Backup, Booten, RAID, LVM, Virtualization

Franz Schäfer

Linux LV, WU Wien

April 23, 2015

Table of contents

Backup: The Problem

- Metadata (Permissions, Timestamps, Symlinks, Hardlinks, Device Files)
- Multiple versions of Files
- Sparse Files
- Easy restore in case of an emergency
- In some cases: encryption

tar - the classic

```
# tar cfpz dip.tgz  diplomarbeit/
# tar tfvz dip.tgz
# cd / ; mount /bla
# tar one-filesystem -c -p -f - . \
| (cd /bla/ && tar xfvp - )
# tar ... | ssh nachbarserver \
"cd /bla/ && tar xfvp - "
# nc -l -p 7777 | tar xfvlp -
# tar ... | nc otherhost 7777
```

tar - incremental backup

- option `--newer` oder `-N` or
- option `--files-from` oder or `-T` and create the list of files with an other program. e.g `find`

```
# find . -ctime -2 >backup.list
# tar -c -T backup.list -z -f backup.tgt
```

encrypting with gpg

```
# tar ... | gpg -c > backup.tgz.gpg|
```

rsync

```
# mount /backup || exit 1
# cd /
# rsync -Hxa  --delete . /backup
# umount /backup
```

rsync: also works remote over ssh or dedicated rsync server

duplicity - encrypted backup

```
# apt-get install duplicity
# duplicity /home/anna/ \ file:///ext/duplicity/
# duplicity /home/anna \
 scp://karl@woanders.at:/bla/
```

Do not forget your passphrase

backends: local, ssh/scp, rsync, ftp

```
# apt-get install duplicity
```

Enterprise Backup

Enterprise features

- central backup for many hosts
- database of backuped files
- managing tape library

Enterprise Backup

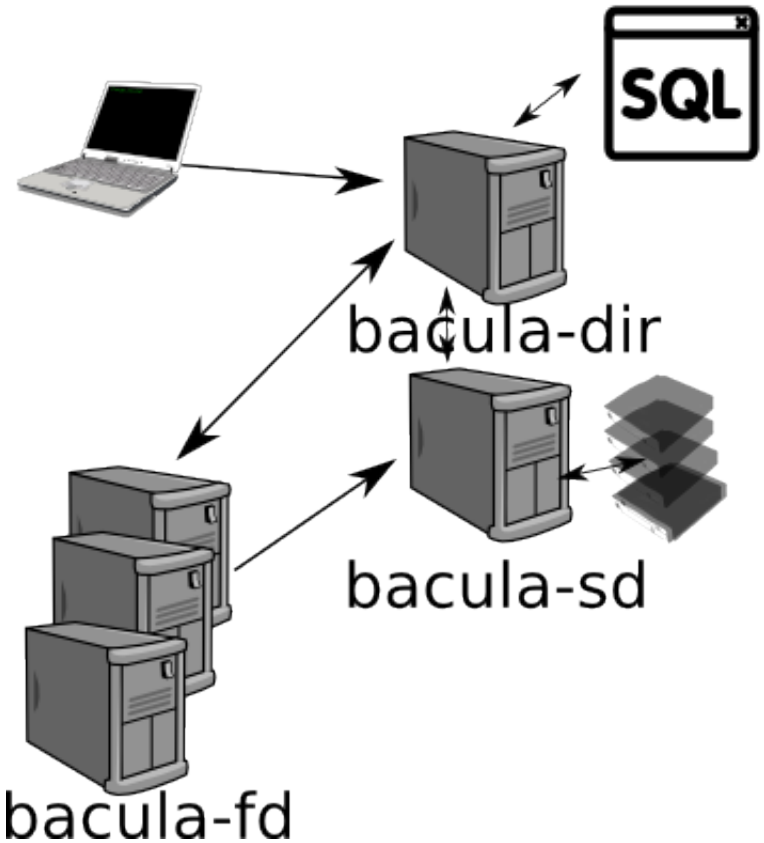
Enterprise features

- central backup for many hosts
- database of backuped files
- managing tape library

Enterprise Backup Systems

- bacula ..... free
- tivoli .... commercial (IBM)

Bacula Overview



bacula-dir **director** controls all other daemons

bacula-fd **file daemon** – daemon that runs on each client

bacula-sd **storage daemon** – actual backup (tape handling, etc, . . .)

cron

backup periodically and automatically. e.g.: via cron.

