Linux Kernel Update - “from scratch (almost)”

Platform: Archlinux, UEFI, GRUB2, and initramfs

Prof. Rossano Pablo Pinto
FATEC Americana

May/2017 - v0.9

Agenda

- Install development software
- Overview of the steps
- Archlinux 64 bits with Linux Kernel 4.11.2

Install development software

- This step prepares Archlinux to configure and compile Linux (choose `all` options - just press enter to all prompts):

```
pacman -S base-devel bc
```

Overview of the steps

- Download: kernel.org
- Unpack
- Configure
- Compile
- Install
  - copy kernel and modules to appropriate places
  - generate initramfs

Overview of the steps

- Configure GRUB
  - GRUB
- Write UEFI partition (`grub-mkconfig -o /boot/...`)
- Reboot!!!!!

Archlinux 64 bits + Linux 4.11.2

- Download:

https://www.kernel.org/pub/linux/kernel/v4.x/linux-4.11.2.tar.xz

- Unpack

```
tar -xvJf linux-4.11.2.tar.xz
```

- Configure

```
cd linux-4.11.2

make clean && make mrproper

zcat /proc/config.gz > .config
```

Archlinux 64 bits + Linux 4.11.2

- .... config cont.

`make olddefconfig` **OR** `make localmodconfig` (this is faster)

`make menuconfig`

- Change the following items:
  - General Setup
    - Local Version: `-rpp`
    - Default hostname: `admsoredes`
  - Processor type and features

Archlinux 64 bits + Linux 4.11.2

- .... cont.
  - Processor type and features
    - Preemption model
      - Choose **Server** or **Low-latency Desktop**
    - Timer frequency
      - Choose from 100Hz to 1000Hz
        - 100Hz is good for servers, 1000Hz is good for desktops....

Archlinux 64 bits + Linux 4.11.2

- Save
- Compile

```
time make -j 2

make modules_install
```

Archlinux 64 bits + Linux 4.11.2

- Copy files to the apropriate places:

```
cp arch/x86_64/boot/bzImage /boot/vmlinuz-4.11.2-rpp

cp .config  /boot/config-4.11.2-rpp

cp System.map /boot/System.map-4.11.2-rpp
```

- Generate an initramfs image:

```
mkinitcpio -k 4.11.2-rpp -g /boot/initramfs-4.11.2-rpp.img
```

Archlinux 64 bits + Linux 4.11.2

- Change video resolution and remove the “quiet” option of the kernel:

```
nano /etc/default/grub

    …

    GRUB_DISABLE_LINUX_UUID=true #OPTIONAL!!!!

    GRUB_CMDLINE_LINUX_DEFAULT=” ”

    GRUB_GFXMODE=640X480

    …
```

Archlinux 64 bits + Linux 4.11.2

- Disable automatic detection of installed kernels and some other features:

```
chmod 644 /etc/grub.d/10_linux

chmod 644 /etc/grub.d/20_linux_xen

chmod 644 /etc/grub.d/30_os-prober
```

Archlinux 64 bits + Linux 4.11.2

- If you want to use UUID instead of device names (you can skip this otherwise):
  - Discover UUIDs: `lsblk -a -o NAME,FSTYPE,UUID`
  - You must see something like this (take note of the UUID in order to use next):

```
NAME  FSTYPE UUID

sda

|-sda1   vfat      70FE-ab75

|-sda2   swap     aee23956-fce4-460a-87bc-120ba4137f2e9

 `-sda3   ext4     7e033a20-566e-4043-a092-b32a78fa30bd
```

Archlinux 64 bits + Linux 4.11.2

- Create new entries (suppose `/dev/sda1` is EFI partition and `/dev/sda3` is the “/”).
- nano `/etc/grub.d/40_custom`

```
#!/bin/sh
exec tail -n +3 $0
# This file provides an easy way to add custom menu entries. Simply type the menu entries you
# want to add after this comment.  Be careful not to change the 'exec tail' line above.
menuentry 'Arch Linux ORIGINAL' {
        load_video
        set gfxpayload=keep
        insmod gzio
        insmod part_gpt
        insmod fat
        set root='hd0,gpt1'
        search --no-floppy --fs-uuid --set=root 70FE-ab75
        echo     'Loading Linux linux...'
        linux       /vmlinuz-linux root=/dev/sda3 rw
        echo                'Loading initial ramdisk ...'
        initrd  /initramfs-linux.img
}
```

Archlinux 64 bits + Linux 4.11.2

- ...

```
menuentry 'Arch Linux RPP' {

load_video

set gfxpayload=keep

insmod gzio

insmod part_gpt

insmod fat

set root='hd0,gpt1'

search --no-floppy --fs-uuid --set=root 70FE-ab75

echo      'Loading Linux linux RPP...'

linux     /vmlinuz-4.11.2-rpp root=/dev/sda3 rw

echo      'Loading initial ramdisk ...'

initrd  /initramfs-4.11.2-rpp.img

}
```

Archlinux 64 bits + Linux 4.11.2

- Generate a new grub.cfg

```
grub-mkconfig -o /boot/grub/grub.cfg
```

- Reboot: `reboot`
- At GRUB screen, choose the new kernel
- Follow-up the msgs of the new kernel
- Login into the new system
- Verify the new kernel version:

```
uname -a
```

- That's it! Enjoy!!!

Archlinux 64 bits + Linux 4.11.2

- Challenge:

Customize the kernel for the machine in which it is installed:

  - Remove all the device drivers that are not going to be used
  - Choose the processor model that represents the machine to be used (check which processor is installed at the machine).

THE END

Paralell compilation with distcc

CLIENT (a single machine):

```
export DISTCC_HOSTS=" \
        localhost 192.168.80.101 192.168.80.103 192.168.80.94"
time make CC=" distcc gcc"  -j8
```

SERVERS (several machines):

```
distccd --allow 192.168.80.0/24 127.0.0.1
```

For the impatient

- pacman -S base-devel bc
- Download/extract kernel... ... zcat /proc/config.gz > .config
- **make localmodconfig (just press enter at every question - just a few)**

  OR
- **make olddefconfig**
- make menuconfig (config everything you need, save and, quit)
- make -j 2
- make modules_install
- cp arch/x86_64/boot/bzImage /boot/vmlinuz-4.11.2-rpp
- mkinitcpio -k 4.11.2-rpp -g /boot/initramfs-4.11.2-rpp.img
- Create grub entry and then type the following command
- grub-mkconfig -o /boot/grub/grub.cfg

Documentation

- https://wiki.archlinux.org/index.php/GRUB
- https://github.com/torvalds/linux/blob/master/Documentation/admin-guide/README.rst