Live-USB-Stick erstellen mit Trick 17

Livesysteme aus virtuellen Maschinen erzeugen

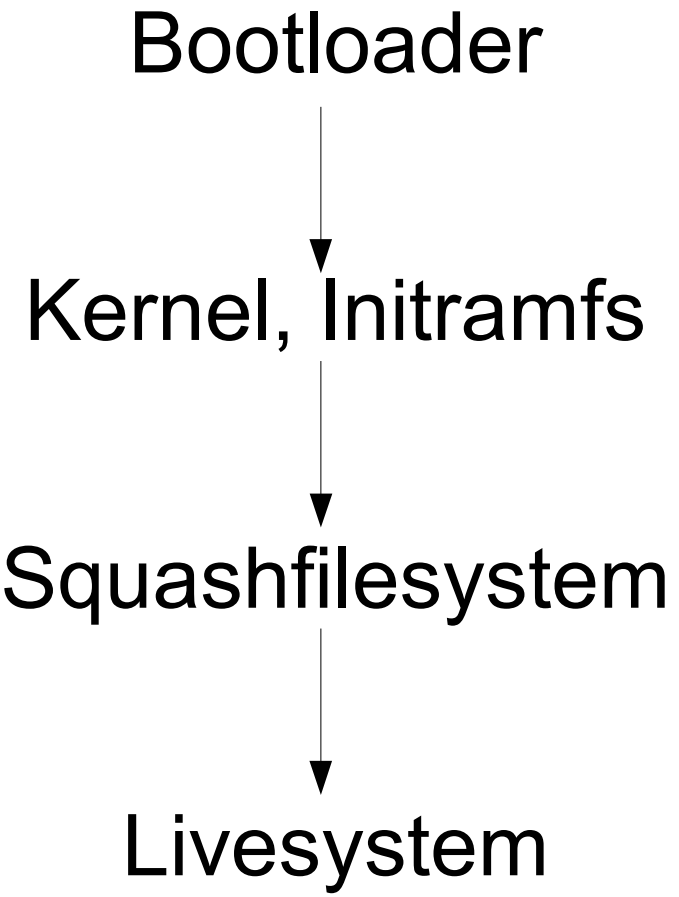
CLT-Workshop, 20. März 2016

Ersteller: Thomas Laube

Gründe:

- festgelegtes Umgebung für bestimmten Zweck schaffen
- Livesysteme nach persönlichen Vorlieben erstellen
- Einsatz von selbst kompilierter Software
- Ausprobieren von Software/Konfigurationen
- Weil es geht!!1!

Wie funktioniert ein Livesystem?



Vorarbeiten in der VM

- Aktualisierungen vornehmen
- ggfs. Fremdquellen hinzufügen
- gewünschte Software installieren
- Nutzer anlegen
- System konfigurieren

Vorarbeiten in der VM

Paketzwischenspeicher löschen

- Debian-basierende Distributionen:

  # apt-get clean

- Pacman-basierende Distributionen:

  # pacman -Scc

Vorarbeiten in der VM

- notwendige Programme installieren, um das System als Liveboot nutzen zu können

  Debian/Ubuntu:

  # apt-get install live-boot live-boot-initramfs-tools squashfs-tools

  Arch Linux:

  # pacman -S archiso

Vorarbeiten in der VM

- Initramfs neubauen

  Debian/Ubuntu

  # update-initramfs -u

  Arch Linux:
- Vorbereitung:

  # mkdir /boot/x86_64

  # cp /usr/lib/initcpio/hooks/archiso /usr/lib/initcpio/hook

  # cp /usr/lib/initcpio/install/archiso /usr/lib/initcpio/install

  Eine Datei /etc/mkinitcpio-archiso.conf erstellen mit dem Inhalt

  HOOKS="base udev archiso modconf block filesystems keyboard"

  COMPRESSION="xz"
- Neubau:
- # mkinitcpio -c /etc/mkinitcpio-archiso.conf -k /boot/vmlinuz-live -g /boot/x86_64/archiso.img

Vorarbeiten im Hostsystem:

- squashfs-tools installieren
- libguestfs-tools installieren
- syslinux installieren
- für ext2/3/4-Partitionen extlinux installieren

USB-Stick vorbereiten

- Partition auf dem USB-Stick erstellen und mit Dateisystem ext2/3/4 versehen
- die Partition mit Bootflag versehen

  # parted /dev/sdb set 1 boot on
- Syslinux in den MBR des Sticks schreiben

  Syslinux 4.x

  # dd bs=440 count=1 if=/usr/lib/syslinux/mbr.bin of=/dev/sdb

  Syslinux 6.x (Debian/Ubuntu)

  # dd bs=440 count=1 if=/usr/lib/syslinux/modules/bios/mbr.bin of=/dev/sdb

  Syslinux 6.x (Arch Linux)

