {
      def_module_gfa_bootpc::ioc_dir{"/ioc": mount => "/import_fin/ioc" }
    }
    HIPA    : {
      def_module_gfa_bootpc::ioc_dir{"/ioc": mount => "/import_hipa/ioc" }
    }
    Proscan : {
      def_module_gfa_bootpc::ioc_dir{"/ioc": mount => "/import_proscan/ioc" }
    }
    TRFCB : {
      def_module_gfa_bootpc::ioc_dir{"/ioc": mount => "/import_trfcb/ioc" }
    }
  }
}
```

Figure 1: An example of a puppet configuration file for a boot server virtual computer. The list of include commands in the first part shows the required configuration for this particular computer type. The case statement similar to the C code syntax in the second part shows how special treatment for each facility is done.

The configuration files are used when the puppet update process is scheduled from a cron-job (or manual request). Fig.2 illustrates a schematic flowchart diagram of the puppet deployment which can be factorized into the following steps:

1. The puppet client sends a request for an update. The "client identity", i.e. the computer name, or class is supplied.
2. The puppet server collects configuration information from the hierarchy of configuration files.
3. The puppet server sends packages and executes the update scripts defined in the configuration files.
4. The client gets the yum updates from the yum repository server.
5. The configuration results are stored in the Oracle database.

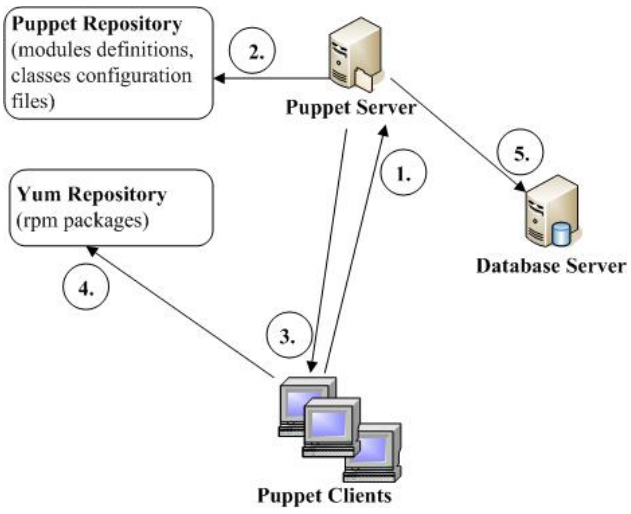
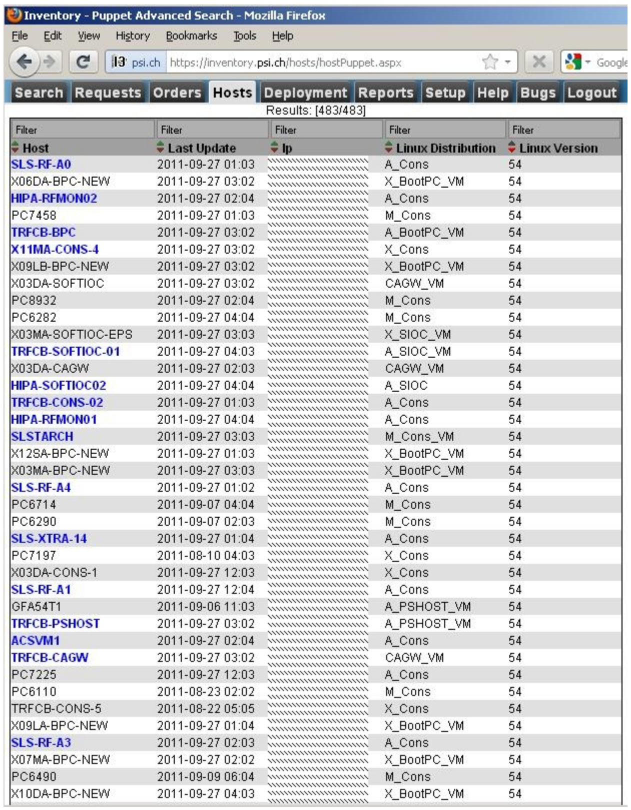


Figure 2: The puppet process configuration of the controls computers.

