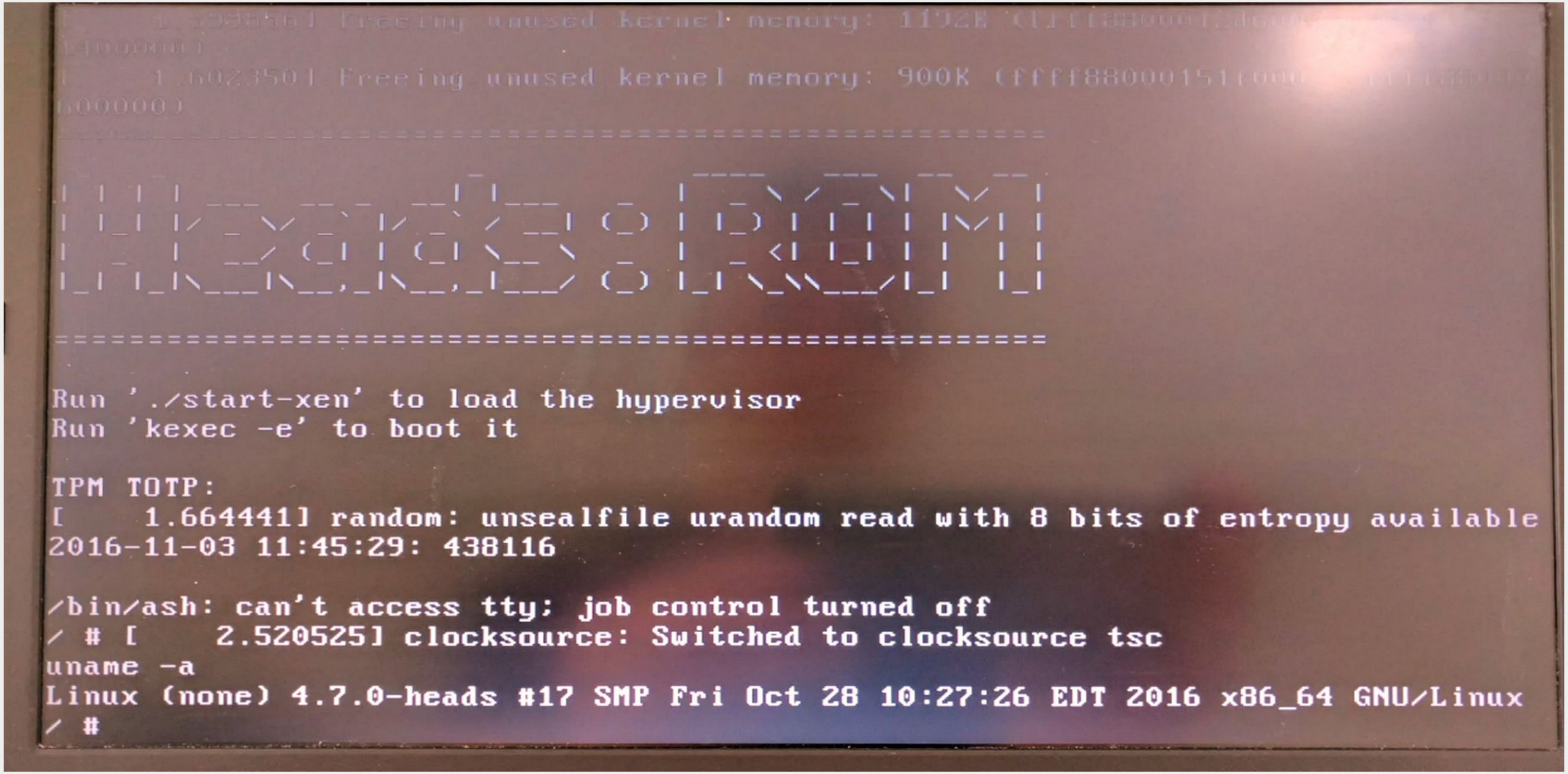
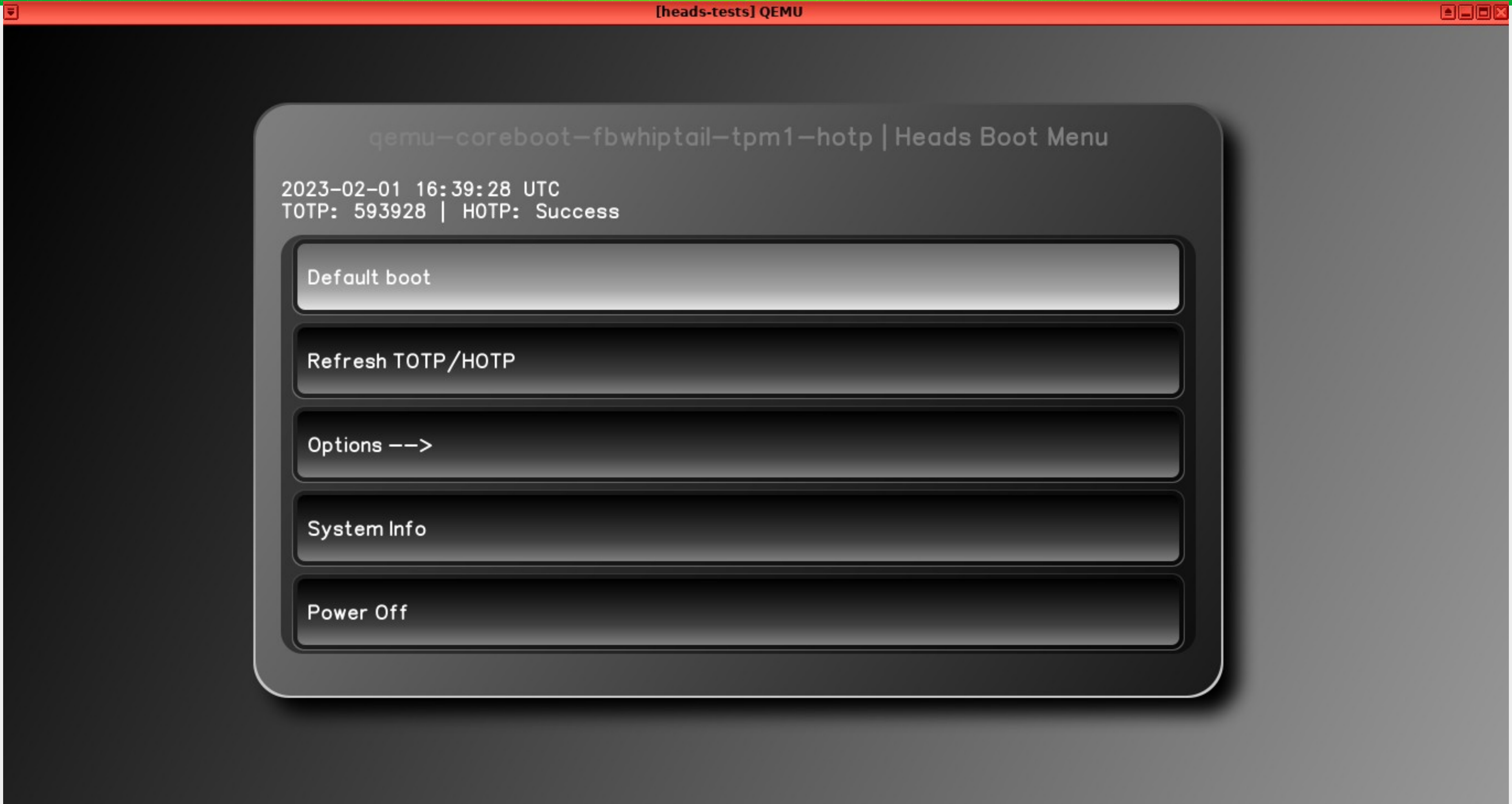
Heads : Status update

Heads at 33c3 (2016!!!)



```
[    1.598856] Freeing unused kernel memory: 1192K (ffff88000[illegible]
1400000)
[    1.602350] Freeing unused kernel memory: 900K (ffff880001511000[illegible]
600000)
===============================================================
 _   _                _        _   ____   ___  __  __
| | | | ___  __ _  __| |___   _   |  _ \ / _ \|  \/  |
| |_| |/ _ \/ _` |/ _` / __| (_)  | |_) | | | | |\/| |
|  _  |  __/ (_| | (_| \__ \  _   |  _ <| |_| | |  | |
|_| |_|\___|\__,_|\__,_|___/ (_)  |_| \_\\___/|_|  |_|

===============================================================

Run './start-xen' to load the hypervisor
Run 'kexec -e' to boot it

TPM TOTP:
[    1.664441] random: unsealfile urandom read with 8 bits of entropy available
2016-11-03 11:45:29: 438116

/bin/ash: can't access tty; job control turned off
/ # [    2.520525] clocksource: Switched to clocksource tsc
uname -a
Linux (none) 4.7.0-heads #17 SMP Fri Oct 28 10:27:26 EDT 2016 x86_64 GNU/Linux
/ #
```

Heads today



```
2023-02-01 16:39:28 UTC
TOTP: 593928 | HOTP: Success
```

What will we be talking about

Plan for today

- **Who am I?**
- What is Heads
- Why Heads
- What’s new?
- What’s next?

Thierry Laurion

- Insurgo Open Technology founder and CEO.
- Former Security Analyst/Psychology Bachelor/Security Researcher and Developer.
- Now freedom defender as a open source firmware researcher/developer/integrator.

- Past collaborator to Libreboot, QubesOS contributor and Heads collaborator/reviewer.
- Currently main Heads maintainer.

- Started Insurgo Open Technologies in 2017.
- Made the PrivacyBeast X230 certified by QubesOS in July 2019.
- Nlnet grantee for the Accessible Security project in April 2019.
- *Nlnet grantee once again for Authenticated Heads (Heads-OpenPGP) project.*

Insurgo's mission is to **facilitate accessibility to security and confidentiality** to the masses.

What will we be talking about

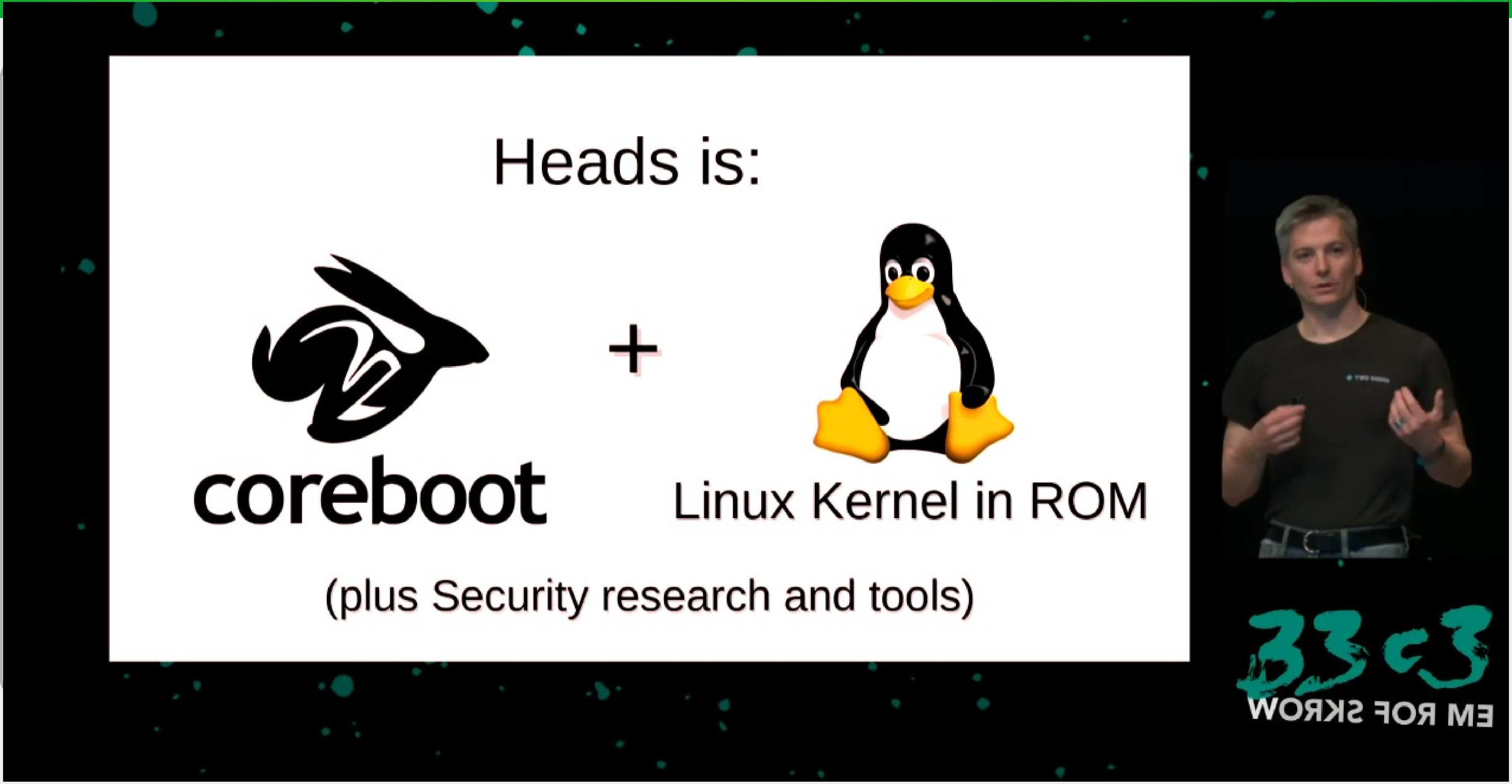
Plan for today

- Who am I?
- **What is Heads**
- Why Heads
- What’s new?
- What’s next?

