Patagonia CloneSys – A Tool to Install Multi-Boot Environments

Christian Kurmann, Felix Rauch, Thomas M. Stricker
Laboratory for Computersystems
ETH Zurich, Switzerland

Revised 24 November 1999, Christian Kurmann

1 Introduction

The Patagonia CloneSys is designed to provide easy system installations on multi-boot and multi-purpose Clusters of PC's. Do keep it easy and circumvent the development of a proprietary system it is based on standard UNIX tools and can easily be ported to other flavours of UNIXes. The scripts and ideas are subject to change and are provided as they are without any support.

2 Cloning: Basic Ideas

Everybody knows the drudgery involved in manually setting up and rolling out new PCs, updating existing PCs, and recovering failed PCs. CloneSys makes a big dent in the time required to conduct these jobs. It first creates an exact image of a PC's hard drive, effectively taking a "snapshot" of all the files — hidden, visible, and active — that make up the operating system, applications, and configuration settings. The image can then be copied to any number of PCs, thereby creating completely identical installations. Moreover, it can be copied to many PCs simultaneously. This process it completely operating system independent which makes the system very easy and fast. As the raw disk data is copied, no file system initialization is needed, even the partitioning is implicitly done with a total clone. To support multi-boot environments the installation of single partitions is supported as well.

To keep the system even easier as other commercially available tools we based it on freely available UNIX tools and Linux. For an initial boot of an uninstalled machine we use *muLinux* [1] which is a minimalistic, but mostly complete, script- based Linux distribution that fits on a single 1722k floppy disk. It includes many basic system functions, such as Ethernet support, NFS, Samba, FTP, DHCP etc which is all we need. Further we install a small Linux distribution permanently on the harddisk which lets us update the cluster very fast by simply booting this Linux on all machines and remotely executing our cloning scripts.

Locking mechanisms on partitions allow to install special setups on a part of the cluster preventing the image to be overwritten by other the clone or restore procedure.

[1] http://mulinux.nevalabs.org/

3 Initial Multi-Boot Installation

The first step before we are able to clone a setup is to provide an initial setup on one of the cluster machines. The idea is to use this machine as the clone master. Partitioning, installation of the different OSes with the selected software as well as configuring of the booting procedure and the security setup is done only once on this master machine. After that it can be copied to the whole cluster.

3.1 Partitioning for Multi-Boot Setup

For the initial installation we partitioned our 9 GB harddisks into a small 20 MB partition holding the boot-manager, two Windows NT partitions for educational use (one in US English and one localized in German) and a third Windows NT partition without security for research work, 2 GB each. Further, we added a partition for the ETH specific System Oberon which is 100 MB in size. For parallel computation with scientific codes we install Linux into a 1 GB partition for the root file system and a smaller 128 MB swap partition. A 1 GB partition is left as spare partition for future operating systems to be installed later (such as e.g. Solaris, NetBSD or Rhapsody) or for more swap space in scientific codes.

The booting procedure is accomplished by a commercial boot utility named *System Commander*[2]. A startup screen permits the users to select a partition with the desired OS image from a list of options and allows password protection for certain partitions as well as for removable devices such as the floppy or the ZIP drive. For unattended operation of the cluster a selectable default OS is booted after a few seconds (timeout).

3.1.1 How the setup procedure works

As we used standard installation tools for the different OSes we had to figure out in which order such a multi-boot installation can be setup. Especially the installer and bootloader of Windows NT has some restrictions. There may be different procedures but this one we know to work.

First we booted the master machine with the WindowsNT installer and deleted all existing partitions. After that three partitions have to be setup. A small one (16 MByte) for the Boot Commander and a 200 MByte large partition for a small Linux installation (CloneSys). After that the partition in which Windows NT has to be placed can be setup. The installer will then format the boot partition as well as the Windows NT partition and start the installation. After that the system has to be rebooted which won't run because of an error in the installer. At this stage *System Commander* comes to help. Its setup program installs its own master boot record which then starts the boot loader. At the first booting procedure System Commander recognizes the Windows NT installation and attaches it to the list of bootable OSes. With this, Windows NT can be started and the setup can be continued.

After this initial procedure we installed our small Linux system on the 200 MB partition which allowes to setup the rest of the machine.