CONFIGURATION COMPUTERS OVERVIEW

The puppet configuration process is scheduled every 24 hours, typically during the night. Once the puppet configuration process is finished, the results are collected in the Oracle database. At the end of the process a database Web tool called Inventory software [4] integrates complete information about the configured hosts. This is a significant help for the system administrators. Fig.3 shows a database output list of all computers configured by puppet with supplementary information such as last update time, IP-address and Linux version. Finally Fig.4 displays even more detailed information for a host selected from Fig. 3.



Inventory - Puppet Advanced Search - Mozilla Firefox

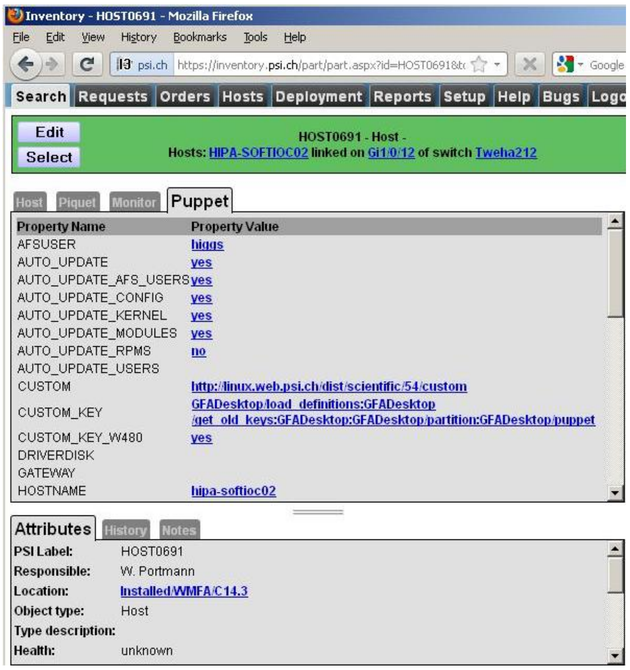
https://inventory.psi.ch/hosts/hostPuppet.aspx

Results: [483/483]

Host	Last Update	Ip	Linux Distribution	Linux Version
SLS-RF-A0	2011-09-27 01:03	[redacted]	A_Cons	54
X06DA-BPC-NEW	2011-09-27 03:02	[redacted]	X_BootPC_VM	54
HIPA-RFMON02	2011-09-27 02:04	[redacted]	A_Cons	54
PC7458	2011-09-27 01:03	[redacted]	M_Cons	54
TRFCB-BPC	2011-09-27 03:02	[redacted]	A_BootPC_VM	54
X11MA-CONS-4	2011-09-27 03:02	[redacted]	X_Cons	54
X09LB-BPC-NEW	2011-09-27 03:02	[redacted]	X_BootPC_VM	54
X03DA-SOFTIOC	2011-09-27 03:02	[redacted]	CAGW_VM	54
PC8932	2011-09-27 02:04	[redacted]	M_Cons	54
PC6282	2011-09-27 04:04	[redacted]	M_Cons	54
X03MA-SOFTIOC-EPS	2011-09-27 03:03	[redacted]	X_SIOC_VM	54
TRFCB-SOFTIOC-01	2011-09-27 04:03	[redacted]	A_SIOC_VM	54
X03DA-CAGW	2011-09-27 02:03	[redacted]	CAGW_VM	54
HIPA-SOFTIOC02	2011-09-27 04:04	[redacted]	A_SIOC	54
TRFCB-CONS-02	2011-09-27 01:03	[redacted]	A_Cons	54
HIPA-RFMON01	2011-09-27 04:04	[redacted]	A_Cons	54
SLSTARCH	2011-09-27 03:03	[redacted]	M_Cons_VM	54
X12SA-BPC-NEW	2011-09-27 01:03	[redacted]	X_BootPC_VM	54
X03MA-BPC-NEW	2011-09-27 03:03	[redacted]	X_BootPC_VM	54
SLS-RF-A4	2011-09-27 01:02	[redacted]	A_Cons	54
PC6714	2011-09-07 04:04	[redacted]	M_Cons	54
PC6290	2011-09-07 02:03	[redacted]	M_Cons	54
SLS-XTRA-14	2011-09-27 01:04	[redacted]	A_Cons	54
PC7197	2011-08-10 04:03	[redacted]	X_Cons	54
X03DA-CONS-1	2011-09-27 12:03	[redacted]	X_Cons	54
SLS-RF-A1	2011-09-27 12:04	[redacted]	A_Cons	54
GFA54T1	2011-09-06 11:03	[redacted]	A_PSHOST_VM	54