What is Heads?

- Heads as a runtime environment
- Heads as a build system

Heads as runtime environment

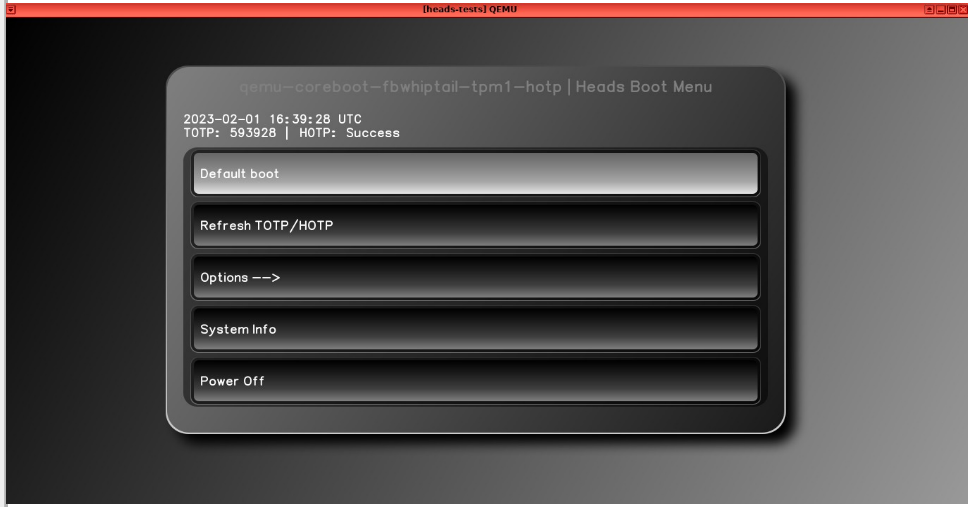


Heads is:

coreboot + Linux Kernel in ROM

(plus Security research and tools)

Heads as a runtime environment



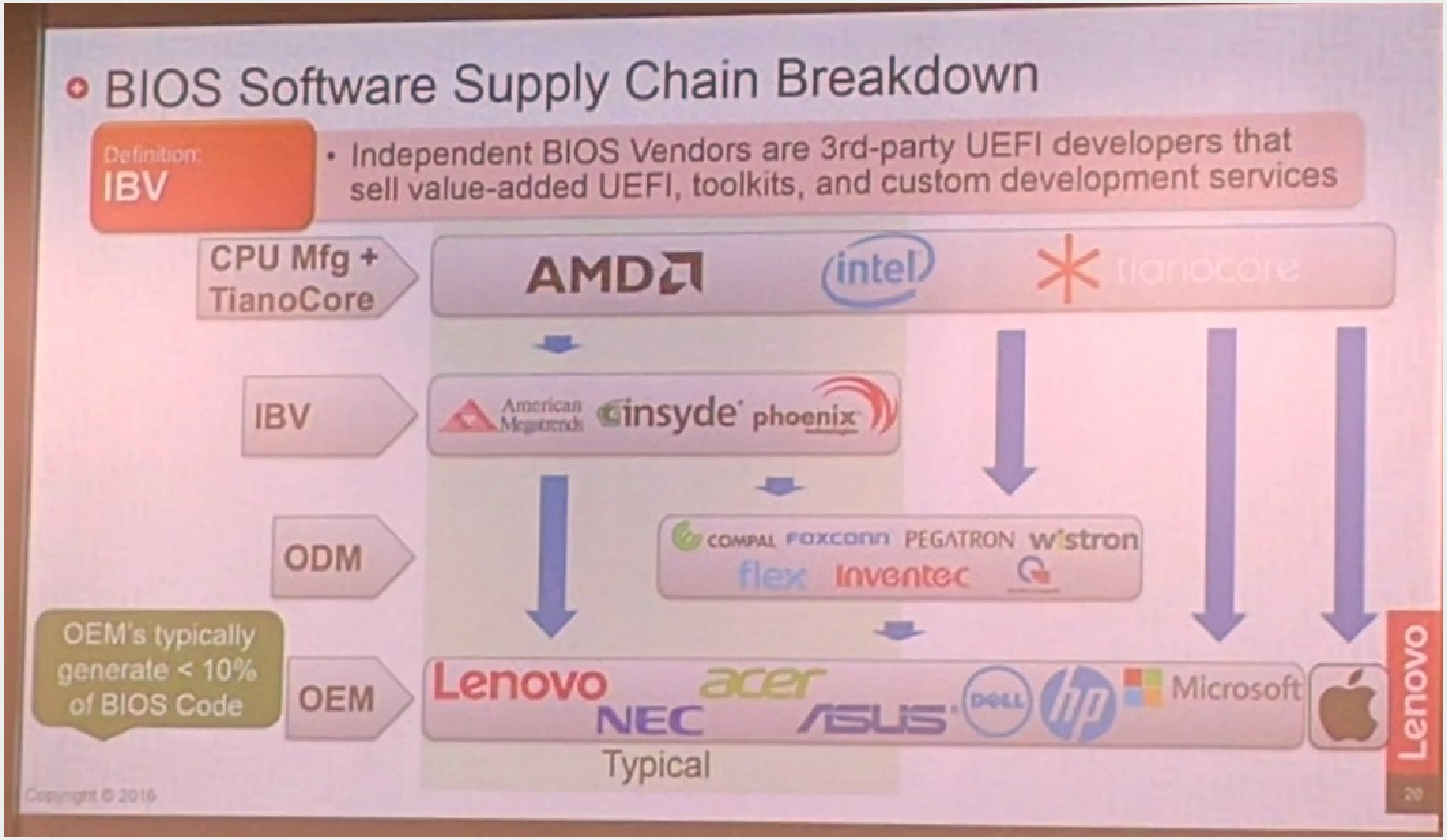
```
2023-02-01  16:39:28 UTC
TOTP:  593928 |  HOTP:  Success
```

- **coreboot**
  - **Hardware initialization**
  - Heads as linux's “bootloader” (kexec)
- Heads: linux as a coreboot payload
  - linux Kernel
  - linux initrd (initramfs)
    - Contain standard linux tools
    - Enforced security policies (shell scripts)

Heads – coreboot native hardware init

- **Openness/ownership/auditability of a Heads' coreboot supported platform depends on its coreboot's blobs requirements**
- Native hardware initialization depends on arch + chipset
  - On x86: Intel - Ivy bridge/ Sandy bridge : all native (no blobs) (watch for Haswell: native ram init coming)
  - On x86: AMD – Fam15h: all native (no blobs)
    - KGPE-D16 (coreboot 4.11 last official supported version)
      - Dasharo/coreboot (based on coreboot 4.16)
  - Power9 : Talos II: all native (no blobs)
- More info: https://github.com/osresearch/heads/issues/692

Closed source firmware / BIOS Supply chain



BIOS Software Supply Chain Breakdown
Definition:
IBV
Independent BIOS Vendors are 3rd-party UEFI developers that sell value-added UEFI, toolkits, and custom development services
CPU Mfg + TianoCore
AMD
intel
tianocore
IBV
American Megatrends
insyde
phoenix
ODM
COMPAL
FOXCONN
PEGATRON
wistron
flex
Inventec
OEM's typically generate < 10% of BIOS Code
OEM
Lenovo
acer
NEC
ASUS
DELL
hp
Microsoft
Typical
Copyright © 2016
Lenovo
20

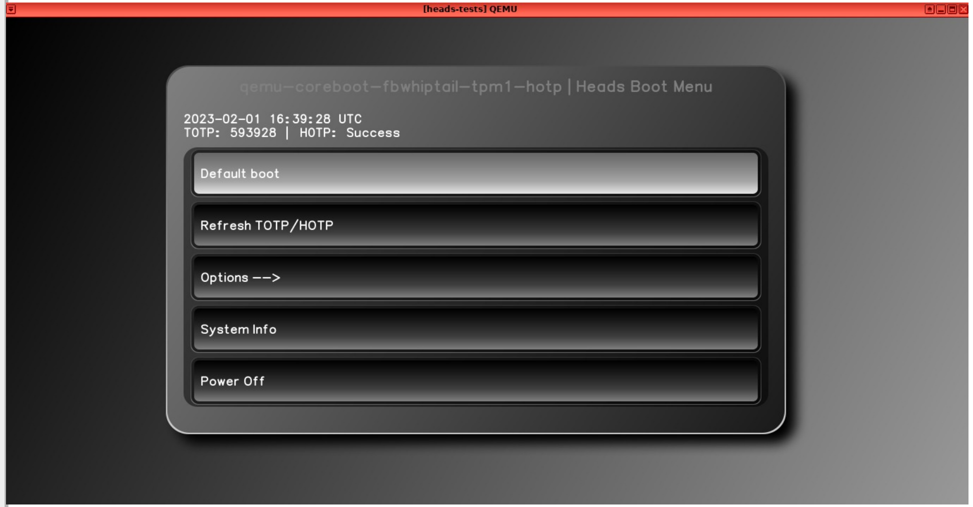
Heads OSFW

- Coreboot source code (and faith in its platform’s blobs dependencies)
- Linux kernel (version + config)
- Linux tools (modules dependencies version+config) of a board configuration

Let’s note that the linux kernel has:

- extremely limited built in modules/dependencies (linux config for a board)
  - drivers built as modules are loaded on demand
    - USB drivers needed for HOTP USB security dongle
      - USB keyboard not present unless no PS2
      - USB HID not present unless no PS2
- Drivers compiled as modules are measured prior of being loaded (TPM : PCR5)
  - consequently, a default boot
    - won’t release TPM NV sealed disk unlock key if cannot be unsealed
      - **PCRs will mismatch from sealed and will not unseal**

Heads as a runtime environment

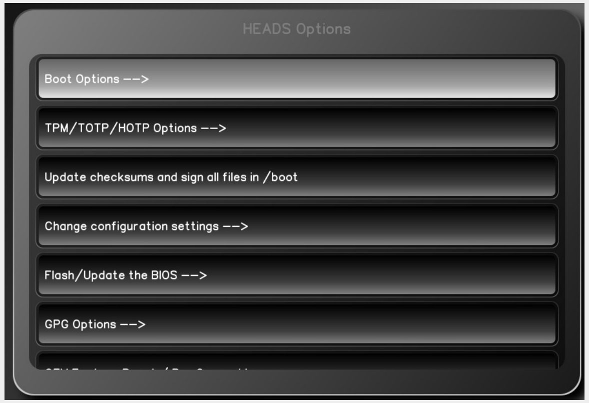


- coreboot
  - Hardware initialization
  - **Heads as linux's "bootloader" (kexec)**
- Heads: linux as a coreboot payload
  - linux Kernel
  - linux initrd (initramfs)
    - Contain standard linux tools
    - Enforced security policies (shell scripts)

Heads as a linux bootloader

- A bootloader is
  - Drivers to be able to deal with I/O
  - Parser for OS configs
  - Pass control to the OS
- Bootloader sits between BIOS and OS to be booted
- **Linux and scripts can do exactly the same without duplicating hardware init nor extending trusted code base…**
  - Busybox (shell)
  - Cryptsetup, lvm, tpm toolstack…

Heads as a linux bootloader


HEADS Options

Boot Options -->

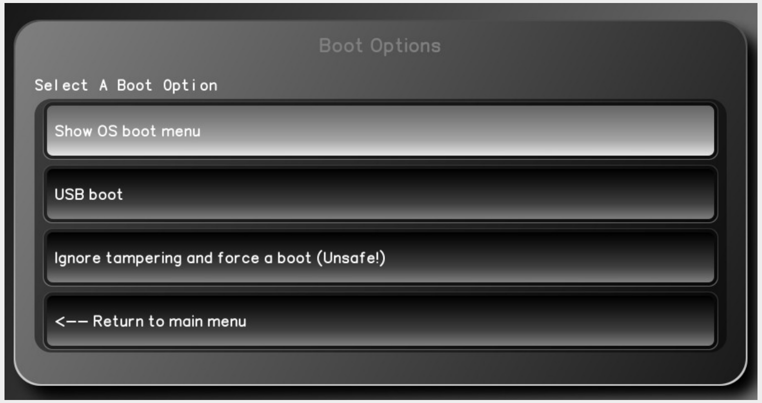
TPM/TOTP/HOTP Options -->

Update checksums and sign all files in /boot

Change configuration settings -->

Flash/Update the BIOS -->

GPG Options -->


Boot Options

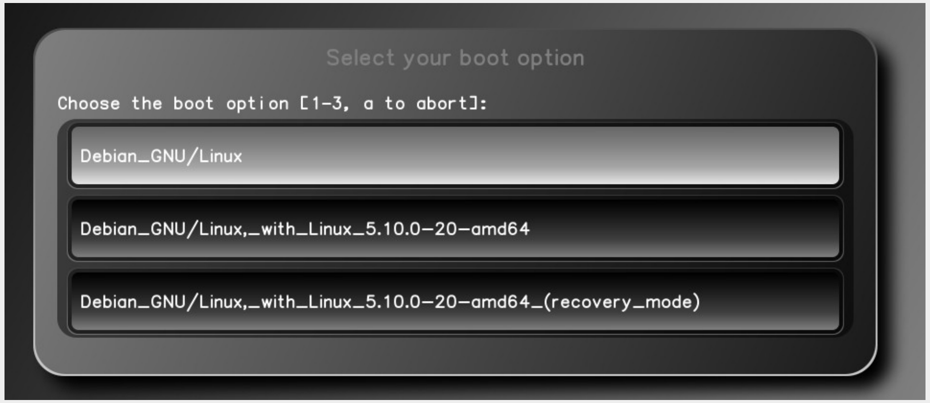
Select A Boot Option

Show OS boot menu

USB boot

Ignore tampering and force a boot (Unsafe!)