As we had different Windows NT installations which we wanted to be installed on a C drive we additionally did a "normal" Windows NT installation on a first partition

[2]System Commander, ©V-Communications, *http://www.v-com.com/*

of a clean disk. This image we then cloned to our multi-boot master machine. After the first boot, the drive letters have to be remapped using the NT Disk Administration Tool. First the boot partition (DOS) has to be mapped to X: as it comes up as the C-drive. After that Windows NT can be mapped back to C: again. This allows to clone the same Windows NT master to more than one partition and adjusting the drive letters back to C for all the installations.

We detected a restriction of the Windows NT boot loader which forced us to install bootable Windows NT systems in partition that start before about 4 GByte on the disk.

3.2 Replication of Different OS Images by "Cloning"

Each OS setup used in the cluster is installed once on a master machine. The master is booted with a small service OS and the disk partitions are copied block-wise onto a server that serves as a repository for all current partition images. To replicate the partition to one or more other machines we reverse the route of the transfer from the image-server to the client machines and write the image block-wise back to the several local disks.

Since the original OS image and its copies on the disk are bit-wise identical we call this way of replicating partitions, entire disks or images with OS installations *cloning*. In most cases a freshly cloned OS can not be brought to life with a simple boot but requires some configuration. After a considerable effort we could keep the individual configurations of the operating systems at a minimum and make freshly cloned images bootable. We use a DHCP server on the same Ethernet segment to assign IP addresses and machine names based on the unique Ethernet MAC address built into the primary network interface of each PC.

3.3 Details of the Security Setup

The most important goal of security settings is to avoid any inconvenience by another operating system that is unrelated to the users work and that might not even be known to that user. A second purpose of the security setup is to protect the integrity of the system installation from corruption and to prevent the different users from clobbering each others personal data stored on the servers. In a multi-boot setup multiple levels of security are needed and the different security mechanisms of the different operating systems need to be coordinated.

The first level of this security setup handles the booting procedure. The *System Commander* utility offers menu controlled selection of the operating system of choice and protects research systems and administration setup with a fairly sophisticated password protection scheme, that allows user groups to be defined.

The second level of system security deals with the visibility (or better invisibility) of partitions with other operating systems including the boot partition with its powerful administrative tools. It should remain impossible to modify data on partitions belonging to other operating systems at least for a normal, mortal user. In order to dissuade educational users from exploring and cracking system security, a shareware tool, called *DeviceLock*[3] hides unused partitions and systematically denies access to

[3] DeviceLock for Windows NT, ©SmartLine, *http://www.protect-me.com/dl/*

them at the Windows NT driver level. The same task is accomplished in UNIX systems by retaining mount as a privileged operation and configuring the default mounting tables accordingly.

Since the NTFS 4.0 file system does not yet support basic OS concepts like mount points multi-boot setups have to worry about drive letters. We enlist the device name remapping mechanism of Windows NT and remap the partitions in a way that the partition of the active OS image is always the C: drive. This allows to use the same application links (and shortcuts stored in the user profiles) across all Windows NT installations on the campus. With this trick it is even possible to switch between two different Windows NT systems transparently for an English or German version.

For our education environment a Windows NT Domain Server authenticates users and controls access to the local and remote files in the cluster — for the research environment a Linux server handles all user authentication according to the Sun NIS protocol.

The lightweight Oberon system that is used for the system software courses copies itself onto a RAM disk upon startup and restricts the access to the disks to read-only at the driver level.

4 Installation and Configuration of CloneSys

4.1 Directory Structure

The CloneSys tools are installed on an image server which stores all the partition images. This server has to export these image directories to all the cluster machines via nfs (or samba). We equipped a designated machine (our cluster server) with an additional large disk, which can be mounted on demand.

On our cluster we use the following directory structure:

```
/images/cloneimages/   <partition- and disk-images>
                       ..
        scripts/       <tools and commands>
                       .cloneconfig
                       ..
        logs/          <logfiles and configuration>
                       clone.log
                       imagedesc.txt
                       clustermap.txt
                       ..
        logs/lastlogs/ <last log files of clone>
                       ..
        config/        <cluster configuration machine files>
                       ..
```

4.2 Clone Images

The `cloneimages` subdirectory contains the gzipped partition images. The file names are constructed as follows: `device_imagename.gz`, e.g. sda1_SysCom.gz.

