USB Multi-boot

Boot multiple Linux distros with a hybrid UEFI/legacy grub and have both a data and encrypted partition for storage.

LXF ref

Original script from Linux Format by "John Lane" modified for my own purposes.

Linux Format Issue 246

Use cases

- Diagnostics/Testing equipment (system rescue)
- Backup (clonezilla/system rescue)
- Fixing grub (supergrub/rescatux)
- Security testing (kali)
- Privacy (tails)
- Installation (debian/mx linux for old hardware)

Pros

- Carry around one USB does everything
- No Optical media
- Easy to upgrade

Cons

- Not point and click
- Takes time to add new systems
- Can override system if not careful

How does it work

1. Calculates partition sizes

GPT, BIOS and EFI partition sizes and placement is important. DATA & ENCRYPTION use the rest of the data split in half

2. Make the partitions

Script uses sgdisk (partitions) and gdisk (Hybrid)

On a 16GB drive I ended up with

1MB	BIOS Boot
50MB	EFI
6GB	boot
4GB	data
4GB	TailsData (encrypted)

3. Formats and Encrypts

Formats using vfat for EFI & data

```
mkfs.vfat
```

EXT4 for boot area (where the ISO's are kept)

```
mkfs.ext4
```

Encrypts with cryptsetup luksFormat

```
cryptsetup luksFormat ${dev}5
```

4. Copy ISOs

Use rsync incase you have to start the copy again

Make sure we sync before unmount

5. Install grub twice

Once for legacy using

```
grub-install --target=i386-pc --recheck --boot-directory="$boot" $dev
```

A second time for UEFI

```
grub-install --target=x86_64-efi --recheck --removable --efi-directory="$efi" --boot-directory="$
```

6. Test

Test using qemu instead of rebooting all the time.

```
qemu-system-x86_64 -vga cirrus -machine accel=kvm -m 2G -hdb $dev
```

Grub config

"GNU GRand Unified Bootloader"

- Took a long time
- Groups of config (Debian, Red Hat, Arch, Gentoo)

Initialise

Load video drivers and font

```
insmod all_video
font_path=/grub/unicode.pf2
loadfont unicode
```

Setup boot path variable

```
imgdevpath='/dev/disk/by-partlabel/boot'
export imgdevpath
```

Debian based config

Example of booting the Debian LIVE ISO

```
menuentry 'Debian 9.8 LIVE (64bit)' {
  set isofile="/iso/debian-live-9.8.0-amd64-xfce.iso"
  loopback loop $isofile
  linux   (loop)/live/vmlinuz-4.9.0-8-amd64 fromiso=$imgdevpath/$isofile boot=live components spl
  initrd  (loop)/live/initrd.img-4.9.0-8-amd64
}
```

NOTE: This is similar for Kali, Linux Mint, Clonezilla, MX Linux, Tails and Rescatux!

Gentoo based config

Old System Rescue CD, but the easiest

```
menuentry 'System Rescue CD 4.8.3 (32/64bit)' {
  isofile="/iso/systemrescuecd-x86-4.8.3.iso"
  loopback loop $isofile
  linux (loop)/isolinux/rescue32 isoloop=$isofile
  initrd (loop)/isolinux/initram.igz
}
```

NOTE: No fromiso, boot, components or splash on the linux line and different directories.

Arch based config

New System Rescue CD 6.x

```
menuentry 'System Rescue CD 6.0.2 (Arch) (32/64bit)' {
  isofile="/iso/systemrescuecd-6.0.2.iso"
  loopback loop $isofile
  linux (loop)/sysresccd/boot/x86_64/vmlinuz img_dev=$imgdevpath img_loop=$isofile earlymodules=l
  initrd (loop)/sysresccd/boot/intel_ucode.img (loop)/sysresccd/boot/amd_ucode.img (loop)/sysresc
}
```

NOTE: Lots of stuff is different here, earlymodules, img_{loop}, img_{dev}, archisodevice, archisobasedir.

Grub Gotchas

- System Rescue CD 4.x was Gentoo, now it's Arch. The config is very different.
- Same with Kali, it used to be based on Knoppix (when called Backtrack) now it's based on Debian Testing.
- SuperGrub is just grub menus so I had to create a nested grub menu item which calls Supergrubs grub.
- Finding the boot line options is hard.