<-- Return to main menu

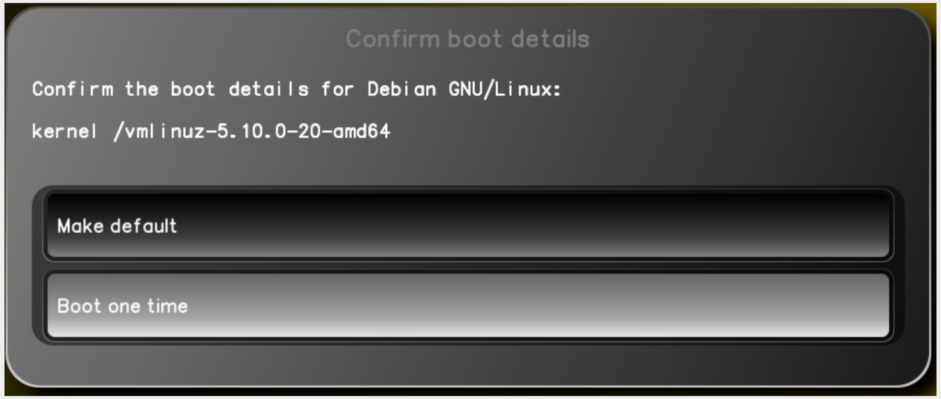

Select your boot option

Choose the boot option [1-3, a to abort]:

Debian_GNU/Linux

Debian_GNU/Linux,_with_Linux_5.10.0-20-amd64

Debian_GNU/Linux,_with_Linux_5.10.0-20-amd64_(recovery_mode)


Confirm boot details

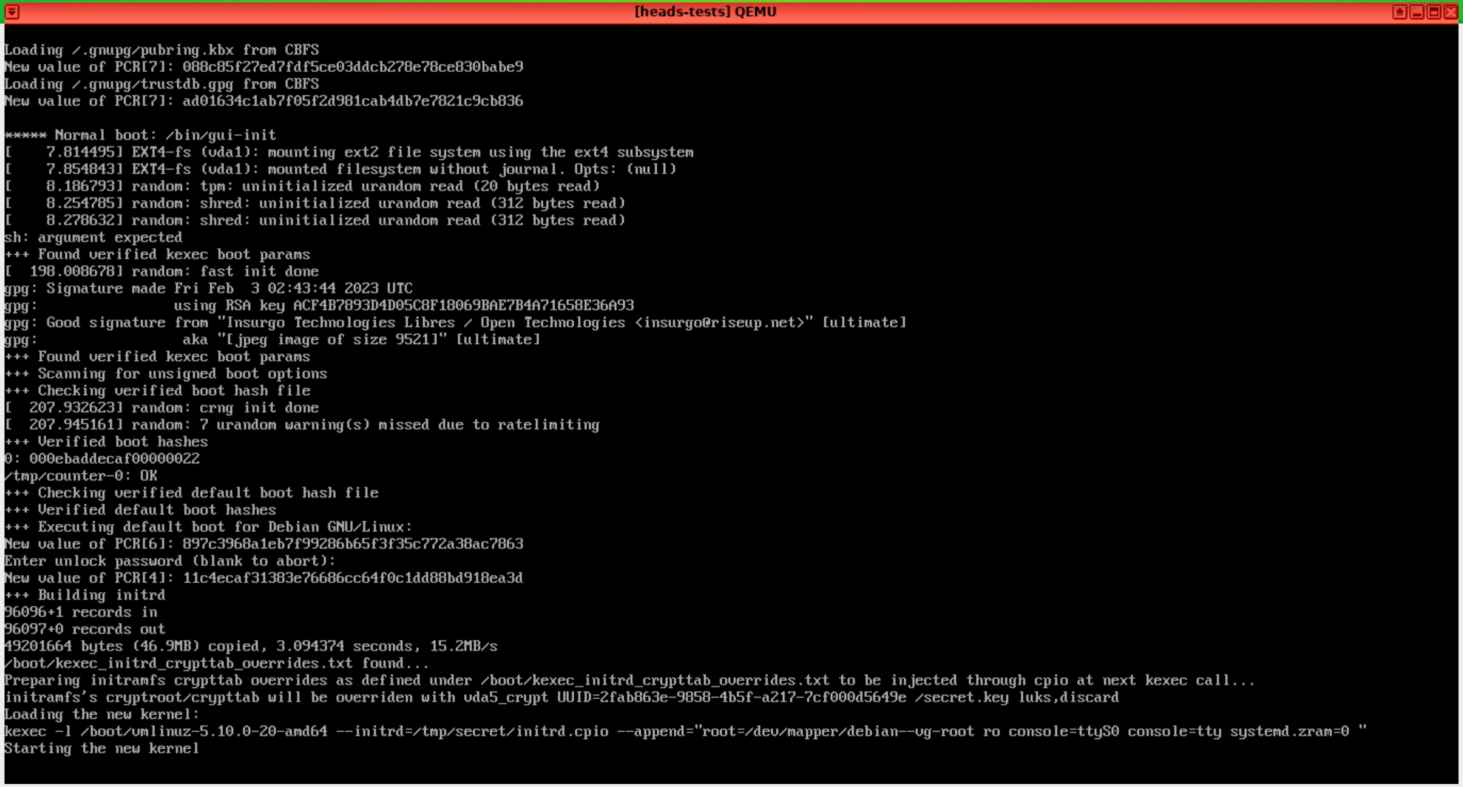
Confirm the boot details for Debian GNU/Linux:

kernel /vmlinuz-5.10.0-20-amd64

Make default

Boot one time

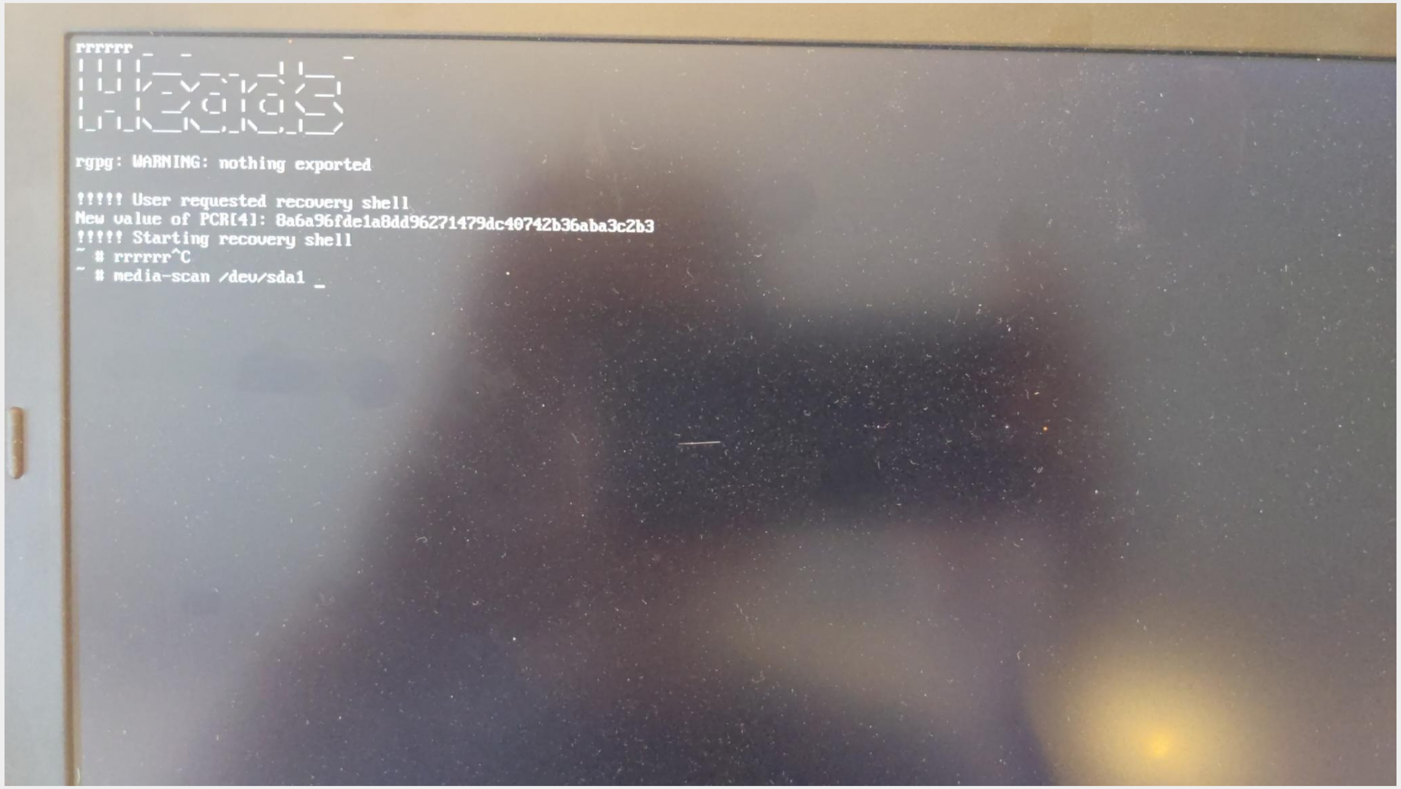
Heads as linux's "bootloader" (kexec)



```
Loading /.gnupg/pubring.kbx from CBFS
New value of PCR[7]: 088c85f27ed7fdf5ce03ddcb278e78ce830babe9
Loading /.gnupg/trustdb.gpg from CBFS
New value of PCR[7]: ad01634c1ab7f05f2d981cab4db7e7821c9cb836

***** Normal boot: /bin/gui-init
[    7.814495] EXT4-fs (vda1): mounting ext2 file system using the ext4 subsystem
[    7.854843] EXT4-fs (vda1): mounted filesystem without journal. Opts: (null)
[    8.186793] random: tpm: uninitialized urandom read (20 bytes read)
[    8.254785] random: shred: uninitialized urandom read (312 bytes read)
[    8.278632] random: shred: uninitialized urandom read (312 bytes read)
sh: argument expected
+++ Found verified kexec boot params
[  198.008678] random: fast init done
gpg: Signature made Fri Feb  3 02:43:44 2023 UTC
gpg:                using RSA key ACF4B7893D4D05C8F18069BAE7B4A71658E36A93
gpg: Good signature from "Insurgo Technologies Libres / Open Technologies <insurgo@riseup.net>" [ultimate]
gpg:                 aka "[jpeg image of size 9521]" [ultimate]
+++ Found verified kexec boot params
+++ Scanning for unsigned boot options
+++ Checking verified boot hash file
[  207.932623] random: crng init done
[  207.945161] random: 7 urandom warning(s) missed due to ratelimiting
+++ Verified boot hashes
0: 000ebaddecaf00000022
/tmp/counter-0: OK
+++ Checking verified default boot hash file
+++ Verified default boot hashes
+++ Executing default boot for Debian GNU/Linux:
New value of PCR[6]: 897c3968a1eb7f99286b65f3f35c772a38ac7863
Enter unlock password (blank to abort):
New value of PCR[4]: 11c4ecaf31383e76686cc64f0c1dd88bd918ea3d
+++ Building initrd
96096+1 records in
96097+0 records out
49201664 bytes (46.9MB) copied, 3.094374 seconds, 15.2MB/s
/boot/kexec_initrd_crypttab_overrides.txt found...
Preparing initramfs crypttab overrides as defined under /boot/kexec_initrd_crypttab_overrides.txt to be injected through cpio at next kexec call...
initramfs's cryptroot/crypttab will be overriden with vda5_crypt UUID=2fab863e-9858-4b5f-a217-7cf000d5649e /secret.key luks,discard
Loading the new kernel:
kexec -l /boot/vmlinuz-5.10.0-20-amd64 --initrd=/tmp/secret/initrd.cpio --append="root=/dev/mapper/debian--vg-root ro console=ttyS0 console=tty systemd.zram=0 "
Starting the new kernel
```