  # dd bs=440 count=1 if=/usr/lib/syslinux/bios/mbr.bin of=/dev/sdb
- Partitionslabel vergeben (nur notwendig bei Arch-Linux-basierenden Distributionen)

  tune2fs -L LABEL /dev/sdb1

USB-Stick vorbereiten

- **Partition einbinden, Ordner für den Bootloader erstellen, extlinux installieren**

  # mount /dev/sdb1 /mnt

  # mkdir -p /mnt/boot/syslinux

  # extlinux --install /mnt/boot/syslinux

- **Syslinux 4.x**

  # cp -r /usr/lib/syslinux/*.c32 /mnt/boot/syslinux/

- **Syslinux 6.x (Debian/Ubuntu)**

  # cp -r /usr/lib/syslinux/modules/bios/*.c32 /mnt/boot/syslinux/

- **Syslinux 6.x (Arch Linux)**

  # cp -r /usr/lib/syslinux/bios/*.c32 /mnt/boot/syslinux/

Daten aufbereiten

- Arbeitsordner erstellen

  # mkdir -p /media/virtual

  # mkdir tmplive
- Virtuelle Festplatte einbinden

  # guestmount -a /PFAD_ZUR_DATEI -i /media/virtual
- Daten in Arbeitsverzeichnis kopieren

  # cp -a /media/virtual/* tmplive/
- Virtuelle Festplatte aushängen

  # guestunmount /media/virtual
- fstab für das Livesystem anpassen

fstab anpassen

- Debian/Ubuntu

  # echo -e "overlayfs / overlayfs rw 0 0\ntmpfs /tmp tmpfs nosuid,nodev 0 0" > tmplive/etc/fstab

- Arch Linux Distributionen

  # echo "# " > tmplive/etc/fstab

Squashfilesystem erstellen

- Debian/Ubuntu

  # mksquashfs tmplive/. filesystem.squashfs -comp xz

- Arch-Linux-Distributionen

  # mksquashfs tmplive/. airootfs.sfs

Und zum Abschluss (Debian/Ubuntu):

- einen Ordner live auf der Zielpartition anlegen

  # mkdir /mnt/live

- das Squashfs dorthin kopieren

  # cp filesystem.squashfs /mnt/live/

  # sync

- Kernel und initramfs auf den Stick kopieren

  # cp ~/tmplive/boot/KERNEL_NAME /mnt/vmlinuz

  # cp ~/tmplive/boot/INITRAMFS_NAME /mnt/initrd

- Syslinux-Konfigurationsdatei erstellen

Und zum Abschluss (Arch-Linux-Distributionen):

- notwendige Verzeichnisstruktur anlegen

  # mkdir -p /mnt/arch/{boot/x86_64,x86_64}

  # echo "airootfs / x86_64 xz ext4 50%" > /mnt/arch/aitab

- das Squashfs ans Ziel kopieren

  # cp airootfs.sfs /mnt/arch/x86_64/

- Kernel und initramfs ans Ziel kopieren

  # cp tmplive/boot/vmlinuz-live /mnt/arch/boot/x86_64/vmlinuz

  # cp tmplive/boot/x86_64/archiso.img /mnt/arch/boot/

- Syslinux Konfigurationsdatei erstellen

Syslinux-Konfiguration Debian/Ubuntu

- Konfigurationsdatei erstellen

  # nano /mnt/boot/syslinux/syslinux.cfg

- Inhalt einfügen

  default menu.c32

  prompt 0

  menu title SYSLINUX

  timeout 100

  label default

  menu label Livesystem

  kernel /vmlinuz

  append initrd=/initrd boot=live

Syslinux-Konfiguration Arch-Linux-Distributionen

- Konfiguratioinsdatei erstellen

  # nano /mnt/boot/syslinux.cfg

- Inhalt einfügen

```
DEFAULT menu.c32
PROMPT 0
MENU TITLE Trick17
TIMEOUT 300


LABEL arch
MENU LABEL Trick17
LINUX /arch/boot/x86_64/vmlinuz
INITRD /arch/boot/x86_64/archiso.img
APPEND archisolabel=Trick17


ONTIMEOUT arch
```

Quellen:

- http://tldp.org/HOWTO/SquashFS-HOWTO/creatingandusing.html
- http://www.ibm.com/developerworks/linux/library/l-initrd/index.html
- http://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/tree/Documentation/filesystems/ramfs-rootfs-initramfs.txt?id=HEAD
- http://debian-live.alioth.debian.org/live-manual/stable/manual/html/live-manual.en.html
- https://github.com/djgera/archiso
- http://www.syslinux.org/wiki/index.php?title=SYSLINUX
- die manpages (und teilweise der Code) der genutzten Software. ;-)