- cron start processes at certain configurable times. E.g.: each sunday night at 3:17
- system wide configuration is at:

  `/etc/crontab`
- each user can have his/her own crontab:

  `crontab -e`

example crontab

```
53 3 * * * /root/meinbackup.sh >> /var/log/backup.log 2>&1
13 07 * * 0   /root/sonntagmorgens.sh
01,21,41 * * * * /root/3malprostunde
17 */3 * * 1-4   /root/8mal_mo-do.sh
# 0:17 3:17 6:17 .. each monday till thursday
30 7 1-24 12 * /root/advent.sh
```

important `PATH=...` in the system wide crontab there is an additional column that specifies the user.

Backup - Summing up

- keep a backup off-site
- keep older versions
- test a restore once in a while

Backup Hardware

Cheap Solution - 2 External USB Disks

- e.g. change weekly
- keep one off-site
- costs €90/3TB
- keep a backup off-site
- keep older versions
- test a restore once in a while

Backup Hardware

Cheap Solution - 2 External USB Disks

- e.g. change weekly
- keep one off-site
- costs €90/3TB
- keep a backup off-site
- keep older versions
- test a restore once in a while

Enterprise - Tape Library

- tape drive starting €1200
- tape media (LTO5) €25/1.5TB
- tape library starting at €5000

Possible Setup

- rsync to different server in different bilding
- keep old version via btrf snapshots
- additional regular full backup to tapes

Cloning your System with tar

```
# cd /
# tar --one-filesystem -c -p -f - . \
  | netcat otherhost 7777
```

- easy with above tar or with `dd if=/dev/sda`
- do not forget to change hostname
- ip address
- ssh-key
- persistent devices `/etc/udev/rules.d/` (MAC address)

Software RAID

RAID is **not** a Backup

Software RAID

RAID is **not** a Backup

Advantages of Software Raid

compared to hardware raid

- no vendor dependent tools for setup, repair and monitoring
- on a partition by partition basis
- over disks on different controller cards
- not much overhead for RAID1
- with todays CPU speed: even higher RAID levels in software possible

mdadm

```
# mdadm create /dev/md0 \
  -l1 -n2 /dev/sdb7 /dev/sdc7
# mkfs.ext3 /dev/md0
```