In action: gui-init boot policy:
- detach signature against public key + hash validation. TPM NV auth + unsealing LUKS
- cpio constructed with parsed OS's crypttab + TPM disk unlock key inserted at kexec (cpio)

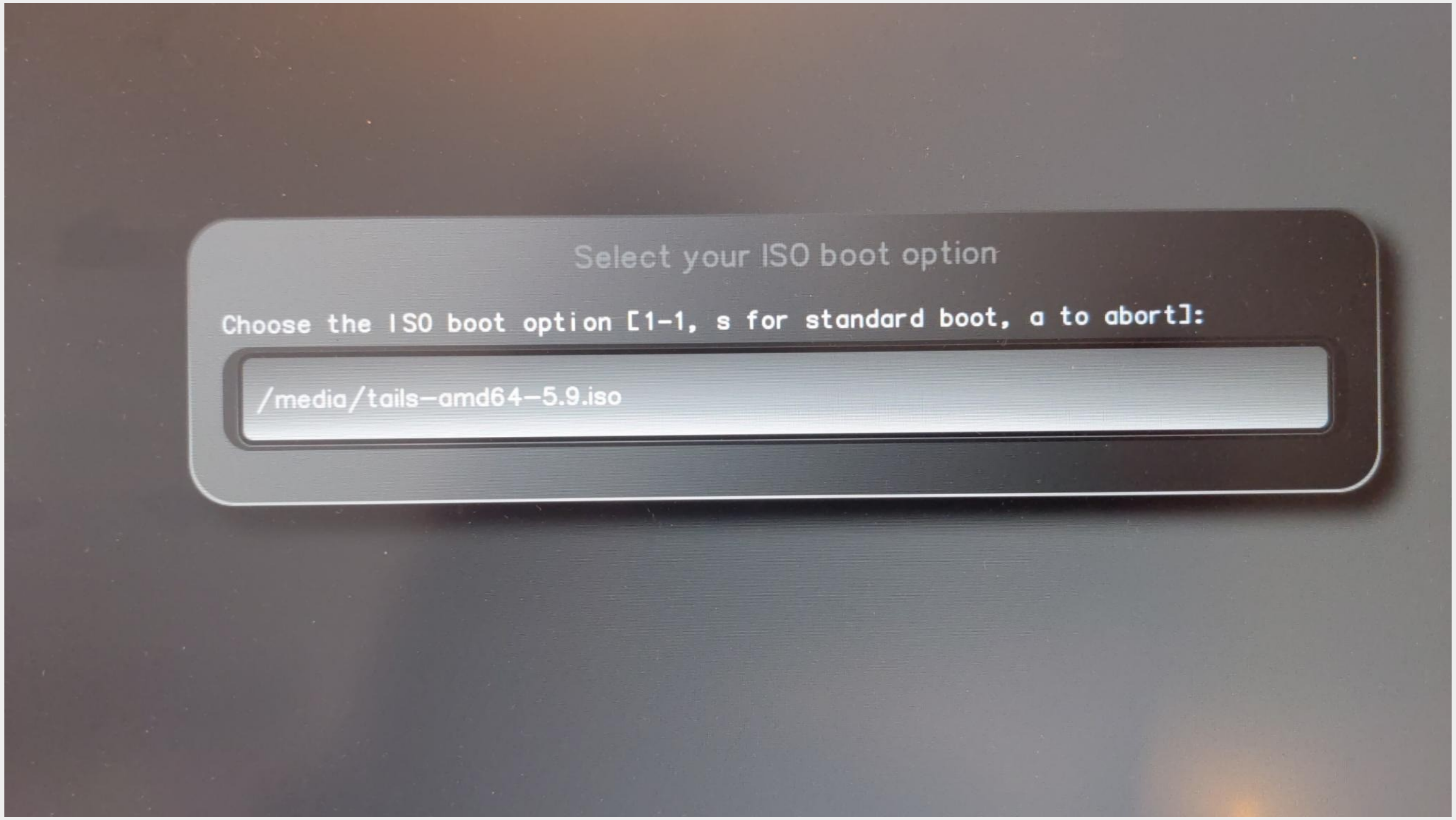
Heads as linux's "bootloader" (kexec)



```
rrrrrr _   _                _
| | | | ___  __ _  __| |___
| |_| |/ _ \/ _` |/ _` / __|
|  _  |  __/ (_| | (_| \__ \
|_| |_|\___|\__,_|\__,_|___/

rgpg: WARNING: nothing exported

!!!!! User requested recovery shell
New value of PCR[4]: 8a6a96fde1a8dd96271479dc40742b36aba3c2b3
!!!!! Starting recovery shell
~ # rrrrrr^C
~ # media-scan /dev/sda1 _
```

Sda1 is flat ext4 partition with ISO+detach signature (iso.asc) put there.
"media-scan" merged recently

• Heads as linux's "bootloader" (kexec)

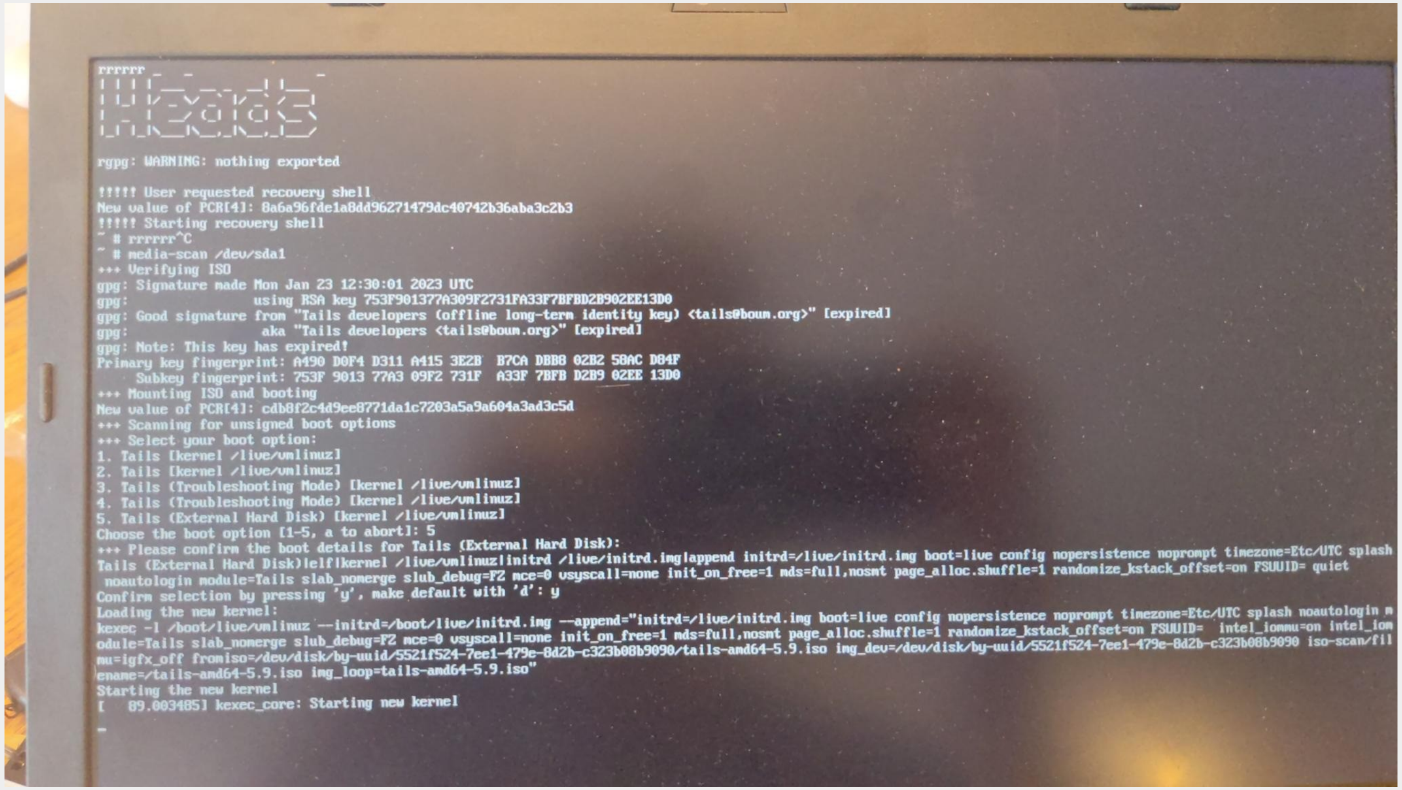


Select your ISO boot option

```
Choose the ISO boot option [1-1, s for standard boot, a to abort]:
```

```
/media/tails-amd64-5.9.iso
```

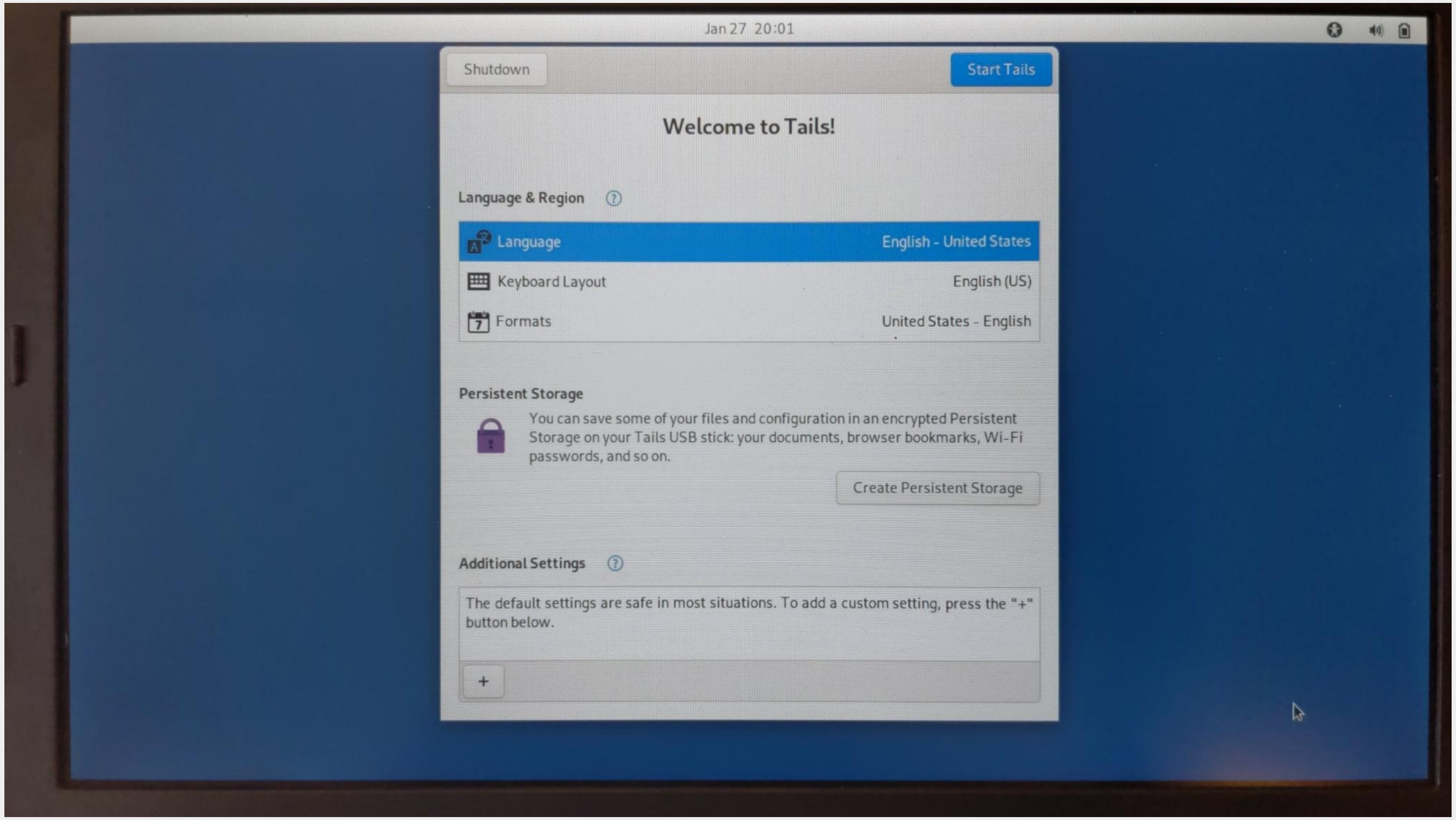
• Heads as linux’s “bootloader” (kexec)



```
rrrrrr _   _                _
| | | |  ___  __ _   __| | ___
| |_| | / _ \/ _` | / _` |/ __|
|  _  ||  __/ (_| || (_| |\__ \
|_| |_| \___|\__,_| \__,_||___/

rgpg: WARNING: nothing exported

!!!!! User requested recovery shell
New value of PCR[4]: 8a6a96fde1a8dd96271479dc40742b36aba3c2b3
!!!!! Starting recovery shell
~ # rrrrrr^C
~ # media-scan /dev/sda1
+++ Verifying ISO
gpg: Signature made Mon Jan 23 12:30:01 2023 UTC
gpg:                using RSA key 753F901377A309F2731FA33F7BFBD2B902EE13D0
gpg: Good signature from "Tails developers (offline long-term identity key) <tails@boum.org>" [expired]
gpg:                 aka "Tails developers <tails@boum.org>" [expired]
gpg: Note: This key has expired!
Primary key fingerprint: A490 D0F4 D311 A415 3E2B  B7CA DBB8 02B2 58AC D84F
     Subkey fingerprint: 753F 9013 77A3 09F2 731F  A33F 7BFB D2B9 02EE 13D0
+++ Mounting ISO and booting
New value of PCR[4]: cdb8f2c4d9ee8771da1c7203a5a9a604a3ad3c5d
+++ Scanning for unsigned boot options
+++ Select your boot option:
1. Tails [kernel /live/vmlinuz]
2. Tails [kernel /live/vmlinuz]
3. Tails (Troubleshooting Mode) [kernel /live/vmlinuz]
4. Tails (Troubleshooting Mode) [kernel /live/vmlinuz]
5. Tails (External Hard Disk) [kernel /live/vmlinuz]
Choose the boot option [1-5, a to abort]: 5
+++ Please confirm the boot details for Tails (External Hard Disk):
Tails (External Hard Disk)|elf|kernel /live/vmlinuz|initrd /live/initrd.img|append initrd=/live/initrd.img boot=live config nopersistence noprompt timezone=Etc/UTC splash
 noautologin module=Tails slab_nomerge slub_debug=FZ mce=0 vsyscall=none init_on_free=1 mds=full,nosmt page_alloc.shuffle=1 randomize_kstack_offset=on FSUUID= quiet
Confirm selection by pressing 'y', make default with 'd': y
Loading the new kernel:
kexec -l /boot/live/vmlinuz --initrd=/boot/live/initrd.img --append="initrd=/live/initrd.img boot=live config nopersistence noprompt timezone=Etc/UTC splash noautologin m
odule=Tails slab_nomerge slub_debug=FZ mce=0 vsyscall=none init_on_free=1 mds=full,nosmt page_alloc.shuffle=1 randomize_kstack_offset=on FSUUID=  intel_iommu=on intel_iom
mu=igfx_off fromiso=/dev/disk/by-uuid/5521f524-7ee1-479e-8d2b-c323b08b9090/tails-amd64-5.9.iso img_dev=/dev/disk/by-uuid/5521f524-7ee1-479e-8d2b-c323b08b9090 iso-scan/fil
ename=/tails-amd64-5.9.iso img_loop=tails-amd64-5.9.iso"
Starting the new kernel
[   89.003485] kexec_core: Starting new kernel

_
```

