Debian Live

Mini-debconf Cambridge

Emanuele Rocca
October 2024

Outline

Live Distros

Debian Live

Under the Hood

Case Studies

Live Distros

What is a Live Distribution?

- A Linux Distribution running directly from a USB stick
- Compact Discs back in the day
- No need to install anything on the hard disk
- Trying out a distro, system recovery, temporary usage, …

ISO 9660 universal
CDROM filesystem with
Rockridge extensions

Yggdrasil Computing,
Incorporated
4880 Stevens Creek Blvd.
Suite 205
San Jose, California
95129-1034
United States of America
toll free: (800) 261-6630
(408) 261-6630
fax: (408) 261-6631

To install, insert floppy in drive A, insert CDROM and reset computer.

Requires 386/486/586 CPU, 4MB RAM, floppy drive, Sony CDU-31A, Sony 535, Mitsumi, Matsushita/Panasonic/LaserMate or SCSI CDROM with Adaptec 1542B/1542C/1740, Always IN-2000, Bustek 542B, Future Domain 8xx/16xx, Seagate ST01/02 (or other other TMC-950 based controller), Ultrastor 14F/24F/34F, Western Digital 7000faast SCSI controller

Figure: A Compact Disc

Early Days

- Yggdrasil Linux (1992)
  - Plug-and-Play Linux!
  - Free Software For The Rest of Us
- Knoppix, Finnix (2000)
  - Based on Debian, general purpose
  - Popularized the idea of live operating systems
- Debian Live (2006)
- Fedora 7, openSUSE 10.3 (2007)

Specialized Live Distros

- Tails - privacy and anonymity
- Kali Linux - forensics, pentesting
- Puppy Linux - ease of use, older hardware
- Grml - data recovery, system repair, sysadmins in general

Debian Live

What is Debian Live?

- Official Live Images provided by the Debian Project
- Based on regular Debian, unchanged Debian packages only
- Available for a variety of desktop environments (GNOME, KDE, XFCE, LXDE, LXQT, Cinnamon, Mate)

Live Build

Most Important Slide of the Whole Presentation?

A framework to create custom Live images based on Debian:

```
$ sudo apt install live-build
$ lb config # in a temporary directory
$ echo live-task-gnome > config/package-lists/desktop.list.chroot
$ sudo lb build
```

Live Build

Image Customization

- Install a package:
  `config/package-lists/live.list.chroot`
- Ship a custom file:
  `config/includes.chroot_after_packages/etc/motd`
- Run a script:
  `config/hooks/normal/9999-remove-doc-man.hook.chroot`

Debian Live Manual

- Comprehensive documentation for advanced users and developers of Live Build
- All you need to know, and a bit more
- `https://live-team.pages.debian.net/live-manual/`

Debian Live Images

Where to get them

- Official Debian Live images can be downloaded from `https://www.debian.org/CD/live/`
- Images available for amd64, arm64 is coming!

Official arm64 Images

How are they built

- Official images built on casulana.d.o, a x86 machine
- Generating non-x86 images means ”cross-building” with `qemu-user-static`
- Live Build scripts use `debootstrap` with:
  1. `--foreign` to unpack the .debs only
  2. `--second-stage` under `qemu-user-static`
- After debootstrap, set things up and install packages in the chroot
- Emulation is very slow, hours instead of minutes

Official arm64 Images

Work done to get them built

- Change obvious things (`grub-efi-arm64` vs `grub-efi-amd64`)
- Native builds vs ”cross-builds” reproducibility
- Automated testing on `openqa.debian.net`
- One official image built!
  `https://get.debian.org/images/weekly-live-builds/arm64/`

Under the Hood

Core Components of Debian Live

- Linux kernel - stock Debian Kernel
- Initramfs – uses the scripts from package `live-boot`
- SquashFS – Compressed read-only file system, this is your system
- OverlayFS – Allows writable overlay on top of SquashFS
- live-config.service - runs the scripts from package `live-config`

Boot Process of Regular Systems

- UEFI firmware loads `/EFI/boot/bootaa64.efi` (EFI removable media path)

Boot Process of Regular Systems

- UEFI firmware loads `/EFI/boot/bootaa64.efi` (EFI removable media path)
- GRUB reads `/boot/grub/grub.cfg`

Boot Process of Regular Systems

- UEFI firmware loads `/EFI/boot/bootaa64.efi` (EFI removable media path)
- GRUB reads `/boot/grub/grub.cfg`
- `linux /vmlinuz-6.1.0-25-amd64 root=/dev/whatever [...]`
- `initrd /initrd.img-6.1.0-25-amd64`