4.3 Scripts

The `scripts` subdirectory contains all the tools as well as the tool configuration file `.cloneconfig` which has to be placed in the same directory as the scripts. The `.cloneconfig` file contains path declarations as well as pointers to the log files. Our configuration looks like this:

```
# Configuration for Clone Sys Scripts
# PATAGONIA CLUSTER ETHZ

# Installation directory of CloneSys
CLONESYS=/images

# Default path where to find images
ipath=$CLONESYS/cloneimages

# Path for config files
configpath=$CLONESYS/config

# Path where to find scripts
scriptpath=$CLONESYS/scripts

# Path for log files
logpath=$CLONESYS/logs

# Log file for all clonesys commands
clonelog=$logpath/clone.log

# Path for lastlog files
lastlog=$logpath/lastlogs

# Image Database File
imagedesc=imagedesc.txt

# lastlog suffixes
suffixrun=running
suffixfinish=finished
clonesuffix=clone_$suffixrun
clonesuffixfinished=clone_$suffixfinish
clonesuffixremote=clone_remote
backupsuffix=backup_$suffixrun
backupsuffixfinished=backup_$suffixfinish
locksuffix=lock
```

```
# Alias for machine name
prefix=rifpc

# Initial Image (Zeroed Partitions and SysCom, CloneSys in one image)
initimage=default      # image specified in config file

# Initial Configuration (must be the same as with initimage)
initconfig=initconfig

# Image Server
imageserver=imageserver.ethz.ch
```

4.4 Hostname Aliases

For easier access to the machines we have defined DNS aliases for all of them. The aliases have all the same prefix, i.e. `rifpc`, followed by a `host_id`. With the `mapname` command these aliases or just the host_ids alone can easily be mapped to the full hostname.

4.5 Log Files

The `logs` subdirectory stores the `clone.log` file which contains log entries of all the clones, backups, locks and unlocks performed with CloneSys commands.

`imagedesc.txt` contains the descriptions of the images stored in `cloneimages`. An entry looks like this:

```
Image  : /images/new_images/sda5_WinNTEduD_SP4.gz
Origin : client1
User   : kurmann
Date   : Tue May 25 12:51:24 CEST 1999
Comment: WindowsNT Educ D with SP4, Xwindows, Netscape 4.6
```

`clustermap.txt` contains a map of the cluster as well as the ip names of the machine. This file is shown by the command `clustermap` just for information.

The `lastlogs` subdirectory contains the last logfiles for all the machines. There are different suffixes used for `clone-`, `backup-`, `lock-` and `error`-logs. These log files can be viewed or deleted with the `lastlogs` command.

The clone- and backup-logs additionally have a suffix `_running` or `_finished` indicating if a process is running or finished. This can be viewed with the `state` command.

4.6 Configuration

The configuration of CloneSys is done in a separate file for all the cluster machines. The config files are located in the `config` subdirectory and have the following structure:

```
partition:path:image:cloned:user:lockedby:reason:
```

For each installed partition there is a partition (or device) identifier, the path, where the image is stored, the image name and the date when the partition was last cloned by the specified user. Additionally there can be entries for locked partitions. The user who has requested the lock and the reason are indicated in the appropriate fields.

We have partitioned our cluster machines as listed in the following table.

Device	Name	Size	Partition
sda1	System Commander	16 MB	Primary, Activ
sda2	Linux CloneSys	200 MB	Primary
sda4			Extended
sda5	Windows NT Education	1.9 GB	Logical in sda4
sda6	ETH Oberon Education	100 MB	Logical in sda4
sda7	ETH Oberon Research	100 MB	Logical in sda4
sda8	Linux Swap	100 MB	Logical in sda4
sda9	Linux	900 MB	Logical in sda4
sda10	Windows NT Research	1.9 GB	Logical in sda4