Detached signature validation against ROM’s OS’s distro signing public key
Then grub entree selection parsing + kexec

• Heads as linux's "bootloader" (kexec)



Welcome to Tails!

Language & Region

Language	English - United States
Keyboard Layout	English (US)
Formats	United States - English

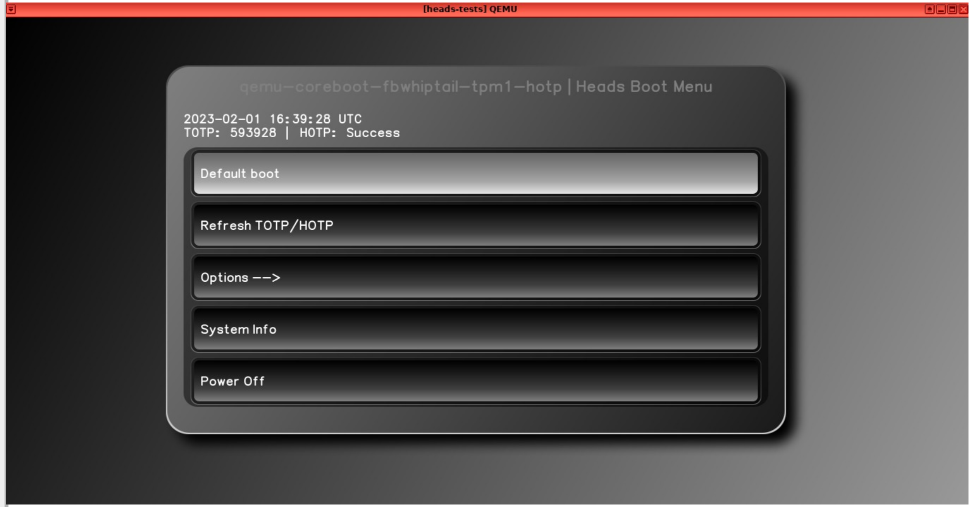
Persistent Storage

You can save some of your files and configuration in an encrypted Persistent Storage on your Tails USB stick: your documents, browser bookmarks, Wi-Fi passwords, and so on.

Additional Settings

The default settings are safe in most situations. To add a custom setting, press the "+" button below.

Heads as a runtime environment



```
2023-02-01 16:39:28 UTC
TOTP: 593928 | HOTP: Success
```

(gui-init through fbwhiptail above)

- coreboot
  - Hardware initialization
  - Heads as linux's "bootloader" (kexec)
- **Heads: linux as a coreboot payload**
  - **linux Kernel**
  - **linux initrd (initramfs)**
    - **Contain standard linux tools**
    - **Enforced security policies (shell scripts)**

What is Heads?

- Heads as a runtime environment
- Heads as a build system

Heads as a build system

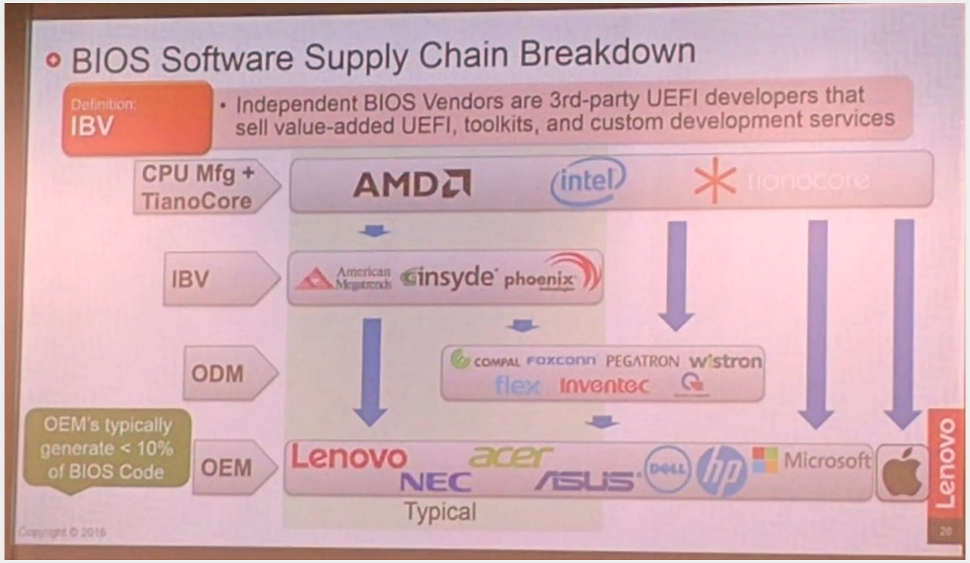
- Heads is basically a ‘Make’ project
  - Global Makefile https://github.com/osresearch/heads/blob/master/Makefile
    - make BOARD=board_name module_name.statement options
    - Boards https://github.com/osresearch/heads/tree/master/boards
    - Existing modules (compilable software): https://github.com/osresearch/heads/tree/master/modules
    - Patches to be applied after module verification + extraction: https://github.com/osresearch/heads/tree/master/patches
  - Produces
    - Artifacts : creboot rom(s) images stitching the following (but produced independently)
    - BzImage (compiled kernel + in-kernel modules)
    - Initrd.cpio.xz
      - tools.cpio (compiled modules stripped binaries)
      - modules.cpio (compiled as modules kernel drivers to be loaded on demand)
      - heads.cpio (scripts and config files generated at build time linked to board config and https://github.com/osresearch/heads/tree/master/initrd content
    - Hashes.txt file containing individual packed files, cpios, initrd.cpio.xz and coreboot roms

What will we be talking about

Plan for today

- Who am I?
- What is Heads
- **Why Heads**
- What’s new?
- What’s next?

Why Heads



BIOS Software Supply Chain Breakdown

Definition: IBV

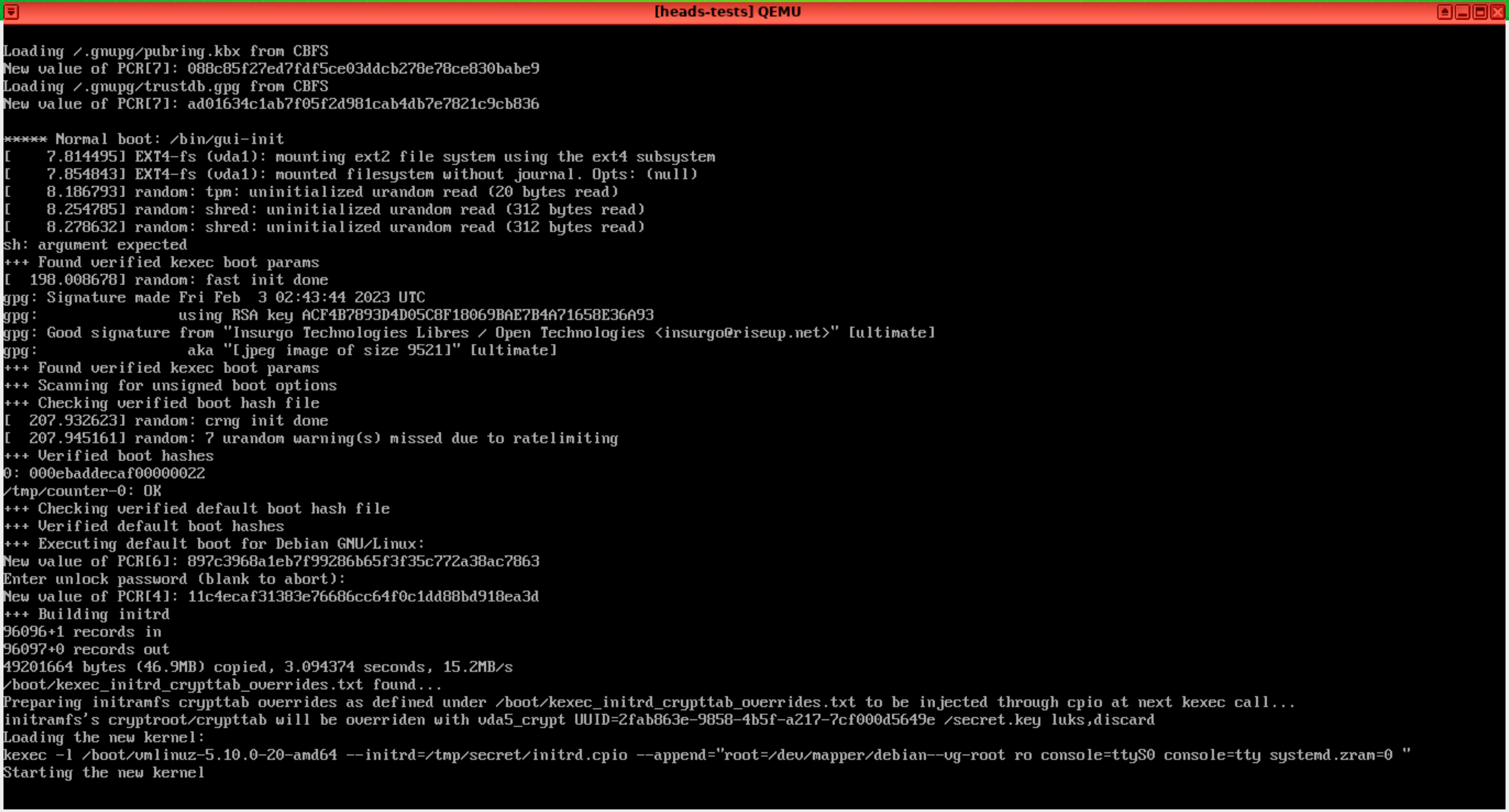
- Independent BIOS Vendors are 3rd-party UEFI developers that sell value-added UEFI, toolkits, and custom development services

- Outside of coreboot's "minimalist" mandate:
  - Linux as its bootloader
    - Linux kernel enumerates devices
    - Linux supports of peripherals/buses
    - Linux support of filesystems
    - Linux kernel setups of IOMMU
    - Linux permits Kexec'ing into final OS
  - Standard linux tools in initrd (**add your own module if missing**)
    - TPM toolstacks (sealing/unsealing of secrets)
  - Bash scripts defines the boot policies launched by init
    - gui-init: validates/creates detached signatures, hashes, LUKS headers
  - Maximized roms takes back all ME neutered freed space, unlocks firmware descriptor, permitting internal full firmware upgrades!