Boot Process of Regular Systems

- UEFI firmware loads `/EFI/boot/bootaa64.efi` (EFI removable media path)
- GRUB reads `/boot/grub/grub.cfg`
- `linux /vmlinuz-6.1.0-25-amd64 root=/dev/whatever [...]`
- `initrd /initrd.img-6.1.0-25-amd64`
- Kernel boots, unpacks initrd, calls `/init`

Boot Process of Regular Systems

- UEFI firmware loads `/EFI/boot/bootaa64.efi` (EFI removable media path)
- GRUB reads `/boot/grub/grub.cfg`
- `linux /vmlinuz-6.1.0-25-amd64 root=/dev/whatever [...]`
- `initrd /initrd.img-6.1.0-25-amd64`
- Kernel boots, unpacks initrd, calls `/init`
- `/init` mounts root and calls `/sbin/init` (→ `/lib/systemd/systemd`)

Boot Process of Debian Live

Initial boot and SquashFS discovery

- `linux /live/vmlinuz-6.10.6-arm64` **boot=live** [...]

Boot Process of Debian Live

Initial boot and SquashFS discovery

- `linux /live/vmlinuz-6.10.6-arm64` **boot=live** [...]
- initrd executes `/scripts/$boot` in classic initrd style

Boot Process of Debian Live

Initial boot and SquashFS discovery

- `linux /live/vmlinuz-6.10.6-arm64` **boot=live** [...]
- initrd executes `/scripts/$boot` in classic initrd style
- `/scripts/live` calls `Live` (package `live-boot`)

Boot Process of Debian Live

Initial boot and SquashFS discovery

- `linux /live/vmlinuz-6.10.6-arm64` **boot=live** [...]
- initrd executes `/scripts/$boot` in classic initrd style
- `/scripts/live` calls `Live` (package `live-boot`)
- `find_livefs` discovers where the SquashFS is

Boot Process of Debian Live

setup_unionfs and init

- `setup_unionfs` takes over and mounts:
  - the SquashFS at `/run/live/rootfs`
  - a writeable tmpfs at `/run/live/overlay`
  - an overlay of the above at `/root`
- The `/init` script calls `run-init /root /sbin/init`

Debugging a Live System

- Pass `debug` to the kernel command line and find the initrd output under `/run/initramfs/initramfs.debug`. Watch out because if you pass `debug=foobar` all output will go to the console instead!
- `initramfs-tools(7)` is where most of the arguments are documented
- `/var/log/live/boot.log` has the output of live-boot
- `journalctl -u live-config.service` has the output of live-config

Case Studies

Burner Laptop

- *Device unconnected to your true identity and free from any trackable purchase history or personally identifiable information*

Burner Laptop

- *Device unconnected to your true identity and free from any trackable purchase history or personally identifiable information*
- You are mostly sad about the stickers if gone

Burner Laptop

- *Device unconnected to your true identity and free from any trackable purchase history or personally identifiable information*
- You are mostly sad about the stickers if gone
- Travelling
- Attending a conference

Burner Laptop

- Default Debian Live image has tons of software installed and ready to go
- Terminal to SSH around, Firefox, PDF reader
- SSH and GPG keys on USB hardware authentication device
- Old Chromebooks with coreboot firmware are perfect for this
- Boot with `toram` in the morning, keep USB key in your pocket all day

Burner Laptop

Suggestions

- Shrinking the ISO may be a good idea if using `toram`
- The default password is `live`. You probably want to change it, and surely want to know it when resuming from suspend

DNSSEC Chain of Trust

Sign and Verify DNS Records

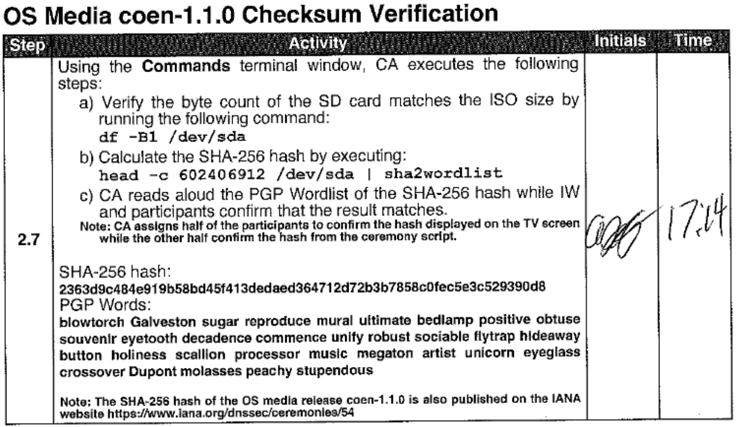
- Chain of Trust
- Validation process goes up the hierarchy, verify the parent
- The world assumes the public key is valid **because** of the security measures in place
- The key for the Root Zone is extremely important
- `"The key to worldwide internet security" -- The Guardian`