The appropriate configuration file is given below:

```
partition:path:image:cloned:user:locked:reason:
sda:/images/cloneimages:sda_InitClone.gz::kurmann:kurmann:admin:
sda1:/images/cloneimages:sda1_SysCom.gz:05/21/99:kurmann:kurmann:admin:
sda2:/images/cloneimages:sda2_CloneSys.gz:05/25/99:rauch:rauch:admin:
sda5:/images/cloneimages:sda5_WinNTEduD_SP4.gz:06/01/99:bmc:::
sda6:/images/cloneimages:sda6_OberonEdu_2.3.6.gz:06/01/99:muller:::
sda7:/images/cloneimages:sda7_OberonRes_2.3.6.gz:06/01/99:muller:::
sda8:/images/cloneimages:sda8_LinuxSwap.gz:06/01/99:rauch:::
sda9:/images/cloneimages:sda9_Linux_6.0_UNI.gz:06/01/99:rauch:::
sda10:/images/cloneimages:sda10_WinNTRes.gz:06/01/99:kurmann:::
```

Note the image for the device `sda`. This is the initial clone image which initialized the whole disk and the CloneSys Linux system as described in the next Chapter. This initial clone generates the configuration file for a new machine automatically.

4.7 Generation of an Initial Clone Image

As described in the previous Chapter, the setup of a master machine is the prerequisite for the cloning process.

In addition to all the partition backups, we use an `InitClone` image which is a backup of the whole disk. This allows us to just clone this image onto an emtpy harddisk to initialize the partitioning as well as the boot commander and the configurations in one step, independent of any operating system specific specialities.

To keep this image short we wipe all the partitions with zeros (e.g. with `dd if=/dev/zero of=/dev/sdaX bs=4096`) except the SysCom and the CloneSys partitions which contain the boot commander as well as the small Linux system needed

for cloning the rest of the system. To generate such an image one can use the backup tool or just type something along the line (with paths adjusted):

```
dd if=/dev/sda bs=1024 |gzip -c |dd of=sda_InitClone.gz bs=1024
```

In addition to the generation of this image a configuration file called `initclone` must be written corresponding to the configuration of the machine and the `initclone`-image. This is stored in the config directory. In our cluster this initial configuration looks like this:

```
partition:path:image:cloned:user:locked:reason:
sda:/images/cloneimages:sda_InitClone.gz:::::
sda1:/images/cloneimages:sda1_SysCom.gz::kurmann:kurmann:admin:
sda2:/images/cloneimages:sda2_CloneSys.gz::kurmann:kurmann:admin:
sda5:/images/cloneimages:sda5_WinNTEduD_SP4.gz:::::
sda6:/images/cloneimages:sda6_OberonEdu_2.3.6.gz:::::
sda7:/images/cloneimages:sda7_OberonRes_2.3.6.gz:::::
sda8:/images/cloneimages:sda8_LinuxSwap.gz:::::
sda9:/images/cloneimages:sda9_Linux_6.0_UNI.gz:::::
sda10:/images/cloneimages:sda10_WinNTRes.gz:::::
```

All the partitions are wiped with zeros except the `sda1` and `sda2` which contain the boot commander and the Service-Linux system.

This initial configuration is automatically installed when an initial install is done with `initinstall`. After that it is possible to just use the `restore` script to install all the remaining partitions.

4.8 Initial Clone

As mentioned in the last Chapter we use a minimalistic boot-floppy / boot-Zip-disk combination based on the Linux distribution *muLinux* to boot a new or broken machine. With such a boot disk the empty machine can be booted and brought to the network.

The next steps are just to mount the `/images` directory from the imageserver over the network and to start the `initinstall` script which installes the initial image and configuration file. With this the machine can be booted in the Service-Linux system an the remaining operating systems can be installed with the `restore` or `rclone` scripts.

4.8.1 Example of an initinstall

Boot the machine with the muLinux boot floppy or zipdisk and login as root. Then mount the `/images` directory from the server

```
mount /images
```

or depending on your `/etc/fstab`

```
mount -t nfs imageserver:/images /images
```

(where *imageserver* is the machine which stores the partition images)

Start the installation script:

```
/images/scripts/initinstall hostname yourname
```

This will install the `sda_InitClone.gz` image to the device `sda` on the local machine an generate an initial configuration file for the machine in the `config`-directory (by copying the `initconfig` file).

After that the machine has to be rebooted into the now newly installed CloneSys (Linux) where the rest of the partitions can be installed with `rclone` or `restore`.

Caution! Don't boot uninstalled systems as they could be automatically removed from the boot-menu.