Why Heads

- Extensive TPM usage
  - Coreboot measured boot mode extends TPM PCR2 (no DRTM as of now on supported platforms. Might change with T440p and other newer platforms)
  - Heads extends others
    - PCR4: Boot mode (0 during /init, then recovery or normal-boot)
    - PCR5: Heads Linux kernel modules
    - PCR6: Drive LUKS headers
    - PCR7: Heads user-specific files stored in CBFS (config.user, GPG keyring, etc).
  - Why important?
    - Sealing secrets in TPM NV memory
      - TOTP/HOTP sealed secret (based on PCRs 0-4) : Firmware integrity attestation
      - TPM Disk Unlock Key (based on PCRs 0-7) sealed secret with custom passphrase: releases key to OS without you having to type passphrase to that OS

gui-init policy + TPM released Disk unlock key



```
Loading /.gnupg/pubring.kbx from CBFS
New value of PCR[7]: 088c85f27ed7fdf5ce03ddcb278e78ce830babe9
Loading /.gnupg/trustdb.gpg from CBFS
New value of PCR[7]: ad01634c1ab7f05f2d981cab4db7e7821c9cb836

***** Normal boot: /bin/gui-init
[    7.814495] EXT4-fs (vda1): mounting ext2 file system using the ext4 subsystem
[    7.854843] EXT4-fs (vda1): mounted filesystem without journal. Opts: (null)
[    8.186793] random: tpm: uninitialized urandom read (20 bytes read)
[    8.254785] random: shred: uninitialized urandom read (312 bytes read)
[    8.278632] random: shred: uninitialized urandom read (312 bytes read)
sh: argument expected
+++ Found verified kexec boot params
[  198.008678] random: fast init done
gpg: Signature made Fri Feb  3 02:43:44 2023 UTC
gpg:                using RSA key ACF4B7893D4D05C8F18069BAE7B4A71658E36A93
gpg: Good signature from "Insurgo Technologies Libres / Open Technologies <insurgo@riseup.net>" [ultimate]
gpg:                 aka "[jpeg image of size 9521]" [ultimate]
+++ Found verified kexec boot params
+++ Scanning for unsigned boot options
+++ Checking verified boot hash file
[  207.932623] random: crng init done
[  207.945161] random: 7 urandom warning(s) missed due to ratelimiting
+++ Verified boot hashes
0: 000ebaddecaf00000022
/tmp/counter-0: OK
+++ Checking verified default boot hash file
+++ Verified default boot hashes
+++ Executing default boot for Debian GNU/Linux:
New value of PCR[6]: 897c3968a1eb7f99286b65f3f35c772a38ac7863
Enter unlock password (blank to abort):
New value of PCR[4]: 11c4ecaf31383e76686cc64f0c1dd88bd918ea3d
+++ Building initrd
96096+1 records in
96097+0 records out
49201664 bytes (46.9MB) copied, 3.094374 seconds, 15.2MB/s
/boot/kexec_initrd_crypttab_overrides.txt found...
Preparing initramfs crypttab overrides as defined under /boot/kexec_initrd_crypttab_overrides.txt to be injected through cpio at next kexec call...
initramfs's cryptroot/crypttab will be overriden with vda5_crypt UUID=2fab863e-9858-4b5f-a217-7cf000d5649e /secret.key luks,discard
Loading the new kernel:
kexec -l /boot/vmlinuz-5.10.0-20-amd64 --initrd=/tmp/secret/initrd.cpio --append="root=/dev/mapper/debian--vg-root ro console=ttyS0 console=tty systemd.zram=0 "
Starting the new kernel
```

What will we be talking about

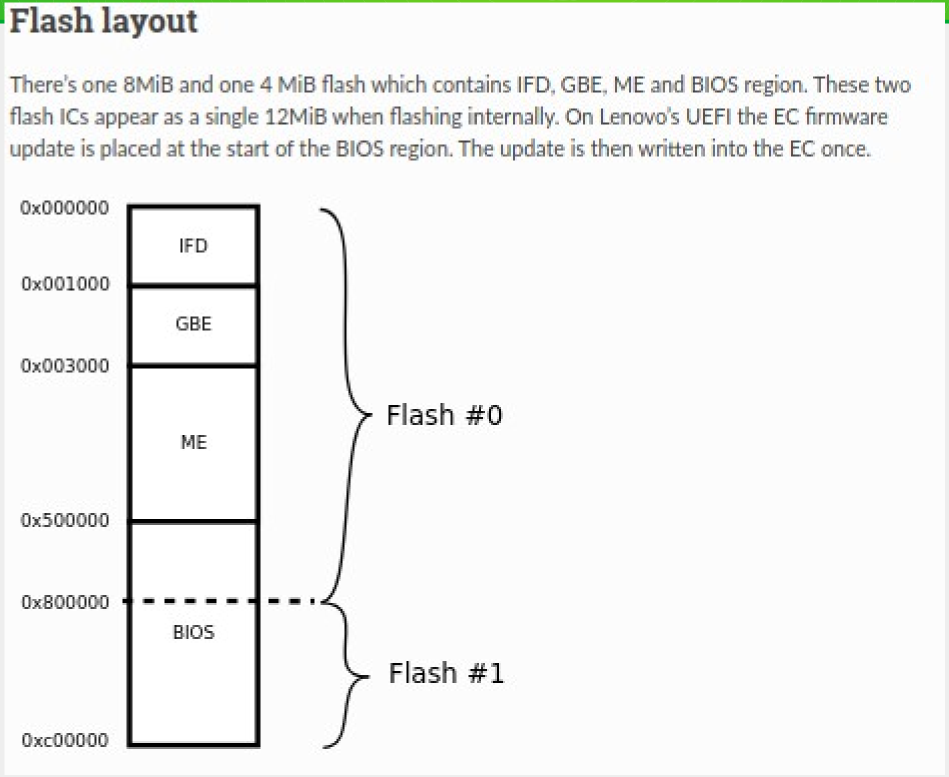
Plan for today

- Who am I?
- What is Heads
- Why Heads
- **What’s new?**
- What’s next?

What’s new

- **Maximized boards vs Legacy boards, or how to dodge blob redistribution legal limitations**
- Whiptail/FBWhiptail: one graphical interface (GUI) to rule them all
- OEM Factory reset/Re-Ownership wizard upstreamed
- QEMU/KVM board configurations with swtpm and USB Security dongle support to ease development/testing

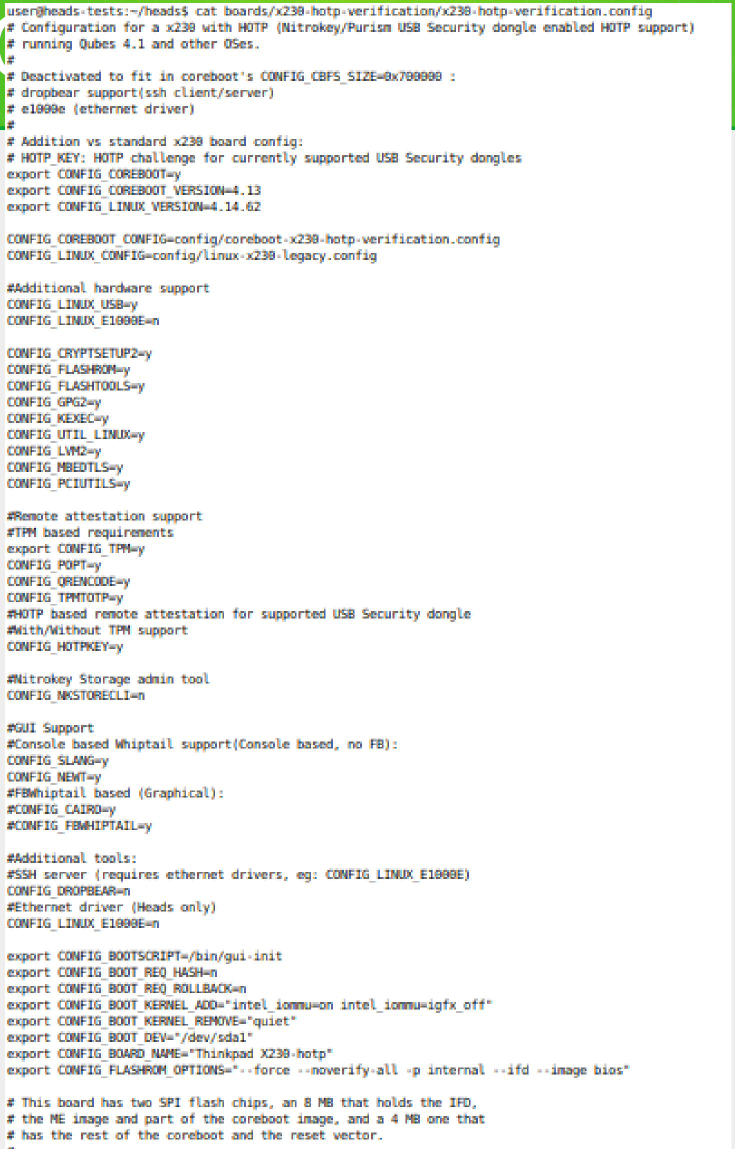
OEM/Legacy BIOS



Flash layout

There's one 8MiB and one 4 MiB flash which contains IFD, GBE, ME and BIOS region. These two flash ICs appear as a single 12MiB when flashing internally. On Lenovo's UEFI the EC firmware update is placed at the start of the BIOS region. The update is then written into the EC once.

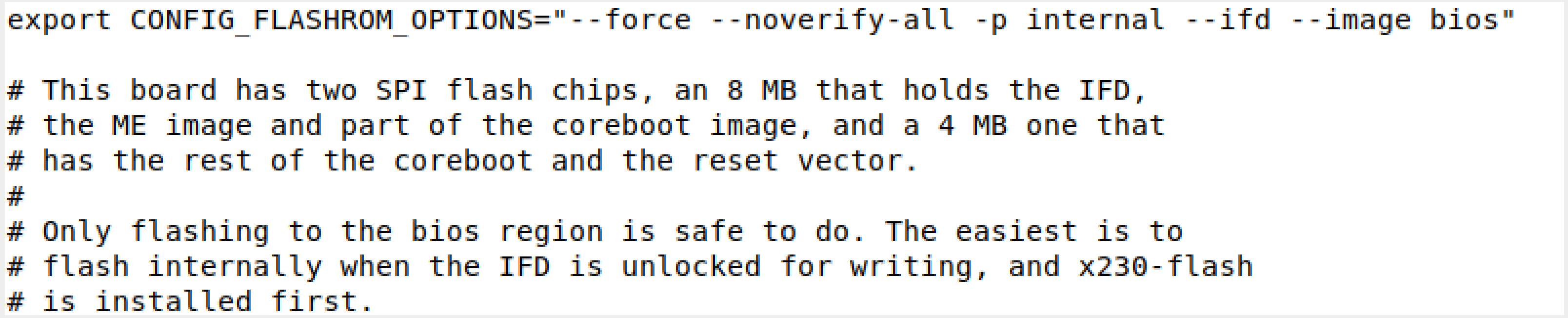
Intel Firmware Descriptor (IFD): Locked. Describes GBE ME BIOS sizes here!
IFD, Intel ME and BIOS regions : cannot be modified/unlocked but externally



```
user@heads-tests:~/heads$ cat boards/x230-hotp-verification/x230-hotp-verification.config
# Configuration for a x230 with HOTP (Nitrokey/Purism USB Security dongle enabled HOTP support)
# running Qubes 4.1 and other OSes.
#
# Deactivated to fit in coreboot's CONFIG_CBFS_SIZE=0x700000 :
# dropbear support(ssh client/server)
# e1000e (ethernet driver)
#
# Addition vs standard x230 board config:
# HOTP_KEY: HOTP challenge for currently supported USB Security dongles
export CONFIG_COREBOOT=y
export CONFIG_COREBOOT_VERSION=4.13
export CONFIG_LINUX_VERSION=4.14.62

CONFIG_COREBOOT_CONFIG=config/coreboot-x230-hotp-verification.config
CONFIG_LINUX_CONFIG=config/linux-x230-legacy.config

#Additional hardware support
CONFIG_LINUX_USB=y
CONFIG_LINUX_E1000E=n

CONFIG_CRYPTSETUP2=y
CONFIG_FLASHROM=y
CONFIG_FLASHTOOLS=y
CONFIG_GPG2=y
CONFIG_KEXEC=y
CONFIG_UTIL_LINUX=y
CONFIG_LVM2=y
CONFIG_MBEDTLS=y
CONFIG_PCIUTILS=y

#Remote attestation support
#TPM based requirements
export CONFIG_TPM=y
CONFIG_POPT=y
CONFIG_QRENCODE=y
CONFIG_TPMTOTP=y
#HOTP based remote attestation for supported USB Security dongle
#With/Without TPM support
CONFIG_HOTPKEY=y

#Nitrokey Storage admin tool
CONFIG_NKSTORECLI=n

#GUI Support
#Console based Whiptail support(Console based, no FB):
CONFIG_SLANG=y
CONFIG_NEWT=y
#FBWhiptail based (Graphical):
#CONFIG_CAIRO=y
#CONFIG_FBWHIPTAIL=y

#Additional tools:
#SSH server (requires ethernet drivers, eg: CONFIG_LINUX_E1000E)
CONFIG_DROPBEAR=n
#Ethernet driver (Heads only)
CONFIG_LINUX_E1000E=n

export CONFIG_BOOTSCRIPT=/bin/gui-init
export CONFIG_BOOT_REQ_HASH=n
export CONFIG_BOOT_REQ_ROLLBACK=n
export CONFIG_BOOT_KERNEL_ADD="intel_iommu=on intel_iommu=igfx_off"
export CONFIG_BOOT_KERNEL_REMOVE="quiet"
export CONFIG_BOOT_DEV="/dev/sda1"
export CONFIG_BOARD_NAME="Thinkpad X230-hotp"
export CONFIG_FLASHROM_OPTIONS="--force --noverify-all -p internal --ifd --image bios"

# This board has two SPI flash chips, an 8 MB that holds the IFD,
# the ME image and part of the coreboot image, and a 4 MB one that
# has the rest of the coreboot and the reset vector.
#
```