After a disk failure:

```
# mdadm --manage /dev/md0 \
 --add /dev/sde5
```

/proc/mdstat

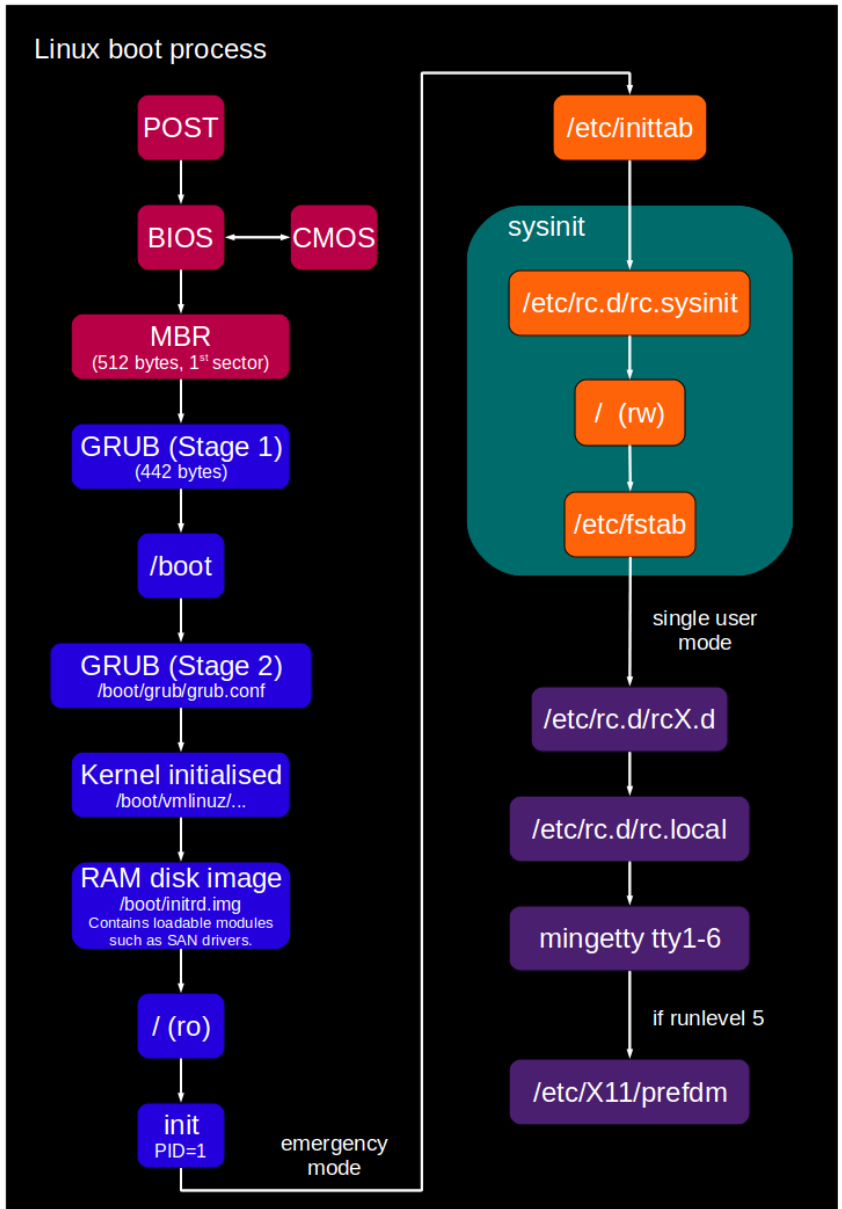
```
# cat /proc/mdstat
Personalities : [raid1]
 md2 : active raid1 sdc3[1] sdb3[0]
           146480576 blocks [2/2] [UU]
 md3 : active raid1 sdc4[1] sdb4[0]
           159252224 blocks [2/2] [UU]
```

config file (important for **monitoring**!)

`/etc/mdadm/mdadm.conf`

Booting Linux

- kernel auf floppy
- lilo
- grub
- loadlin
- syslinux, isolinux, pxelinux
- tftp, nfsroot,
- ...



From: `http://ilostmynotes.blogspot.co.at/2011/01/linux-boot-process-digram.html` (unkown copyright)

The Linux Boot Process

- BIOS starts GRUB from MBR
- GRUB loads kernel and inittrd
- starts kernel with initial RAM-disk
- initrd loads kernel modules and mounts /
- with pivot_root changes to / and init loaded

SysV init vs systemd

- systemd: dependency handling - faster boot times
- systemd: better monitoring of running services
- sysvinit: runlevel vs. systemd targets
- systemsd: cgroups, keeps track of services, builtin logic
- sysvinit simpler

systemd cheatsheet

```
# systemctl list-units
# systemctl restart someservice
# systemclt status someserive
# journalctl
```

config files in /usr/lib/systemd/ /etc/systemd /lib/systemd
start stops scripts in /etc/init.d/ still provided by distributions

How to set a new root password

On the GRUB prompt `ctrl-E` and then use `init=/bin/bash` kernel parameter.

```
# mount -o remount -rw -n /
# passwd
# mount -o remount -r -n /
# sync
# reboot
```

Use Live CD to reset root password

Use rescue or live CD (knoppix, grml, . . .) or similar

```
# sudo bash ; su -
# mkdir /bla
# mount -t ext3 /dev/sda7 /bla
# chroot /bla /bin/bash
# passwd
# exit ; umount /bla
```

Also use it for

- repair boot sector
- emergency backup, restore
- fix hanging boot scripts, ...