Restoring the partitions means that the same images that are specified in the `initconfig` shall be initialized. To restore one or more partitions on a cluster the `restore` script can be used as follows:

```
restore yourname -d sda5 sda6 sda7 sda8 sda9 sda10 -h hostname
```

This restores the images for all the partitions sda5 to sda10 on the machine with the given hostname. This command can be executed local or on a server.

It calls the `rclone` command for all the specified devices and machines with the `default` image which is the image specified in the config files.

If different images shall be installed the rclone script can be used as follows:

```
rclone sda5 WinNTEducNew yourname hostname
```

which would install the image WinNTEducNew on sda5.

5 Partition Backups

For saving installations and releases as well as to clone images to the cluster, backups to the image server have to be made. Therefore the `backup` and `rbackup` scripts are provided. They perform a backup of the raw device and gzip the data with the following command:

```
dd if=/dev/device bs=1024 |gzip -c |dd of=ipath/iname bs=1024
```

The file names are constructed as follows: `device_imagename.gz`, e.g. sda1_SysCom.gz.

A comment has to be provided for each image that is generated. This will be stored in the image description log which can be displayed with the `imagedesc` command.

5.1 Example of an image backup

```
backup sda5 WinNTEduc yourname "WindowsNT 4.0 SP4"
rbackup sda5 WinNTEduc clientname yourname "WindowsNT 4.0 SP4"
```

This backs up the device sda5 to the imageserver. The image name will be sda5_WinNTEduc.gz and the comment in the image description database "WindowsNT 4.0 SP4". Note that the double quotes ("") are important! The `rbackup` script can be used to back up a remote machine. The only difference to `backup` is the parameter clientname which has to be provided.

6 Partition Installations, Releases and Restores

6.1 Partition Installations and Releases

To install the saved partitions on to cluster machines the `clone` and `rclone` scripts are provided which read the image over the network, gunzip them and install write the data to the specified partition. The script essentially uses the following command:

```
gunzip -c /ipath/iname |dd of=/dev/device bs=1024
```

6.1.1 Example of a clone

```
rclone sda5 WinNTEduc yourname client1
rclone sda5 default yourname 3 4 5
```

The first command will clone the image `sda5_WinNTEduc.gz` to the device sda5 on host client1. After a successful clone it changes the entry in the config file to the appropriate values.

The second command runs the clone script on the machines with the alias rifpc3, rifpc4 and rifpc5, which restores the last installed image defined in the config file of the machine.

6.2 Restores

Restoring partitions on a cluster means that the same image that was installed on the machines shall be reinitialized. To restore one or more partitions on a cluster the `restore` script can be used. It calls the `rclone` command for all the specified devices and machines with the `default` image which restores the last installed image specified in the config files.

6.2.1 Example of a restore

```
restore yourname -d sda6 sda7 -h 10 12 14
```

This restores the images for the partitions sda6 and sda7 on the machines with the aliases rifpc10, rifpc12 and rifpc14.

7 Tools and Commands

The following tools and commands are provided with CloneSys. They are all based on the bash shell.

7.1 clone

Usage:

```
clone <device> <image_name> <user>
```

`clone` clones an image given by its `image_name` to the mentioned `device` on the *local* machine. `image_name` can be set to `default` to restore the last installed image.

7.2 rclone

Usage:

```
rclone <device> <image_name> <user> [<host_ids>]
```

`rclone` is used similar to `clone` but it allowes to clone multiple machines at once. The image `image_name` is installed to `device` on all the machines given by the `host_ids`. Use `clustermap` to show the cluster map with the ids or use the real hostnames instead." With `image_name` set to `default` the last installed image will be restored."

The script also mounts the image directory and finally tests whether the `clone` script could have been started on all the machines. If an error occured on one of the machines, it is displayed and the job is not started on that machine.

7.3 backup

Usage:

```
backup <device> <image_name> <user> <"comment">
```

`backup` generates a backup image of the specified `device` on the image server under the given `image_name`. `user` specifies the user that initiated the backup. The user name will be copied together with the `command` to an image description file which can be showed with `imagedesc`. Please make sure to deliver a comment including release number of the system and changes (i.e. new installed software, patches). The comments have to be set in quotes.

A user must be specified because we usually do all the clonings with a special username and it would be impossible to find out who was responsible for a certain backup.