OEM/Legacy BIOS

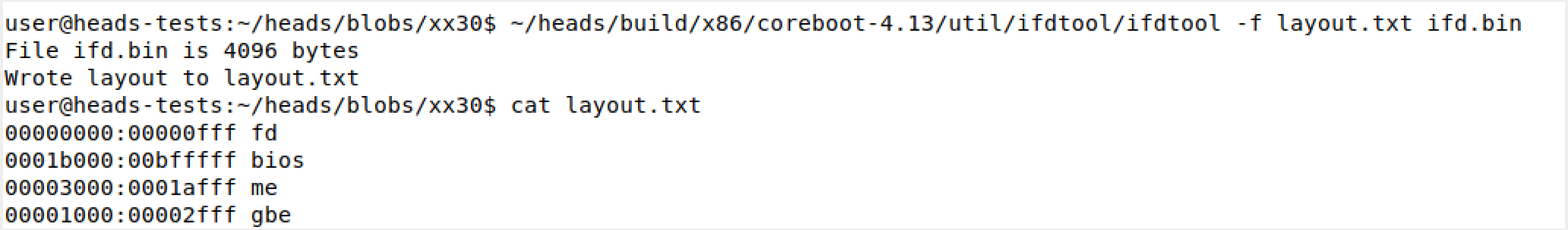
x230-flash/x230-hotp-verification (legacy example)



```
export CONFIG_FLASHROM_OPTIONS="--force --noverify-all -p internal --ifd --image bios"

# This board has two SPI flash chips, an 8 MB that holds the IFD,
# the ME image and part of the coreboot image, and a 4 MB one that
# has the rest of the coreboot and the reset vector.
#
# Only flashing to the bios region is safe to do. The easiest is to
# flash internally when the IFD is unlocked for writing, and x230-flash
# is installed first.
```

Maximized boards

- **Contains unlocked + modified IFD**



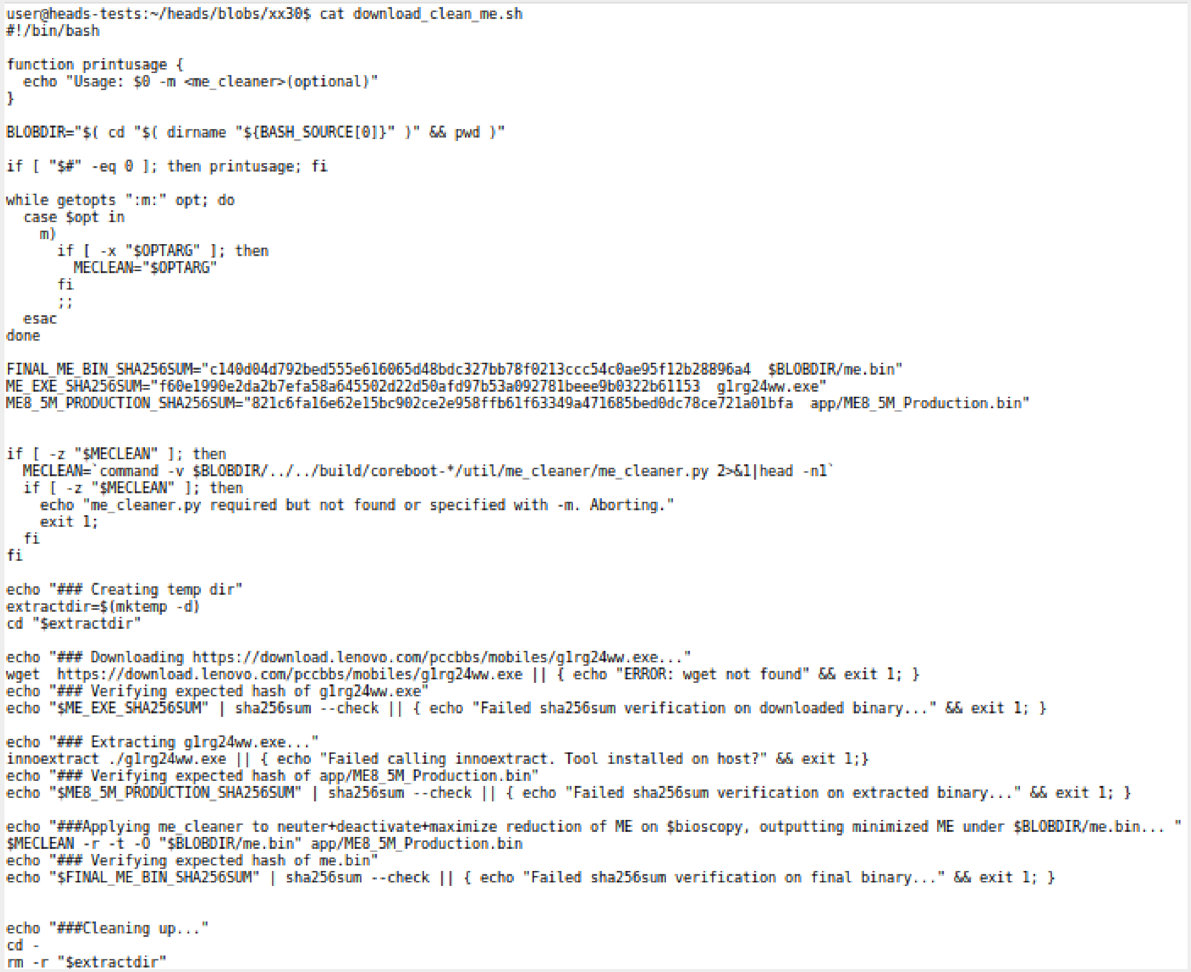
```
user@heads-tests:~/heads/blobs/xx30$ ~/heads/build/x86/coreboot-4.13/util/ifdtool/ifdtool -f layout.txt ifd.bin
File ifd.bin is 4096 bytes
Wrote layout to layout.txt
user@heads-tests:~/heads/blobs/xx30$ cat layout.txt
00000000:00000fff fd
0001b000:00bfffff bios
00003000:0001afff me
00001000:00002fff gbe
```

  - **00bfffff – 0001b000 = BE4FFF**
    - **Can be maximized even more since should match coreboot's CBFS_SIZE!**
- **Contains Neutered ME**
- **Contains generated GBE (per specs)**

Maximized boards

- **Blobs redistribution legal issue dodging**
  - **Github contains download+extraction code**
  - **CircleCI downloads and stitches ROMs**
  - **No blobs are hosted**

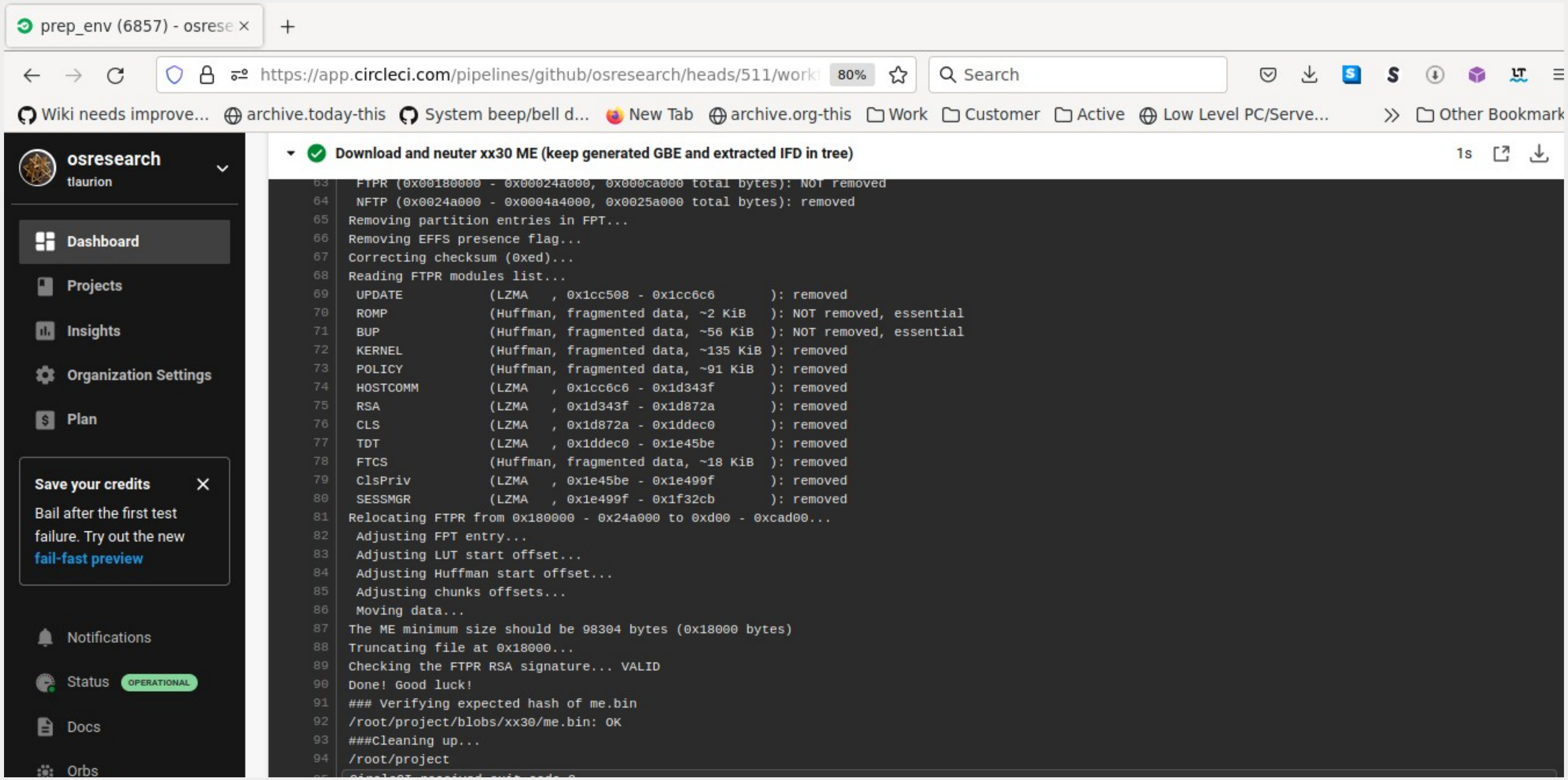
Maximized boards



```
user@heads-tests:~/heads/blobs/xx30$ cat download_clean_me.sh
#!/bin/bash

function printusage {
  echo "Usage: $0 -m <me_cleaner>(optional)"
}

BLOBDIR="$( cd "$( dirname "${BASH_SOURCE[0]}" )" && pwd )"

if [ "$#" -eq 0 ]; then printusage; fi

while getopts ":m:" opt; do
  case $opt in
    m)
      if [ -x "$OPTARG" ]; then
        MECLEAN="$OPTARG"
      fi
      ;;
  esac
done

FINAL_ME_BIN_SHA256SUM="c140d04d792bed555e616065d48bdc327bb78f0213ccc54c0ae95f12b28896a4  $BLOBDIR/me.bin"
ME_EXE_SHA256SUM="f60e1990e2da2b7efa58a645502d22d50afd97b53a092781beee9b0322b61153  g1rg24ww.exe"
ME8_5M_PRODUCTION_SHA256SUM="821c6fa16e62e15bc902ce2e958ffb61f63349a471685bed0dc78ce721a01bfa  app/ME8_5M_Production.bin"


if [ -z "$MECLEAN" ]; then
  MECLEAN=`command -v $BLOBDIR/../../build/coreboot-*/util/me_cleaner/me_cleaner.py 2>&1|head -n1`
  if [ -z "$MECLEAN" ]; then
    echo "me_cleaner.py required but not found or specified with -m. Aborting."
    exit 1;
  fi
fi

echo "### Creating temp dir"
extractdir=$(mktemp -d)
cd "$extractdir"

echo "### Downloading https://download.lenovo.com/pccbbs/mobiles/g1rg24ww.exe..."
wget  https://download.lenovo.com/pccbbs/mobiles/g1rg24ww.exe || { echo "ERROR: wget not found" && exit 1; }
echo "### Verifying expected hash of g1rg24ww.exe"
echo "$ME_EXE_SHA256SUM" | sha256sum --check || { echo "Failed sha256sum verification on downloaded binary..." && exit 1; }

echo "### Extracting g1rg24ww.exe..."
innoextract ./g1rg24ww.exe || { echo "Failed calling innoextract. Tool installed on host?" && exit 1;}
echo "### Verifying expected hash of app/ME8_5M_Production.bin"
echo "$ME8_5M_PRODUCTION_SHA256SUM" | sha256sum --check || { echo "Failed sha256sum verification on extracted binary..." && exit 1; }

echo "###Applying me_cleaner to neuter+deactivate+maximize reduction of ME on $bioscopy, outputting minimized ME under $BLOBDIR/me.bin... "
$MECLEAN -r -t -O "$BLOBDIR/me.bin" app/ME8_5M_Production.bin
echo "### Verifying expected hash of me.bin"
echo "$FINAL_ME_BIN_SHA256SUM" | sha256sum --check || { echo "Failed sha256sum verification on final binary..." && exit 1; }


echo "###Cleaning up..."
cd -
rm -r "$extractdir"
```