GRUB Setup

- initial install: `grub-install /dev/sda`
- directory `/boot/grub`
- debian settings `/etc/default/grub`
- `update-grub` to update configuration

Sample from the generated `/boot/grub/grub.cfg`

```
menuentry 'Debian GNU/Linux, with Linux 2.6.26-1-686' {
        set root='(mduuid/9c28e79e20828a716cd5a85366beffb2)
        linux   /vmlinuz-2.6.26-1-686
        root=UUID=d69d1d3a-4f49-447d-8cd4-b3b56884458c ro s
        initrd  /initrd.img-2.6.26-1-686
}
```

example serial console

in /etc/default/grub

```
GRUB_CMDLINE_LINUX="console=tty0 console=ttyS1,115200n8"
GRUB_TERMINAL=serial   # for both use: =’serial console’
GRUB_SERIAL_COMMAND="serial --speed=115200 --unit=1 --word=
  --parity=no --stop=1"
```

in /etc/inittab

```
T1:23:respawn:/sbin/getty -L ttyS1 115200 vt100
```

Logical Volume Management - LVM

- Flexibilty with Disk Layout
- Snapshots
- based on device-mapper

Terminology:

- PV (physcial volume)
- LV (logical volume)
- VG (volume group)

LVM Example

```
# pvcreate /dev/sda2
# pvcreate /dev/md1
# vgcreate meinvg /dev/hda2\ /dev/sda2 /dev/md1
# lvcreate -L20G -nmp3lv meinvg
# mkfs.ext3 /dev/meinvg/mp3lv
# pvscan
# lvdisplay
```

LVM advanced

Snapshots:

```
# lvcreate -L22G -s -n dbbackup \ /dev/meinvg/datenbank
```

Remove Disk:

```
# pvmove /dev/sda2
# vgreduce meingv /dev/sda2
```

Resize:

```
# lvresize -L +30G /dev/meinvg/mp3lv
# resize2fs ...
```

newer LVM features

- lvmthin - Thin Volumes with cheap snapshots
- lvmcache - Use a fast SSD as cache for slower rotating disks
- raid directly done by LVM/device mapper instead of md driver

Definition of Virtualization

virtualization

In computing, **virtualization** is the creation of a virtual (rather than actual) version of something, such as a hardware platform, operating system (OS), storage device, or network resources.

While a **physical computer** in the classical sense is clearly a complete and actual machine, both subjectively (from the user's point of view) and objectively (from the hardware system administrator's point of view), a **virtual machine** is subjectively a complete machine (or very close), but objectively merely a set of files and running programs on an actual, physical machine (which the user need not necessarily be aware of).


STEELGOHST@HOTMAIL.COM

Types of Virtualization

- CPU **Emulation** - e.g.: VICE (C64 Emulator), QEMU
- Hardware **Virtualization** - e.g.: on native CPU (e.g. KVM, VMware, UML)
- **Containers** (Shared Kernel) - e.g: chroot, BSD Jails, OpenVZ, Linux-VServer (Operating system-level virtualization), LXC (Docker . . .)

Cool Feature: using QEMU to work with chroots of different CPU types.

Type1 vs. Type2 Hypervisor

Type 1 (or native)

runs directly on hardware - e.g. Xen, KVM, VMware ESX/ESXi, Hyper-V

Type2 (hosted)

runs under OS:- e.g. VMware Workstation and VirtualBox, all CPU emulators

Emulation

Microsoft Virtual PC

x86 to x86, powerpc to x86
only runs on windows and osx, limited number of guest OS

QEMU

Host CPU: x86, x86-64, IA-64, PowerPC, SPARC 32/64, ARM, S/390, MIPS
Target CPU: x86, x86-64, ARM, CRIS, LM32, MicroBlaze, MIPS, SPARC 32/64, PowerPC, M68k, Alpha, S/390, Unicore32, SH4, xtensa

KVM, Xen, VMware compared

Xen

Dedicated small Hypervisor, Most Hardware Emulation with support from Dom0

KVM

Fullblown Linux Kernel as Hypervisor, needs assitance from the CPU (only newer CPUs), uses the QEMU Framwork for handling of virtual machines and hardware

VMware-Server

ESX as Hypervisor, Dom0 Linux just for maintenance, Most Hardware drivers integrated in ESX.

Paravirtualization

The guest operating system is aware of the fact that it is running virtualized and helps the host system in its tasks.
e.g.: adding device drivers that are optimized for virtualization.

Why are we using virtualization anyways?

- Efficient hardware utilization: many small servers do not need 100% of CPU all the time
- Administrative isolation
- Secruity barriers
- Redundancy, protection against hardware failure

more topics

- KVM/QEMU hands on
- libvirt handson
- pacemaker
- simulation of networking
- shared storage
- disk formats