DNSSEC Root Zone Signing Ceremonies

- Very public, highly audited procedures
- Full videos and annotated scripts available: `https://www.iana.org/dnssec/ceremonies/54`
- The OS is a Live distro
- One step of the ceremony verifies that there is no hard drive in the laptop
- DVDs used till ceremony 50 (July 2023)
- Read-only SD card since ceremony 51 (November 2023)

Ceremony Operating ENvironment (COEN)

- `https://github.com/iana-org/coen`
- Debian Live system built with a custom procedure
- Snapshot-based, generate the same hash any time the COEN ISO image is built



OS Media coen-1.1.0 Checksum Verification

Step	Activity	Initials	Time

2.7	Using the **Commands** terminal window, CA executes the following steps:
	a) Verify the byte count of the SD card matches the ISO size by running the following command:
	`df -B1 /dev/sda`
	b) Calculate the SHA-256 hash by executing:
	`head -c 602406912 /dev/sda | sha2wordlist`
	c) CA reads aloud the PGP Wordlist of the SHA-256 hash while IW and participants confirm that the result matches.
	Note: CA assigns half of the participants to confirm the hash displayed on the TV screen while the other half confirm the hash from the ceremony script.
	SHA-256 hash:
	2363d9c484e919b58bd45f413dedaed364712d72b3b7858c0fec5e3c529390d8
	PGP Words:
	blowtorch Galveston sugar reproduce mural ultimate bedlamp positive obtuse souvenir eyetooth decadence commence unify robust sociable flytrap hideaway button holiness scallion processor music megaton artist unicorn eyeglass crossover Dupont molasses peachy stupendous
	Note: The SHA-256 hash of the OS media release coen-1.1.0 is also published on the IANA website https://www.iana.org/dnssec/ceremonies/54
			17:14

Figure: Ceremony 54, SD card verification

Server Benchmarks

- Network boot of live ISO
- Setup the environment with Ansible
- Run benchmarks

Server Benchmarks

- Very easy to create a custom image with all needed software that runs Ansible as part of the boot process
- Repeatable, simple to guarantee that two systems are running the same stack
- No need to deal with partitioning and installing software and boot loaders
- Not overwriting whatever is already on the server’s disk

Commonalities

- Guarantee of identity
- Temporariness of changes

Commonalities

- Guarantee of identity
- Temporariness of changes
- Simple and clear mental model

Commonalities

- Guarantee of identity
- Temporariness of changes
- Simple and clear mental model
- Trivial rollbacks to known state

Commonalities

- Guarantee of identity
- Temporariness of changes
- Simple and clear mental model
- Trivial rollbacks to known state
- Simple configuration management

Security

Traditional

- Immutable `/usr`
- A/B partitions
- TPM

Security

Traditional

- Immutable `/usr`
- A/B partitions
- TPM

Live

- Media is read-only (or absent!)
- USB stick A, USB stick B
- `sha2wordlist`

Conclusions

- Live Distros are cool and Debian is the best
- First official pre-built Debian Live images for arm64 available!
- Live Build is a very easy way to create your own

Thank You!

Danke!

Merci!

谢谢！

ありがとう！

Gracias!

Kiitos!

감사합니다

धन्यवाद

arm

Creating a Small Live Image

General Idea Recap

- Tools: `lb_config(1)` and `lb_build(1)`
- Steps:

  1. `lb config --apt-indices false [...] --distribution sid`
  2. Customize stuff under `config/`
  3. `sudo lb build`

Creating a Small Live Image

Better initrd compression

- `echo xz-utils >> config/package-lists/live.list.chroot`
- Files in `config/includes.chroot_after_packages/` are copied under /
- Set `COMPRESS=xz` and `COMPRESSLEVEL=9` under `etc/initramfs-tools/conf.d/compress`

Creating a Small Live Image

Remove What is Not Needed

- Write a hook such as `config/hooks/normal/9020-remove-doc-man.hook.chroot`
- Remove `/usr/share/doc/`, `/usr/share/man/`, `/usr/share/locale/`

Creating a Small Live Image

- It would be great to build test images in CI
- salsa.debian.org has a hard limit of 250M for artifact size
- Working live ISOs for amd64 and arm64 can be built
- `https://salsa.debian.org/ema/live-build/-/pipelines/726344`

Slow parts of cross-image building

- lb bootstrap_debootstrap 44 minutes
- lb chroot_install-packages install 6 hours, 29 minutes
- lb installer_debian-installer 2 hours, 39 minutes
- lb binary_rootfs 1 hour