TRFCB-PSHOST	2011-09-27 03:02	[redacted]	A_PSHOST_VM	54
ACSVM1	2011-09-27 02:04	[redacted]	A_Cons	54
TRFCB-CAGW	2011-09-27 03:02	[redacted]	CAGW_VM	54
PC7225	2011-09-27 12:03	[redacted]	A_Cons	54
PC6110	2011-08-23 02:02	[redacted]	M_Cons	54
TRFCB-CONS-5	2011-08-22 05:05	[redacted]	X_Cons	54
X09LA-BPC-NEW	2011-09-27 01:04	[redacted]	X_BootPC_VM	54
SLS-RF-A3	2011-09-27 02:03	[redacted]	A_Cons	54
X07MA-BPC-NEW	2011-09-27 02:02	[redacted]	X_BootPC_VM	54
PC6490	2011-09-09 06:04	[redacted]	M_Cons	54
X10DA-BPC-NEW	2011-09-27 04:03	[redacted]	X_BootPC_VM	54

File Edit View History Bookmarks Tools Help
psi.ch
Google
Search Requests Orders Hosts Deployment Reports Setup Help Bugs Logout
Filter

Figure 3: List of controls computers configured by puppet.



Inventory - HOST0691 - Mozilla Firefox

https://inventory.psi.ch/part/part.aspx?id=HOST0691&t

HOST0691 - Host -
Hosts: HIPA-SOFTIOC02 linked on Gi1/0/12 of switch Tweha212

Property Name	Property Value
AFSUSER	higgs
AUTO_UPDATE	yes
AUTO_UPDATE_AFS_USERS	yes
AUTO_UPDATE_CONFIG	yes
AUTO_UPDATE_KERNEL	yes
AUTO_UPDATE_MODULES	yes
AUTO_UPDATE_RPMS	no
AUTO_UPDATE_USERS	
CUSTOM	http://linux.web.psi.ch/dist/scientific/54/custom
CUSTOM_KEY	GFADesktop/load_definitions:GFADesktop /get_old_keys:GFADesktop:GFADesktop/partition:GFADesktop/puppet
CUSTOM_KEY_W480	yes
DRIVERDISK	
GATEWAY	
HOSTNAME	hipa-softioc02

PSI Label:	HOST0691
Responsible:	W. Portmann
Location:	Installed/WMFA/C14.3
Object type:	Host
Type description:	
Health:	unknown

File Edit View History Bookmarks Tools Help
psi.ch
Google
Search Requests Orders Hosts Deployment Reports Setup Help Bugs Logo
Edit
Select
Host Piquet Monitor Puppet
Attributes History Notes

Figure 4: Detailed puppet configuration information for HIPA-SOFTIOC02 selected from Fig.3, visible on the PSI intranet Web.

ACKNOWLEDGMENTS

We want to acknowledge our colleagues M. Gasser, V. Markushin and H. Billich from the PSI AIT group, for their collaboration and for supporting the controls system administration.

REFERENCES

[1] http://www.scientificlinux.org/
[2] http://www.hpc-ch.org/wp/wp-content/uploads/2010/06/KS_Puppet_VM_20100520_printout.pdf
[3] http://projects.puppetlabs.com/
[4] http://gfa-it.web.psi.ch/invent_help/