Linux System Notes

Useful notes on Linux systems.

Linux Kernel

Kernel-related stuffs.

Intel Motherboard

On startup, got a kernel warning regarding a `firmware bug, TSC_DEADLINE disabled…` or something similar. The required 'fix' is to install `intel-microcode` package.

On **Devuan**, install the 'parent' package - `firmware-linux` (which covers both intel-microcode and amd64-microcode)

```
# apt install firmware-linux
```

Initramfs will be auto regenerated.

On **Slackware**, install `intel-microcode` using script from Slackbuilds.org. Create a new initramfs (initrd) using mknitrd (with `-P /boot/intel-ucode.cpio`).

Kernel Error Message 20210328

After an upgrade on Devuan, I got

```
pstore: crypto_comp_decompress failed, ret = -22!
pstore: decompression failed: -22
```

kernel message after rebooting.

Fixed it by deleting everything in `/sys/fs/pstore`.

Linux LVM

I have some experience playing with AIX and I am very fascinated by its LVM implementation (which is a MUST-USE). I also found out that Linux also has a slightly different LVM implementation - and obviously optional. I tried it on my home machine and I want to put my personal note here. I will simply put it as it is for now… and maybe discuss with a little more detail later.

linux_lvm_notes.txt

```
* DISCLAIMER: this is my personal note. use at your own risk.

* My home machine is running Devuan
- note: mount | grep -v -E "/sys|tmpfs|proc|devpts|gvfs"
- current disk layout on /dev/sda (GPT):
  = p1: UEFI partition (fat32) 127M (1M offset)
  = p2: Root partition (ext4) 30G
  = p3: Home partition (ext4) ~898G
  = p4: Swap partition (swap) 4G

@ TASK: Create a new lv from slice of p3
- make sure basic lvm terms/concepts are covered
  = volume group (vg)
  = physical volume (pv) [aix:pv MUST BE a disk]
  = physical element/extent (pe) [aix:pp=physical partition]
  = logical volume (lv)
  = logical element/extent (le) [aix:lp=logical partition]
- need to unmount /home
  = login as root and make sure no user logged in
- target disk layout:
  = p1: UEFI partition (fat32) 127M (1M offset)
  = p2: Root partition (ext4) 30G
  = p3: Home partition (ext4) 600G
  = p4: Xtra partition (lvm) ~298G
  = p5: Swap partition (swap) 4G
- current p3 content is <500G

* repartition disk to create space for new lv
# umount /home
# e2fsck -f /dev/sda3
# resize2fs -p /dev/sda3 500G
# swapoff /dev/sda4
# gdisk /dev/sda
> delete /dev/sda4
> note /dev/sda3 starting sector
> delete /dev/sda3
> create new /dev/sda3 (linux:8300)
  = make sure use same starting sector
  = specify size +600G
> create new /dev/sda4 (linux lvm:8e00)
  = default starting sector
  = specify -4G to leave space for swap
> create new /dev/sda5 (swap:8200)
  = should have 4G remaining
> exit
* reclaim /dev/sda3 space (600G)
# resize2fs -p /dev/sda3
* edit /etc/fstab accordingly
  - e.g. specify new uuid for swap & /dev/sda3
* restore swap
# swapon -a
```

```
- OR, 'swapon /dev/sda5'

* install lvm
- using lvm2 implementation (uses device mapper)
# apt install lvm2

* create pv
# pvcreate /dev/sda4
* create volume group vg
# vgcreate xtravg /dev/sda4
* playing around (create/remove lv)
# lvcreate -L 100G -n store xtravg
# lvcreate -l +100%FREE -n movie xtravg
# lvremove movie
# lvremove xtravg/store

* create single lv on vg
# lvcreate -l +100%FREE -n share xtravg
# vgdisplay xtravg
* create ext4 fs on lv
# mkfs.ext4 -L LVMSHARE /dev/xtravg/share
* create a mountpoint for the new lv
# mkdir /ext
* check entry in /dev/mapper/
  = should have xtravg-share
* add entry in fstab
  = e.g. /dev/mapper/xtravg-share /ext ext4 defaults 0 2
* manually mount
# mount /ext

@ TASK: Add new disk to existing lv
- got a new disk, using whole disk (1TB)
- extend existing lv

# pvcreate /dev/sdb
# vgdisplay xtravg
# vgextend xtravg /dev/sdb
# vgdisplay xtravg
* use all newly available space
# lvextend -l +100%FREE /dev/xtravg/share

@ TASK: Remove PV from VG
- need to use original /dev/sda4 for something else
- note: pvs -o+pv_used
- PV is in use, so need to free up

# umount /ext
# e2fsck -f /dev/xtravg/share
# resize2fs -p /dev/xtravg/share 800G
# lvreduce -L 850G /dev/xtravg/share
* reclaim space
```

```
# resize2fs -p /dev/xtravg/share
* check usage
# pvs -o+pv_used
- OR, pvdisplay
* IF any PEs still in use
# pvmove /dev/sda4
* remove pv from vg
# vgreduce xtravg /dev/sda4
* remove pv from system (no longer id as pv)
# pvremove /dev/sda4
* reclaim all lv space
# lvextend -l +100%FREE /dev/xtravg/share
# resize2fs -p /dev/xtravg/share
```