Commands used

sgdisk zap-all

Destroy any old partitioning information to start clean

```
sgdisk --zap-all $dev
```

read

- Ask a question and store the answer in partitions

```
read -n1 -p "$cr Create partitions [y,n]?" partitions
```

-n1	Read after one character and return
-p <message>	Prompt with message

NOTE: $cr is a Carriage Return defined as, (it's cheating but it works)

```
cr=`echo $'\n.'`
cr=${cr%.}
```

mktemp

- Create a temp dir called tmp.XXXXXXX these directories are removed after the script

```
mktemp -d
```

-d	make directory (default: file)

sgdisk GPT partitioning

```
sgdisk -o -n 2:$EFI_START:$EFI_END -t 2:EF00 -c 2:'EFI System Partition' \
       -n 3:$BOOT_START:$BOOT_END -t 3:8300 -c 3:boot \
       -n 4:$DATA_PLAIN_START:$DATA_PLAIN_END -t 4:0700 -c 4:DATA \
       -n 5:$DATA_CRYPT_START:$DATA_CRYPT_END -t 5:8301 -c 5:TailsData \
       -n 1:$BIOS_START:$BIOS_END -t 1:EF02 -c 1:'BIOS Boot Partition' \
       /$dev
```

Option	Description
-o	Clear partition data
-n <partnum>	New GPT partition given a number
-t <hexcode>	Type of file system given a hexcode
-c <partnum:name>	Set a name for the partition

gdisk

- Adds Hybrid MBR (better than sgdisk at this task so LXF article says)
- Problem: It's a menu driven command only so you have to sendkeys to the command

```
    echo "Hybrid MBR partitioning (uses sed to pipe de-commented here-script into gdisk)"
    <<EOF sed -e 's/^ *\([^#]*\) \+# .*$/\1/' | gdisk $dev && printf ', hybrid MBR'
      r    # recovery and transformation options
      h    # make hybrid MBR
      4    # add data partition
      N    # do not place EFI GPT (0xEE) partition first
      0c   # partition type (Win95 FAT32 LBA)
      N    # don't set the bootable flag
      Y    # additional protective partition
      EE   # additional protective partition type
      w    # write table to disk and exit
      Y    # proceed, possibly destroying data
EOF
```

mkfs.*

- Makes the filesystem

Make MS VFAT system for EFI & Data as it's compatible with the most OS's

```
mkfs.vfat
```

Linux filesystem for boot area where the ISO files are kept

```
mkfs.ext4
```

e2label

- Label EXT4 systems

```
e2label ${dev}3 boot
```

dosfslabel

- Label DOS partitions

```
dosfslabel ${dev}4 DATA
```

cryptsetup

- Set initial passphrase

```
cryptsetup luksFormat ${dev}5
```

- Open LU KS device for use (formatting & mounting in our case)

```
cryptsetup open ${dev}5 $dm
```

rsync

- Copy OS's with ability to resume.

```
rsync -Pav systemrescuecd-6.0.2.iso \
      systemrescuecd-x86-4.8.3.iso \
      kali-linux-xfce-2019.1-amd64.iso \
      MX-18.1_386.iso \
      rescatux-0.51b3.iso \
      super_grub2_disk_hybrid_2.02s10.iso \
      debian-live-9.8.0-amd64-xfce.iso \
      clonezilla-live-2.6.0-37-amd64.iso \
      tails-amd64-3.12.1.iso \
      linuxmint-19.1-xfce-64bit.iso $boot/iso
```

-a	archive (keep permissions)
-P	show progress
-v	verbose

grub

- Install Legacy grub

```
grub-install --target=i386-pc --recheck --boot-directory="$boot" $dev
```

- Install UEFI grub

```
grub-install --target=x86_64-efi --recheck --removable --efi-directory="$efi" --boot-directory="$
```

–target	platform
–recheck	delete existing device map
–removable	EFI only: For removable drives such as USB
–efi-directory	This is set to the device EFI partition (partnum 2)
–boot-directory	Specify a directory to install grub (default: /boot/grub)

qemu

- System emulator which boots 64bit images instead of rebooting

```
qemu-system-x86_64 -vga cirrus -machine accel=kvm -m 2G -hdb $dev
```

-vga cirrus	Use Cirrus CLGD 5446 PCI VGA card seems to work on anything
-machine accel=kvm	Use an accelerator, I've found kvm is the best (default: tcg)
-m 2G	Set memory to 2GB (default: 128MB)
-hdb $dev	Set our USB as hard drive 1

Creating using script

```
sudo bash write_usb /dev/<usb device>
```

LIVE DEMO

(Testing with one ISO "System Rescue CD 6.x" hint: unremark line #177)

Improvements

- Expand all options on commands to full option names in script eg: change -a to –archive
- Test using sgdisk for MBR
- Make it easier to use
- Split up the file into modules

References

- Linux Format 246 (C-x r l)

Questions

Email	map7777@gmail.com
Twitter	@map7
Github	github: map7