7.4 rbackup

Usage:

```
rbackup <device> <image_name> <host_id> <user> <"comment">
```

`rbackup` starts the `backup` script on the machine with the given hostname or host_id which generates a backup image of the specified `device` to the image server under the provided `image_name`. `user` specifies the user that initiated the backup. The user name will be copied together with the `comment` to an image description file which can be showed with `imagedesc`. Please make sure to deliver a comment including release number of the system and changes (i.e. new installed software, patches). The comments have to be set in quotes.

The script also mounts the image directory and tests whether the `backup` script could be started on the remote machine. If an error occured it is displayed and the job is not started.

7.5 initinstall

Usage:

```
initinstall <hostname> <user>
```

`initinstall` initializes the local machine named hostname with the actual partitioning given in a special image which contains the whole disk. It wipes the whole disk, writes the partitioning information and the MBR and installes the boot loader as well as a small Cloning-System (Linux) which can be used to install further images. `user` is the user who initialized the disk.

In addition to the initinstall the script initializes the configuration file for the given hostname by copying the corresponding

Caution! This will delete the whole disk!

7.6 restore

Usage:

```
restore <user> -d [<devices>] -h [<host_ids>]
```

`restore` restores the specified `devices` for all the machines given by their `host_ids` or name by calling `rclone` with the `default` image. This allows to clone more than one device with one command. The synchronization before a second cloning process is startet is handled through the `waitfinished` command. So be sure that there are no lastlogs with the `_running` suffix. Use the `lastlogs` command to delete them if there are.

Caution! This will restore all the partitions on all the machines!

7.7 lockimage

Usage:

```
lockimage <device> <hostname> <user> <reason>
```

`lockimage` locks the given `device` for a specified host. `user` and `reason` are copied to the cluster configuration files, indicating who locked a device and why. Make sure that reason is only one word.

This locking scheme is a weak one and is only used to prevent administrators to overwrite each others partitions with special setups.

To unlock use `unlockimage`.

7.8 unlockimage

Usage:

```
lockimage <device> <hostname> <lockuser> <user>
```

`lockimage` unlocks the given `device` locked by `lockuser` for the specified host by cleaning the lock in the configuration file. Make sure to have asked `lockuser` to cancel his lock!

To lock again use `lockimage`.

7.9 state

Usage:

```
state
```

`state` shows all running clone or backup processes as well as a list of the last processes which have finished. It can be used to check whether a cloning process initiated by `rclone` has terminated.

This simply scans the lastlogs directory for occurences of running or finished processes.

7.10 lastlogs

Usage:

```
lastlogs [<host_id>]
lastlogs -c <logfile> [<host_id>]
```

`lastlogs` shows the last log files for the hosts given by their host_ids or hostnames.
The options are:

- `-c` Deletes the specified logfiles containing the following suffixes: `clone`, `backup`, `lock` or `error`. This can be used to delete last log entries of crashed clone or backup processes which did not change their logfiles anymore.

7.11 waitfinished

Usage:

```
waitfinished mailaddress
```

`waitfinished` waits until all clone or backup processes have finished. Optionally writes an email to `mailaddress` to confirm that the cloning has finished.

Like the `state` command this tests the lastlogs directory for occurences of running processes.

7.12 clonesyslog

Usage:

```
clonesyslog
```

`clonesyslog` shows the CloneSys log that contains all logged clone, backup, locking or unlocking operations.

7.13 mapname

Usage:

```
mapname <alias>
```

`mapname` remaps alias names to real hostnames. Allowed aliases are in our case rifpcID (specified in `.cloneconfig`) or just the hosts ID.

7.14 clustermap

Usage:

```
clustermap
```

`clustermap` shows a map of the while cluster. This is just a textfile that has to be changed by hand.

7.15 config

Usage:

```
config <hostname>
```

`config` shows the configuration file of the host given by its id or hostname. In addition it lists all the locked partitions.

7.16 rex

Usage:

```
rex
```

`rex` stands for remote execution and is a script that is used for remote invocation of clone and backup processes. It just executes the command with its arguments which were passed over to rex in the background and logs errors if they occure.

References

[CC99PATA] F. Rauch Ch. Kurmann T. Stricker and Blanca Maria Müller : Patagonia - A Dual Use Cluster of PCs for Computation and Education. *2. Workshop Cluster Computing, Karlsruhe, March 1999.*