Linux AUFS

Dumped from my personal note on making AUFS-patched Linux kernel.

linux_aufs_notes.txt

```
---------------------------------
How I built an aufs-patched kernel
---------------------------------

- get kernel source
  = got 5.4.26 (latest longterm atm)
  $ getlinux 5.4.26 -x
- prepare kernel source
  = extract
  $ tar xf linux-5.4.26.tar.xz
  = actually, i got 5.4.25 like a day before 5.4.26 is release
  = got patch-5.4.25-26 and apply
  $ cd linux-5.4.25
  $ patch -p1 < ~/temp/patch-5.4.25-26
  = renamed the path as linux-5.4.26

- get the required aufsX
  = need aufs5 because i chose kernel v5
  $ git clone git://github.com/sfjro/aufs5-standalone.git
- prepare aufs5
  $ git checkout -b aufs5.4 origin/aufs5.4

- get aufs-util (for later use)
  $ git clone git://git.code.sf.net/p/aufs/aufs-util
- prepare aufs-util
  = aufs-util does not have exact kernel version (get closest lower)
  $ git checkout -b aufs-5.0 origin/aufs5.0
```

```
- apply patches in aufs5-standalone into linux
  = not all patches are required
  = refer to README in aufs5-standalone
  = i use method 1
  $ cd linux-5.4.26
  $ patch -p1 < ../aufs5-standalone/aufs5-kbuild.patch
  $ patch -p1 < ../aufs5-standalone/aufs5-base.patch
  $ patch -p1 < ../aufs5-standalone/aufs5-mmap.patch
  $ patch -p1 < ../aufs5-standalone/aufs5-standalone.patch
  = OR (in a single line command)
  $ for a in kbuild base mmap standalone ; do patch -p1 < ../aufs5-
standalone/aufs5-$a.patch ; done
- need to copy some files
  = still in linux-5.4.26
  $ cp -av ../aufs5-standalone/Documentation/* Documentation/
  $ cp -av ../aufs5-standalone/fs/* fs/
  $ cp -av ../aufs5-standalone/include/uapi/linux/aufs_type.h
include/uapi/linux/
- repack this for easy reuse
  = rename path as linux-5.4.26_aufs
  $ cd .. ; mv linux-5.4.26 linux-5.4.26_aufs
  $ cd linux-5.4.26_aufs
  $ packthis

- prepare to build
  = unpack to /usr/src
  $ cd /usr/src ; tar xf linux-5.4.26_aufs.tar.xz
- configure kernel
  $ cd linux-5.4.26_aufs
  = use make_kernel
  $ make_kernel -i -c
  = that will run make menuconfig
  = select aufs as module
- build kernel
  = use make_kernel
  $ make_kernel -x
  = wait...

- pack kernel (slackware)
  = use pack_kernel (duh!)
  $ pack_kernel --all
  = by default package will be at $HOME/temp
```

From:
https://www.azman.my1matrix.cc/ - **Azman @MY1**

Permanent link:
https://www.azman.my1matrix.cc/doku.php?id=linux:linux_system&rev=1777439821

Last update: **2026/04/29 05:17**