Maximized boards

CircleCI downloads/clean/put blobs in place



Save your credits

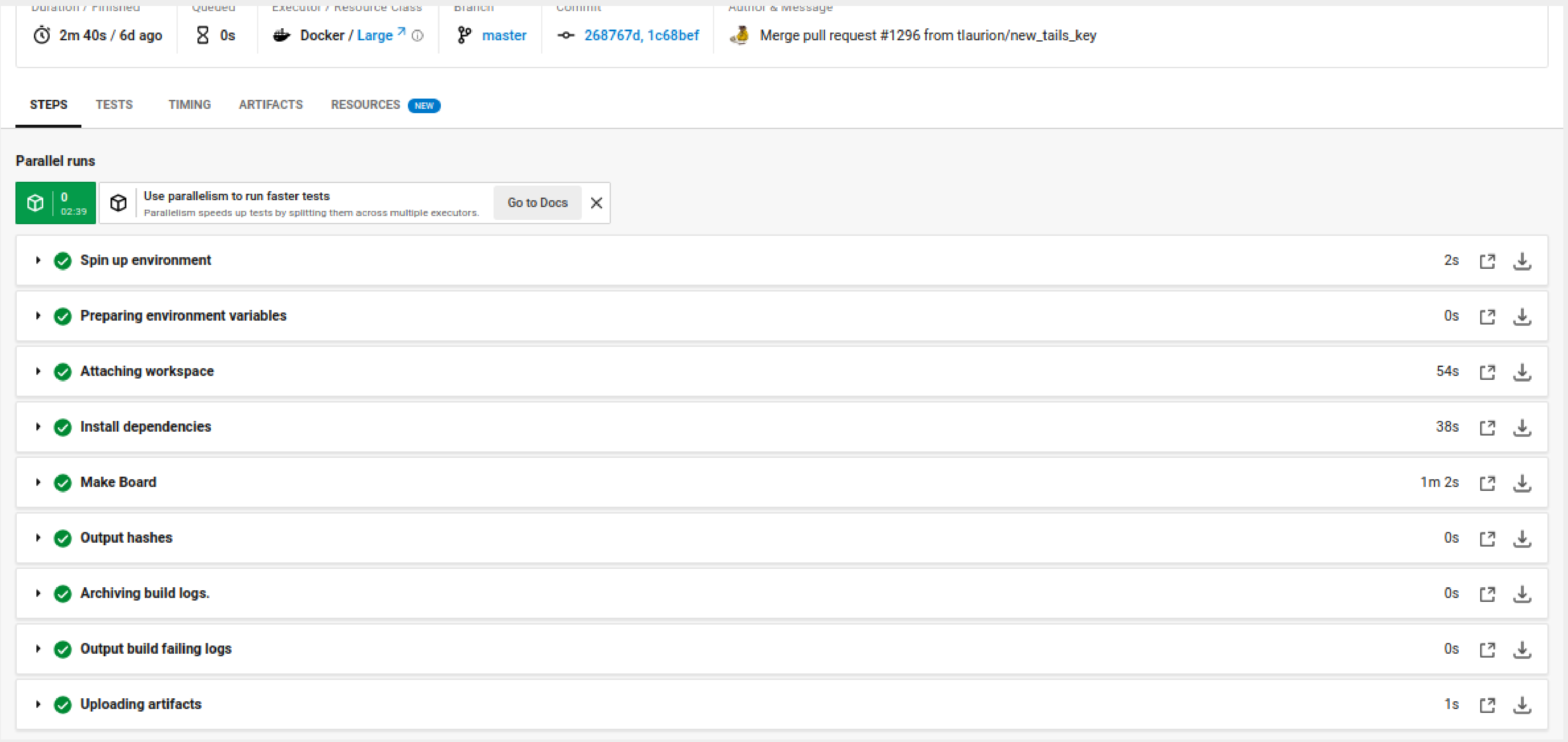
Bail after the first test failure. Try out the new fail-fast preview

Download and neuter xx30 ME (keep generated GBE and extracted IFD in tree)

```
 FTPR (0x00180000 - 0x00024a000, 0x000ca000 total bytes): NOT removed
 NFTP (0x0024a000 - 0x0004a4000, 0x0025a000 total bytes): removed
Removing partition entries in FPT...
Removing EFFS presence flag...
Correcting checksum (0xed)...
Reading FTPR modules list...
 UPDATE           (LZMA    , 0x1cc508 - 0x1cc6c6       ): removed
 ROMP             (Huffman, fragmented data, ~2 KiB    ): NOT removed, essential
 BUP              (Huffman, fragmented data, ~56 KiB   ): NOT removed, essential
 KERNEL           (Huffman, fragmented data, ~135 KiB ): removed
 POLICY           (Huffman, fragmented data, ~91 KiB   ): removed
 HOSTCOMM         (LZMA    , 0x1cc6c6 - 0x1d343f       ): removed
 RSA              (LZMA    , 0x1d343f - 0x1d872a       ): removed
 CLS              (LZMA    , 0x1d872a - 0x1ddec0       ): removed
 TDT              (LZMA    , 0x1ddec0 - 0x1e45be       ): removed
 FTCS             (Huffman, fragmented data, ~18 KiB   ): removed
 ClsPriv          (LZMA    , 0x1e45be - 0x1e499f       ): removed
 SESSMGR          (LZMA    , 0x1e499f - 0x1f32cb       ): removed
Relocating FTPR from 0x180000 - 0x24a000 to 0xd00 - 0xcad00...
 Adjusting FPT entry...
 Adjusting LUT start offset...
 Adjusting Huffman start offset...
 Adjusting chunks offsets...
 Moving data...
The ME minimum size should be 98304 bytes (0x18000 bytes)
Truncating file at 0x18000...
Checking the FTPR RSA signature... VALID
Done! Good luck!
### Verifying expected hash of me.bin
/root/project/blobs/xx30/me.bin: OK
###Cleaning up...
/root/project
```

Maximized boards

CircleCI stitches the ROM together



Duration / Finished 2m 40s / 6d ago | Queued 0s | Executor / Resource Class Docker / Large | Branch master | Commit 268767d, 1c68bef | Author & Message Merge pull request #1296 from tlaurion/new_tails_key

STEPS TESTS TIMING ARTIFACTS RESOURCES NEW

Parallel runs

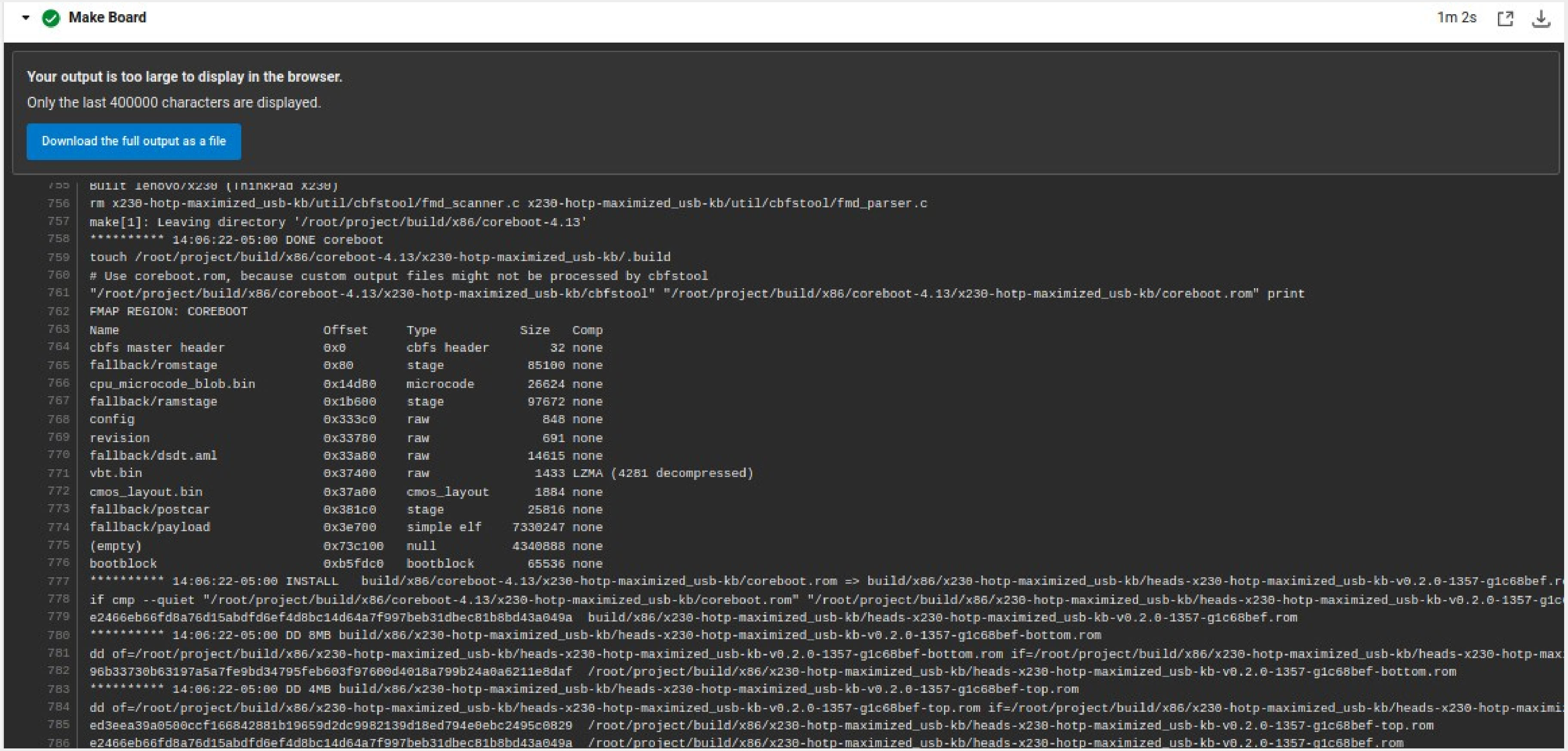
0 02:39

Use parallelism to run faster tests
Parallelism speeds up tests by splitting them across multiple executors.
Go to Docs

Spin up environment	2s
Preparing environment variables	0s
Attaching workspace	54s
Install dependencies	38s
Make Board	1m 2s
Output hashes	0s
Archiving build logs.	0s
Output build failing logs	0s
Uploading artifacts	1s

Maximized boards

CircleCI stitches the ROM together

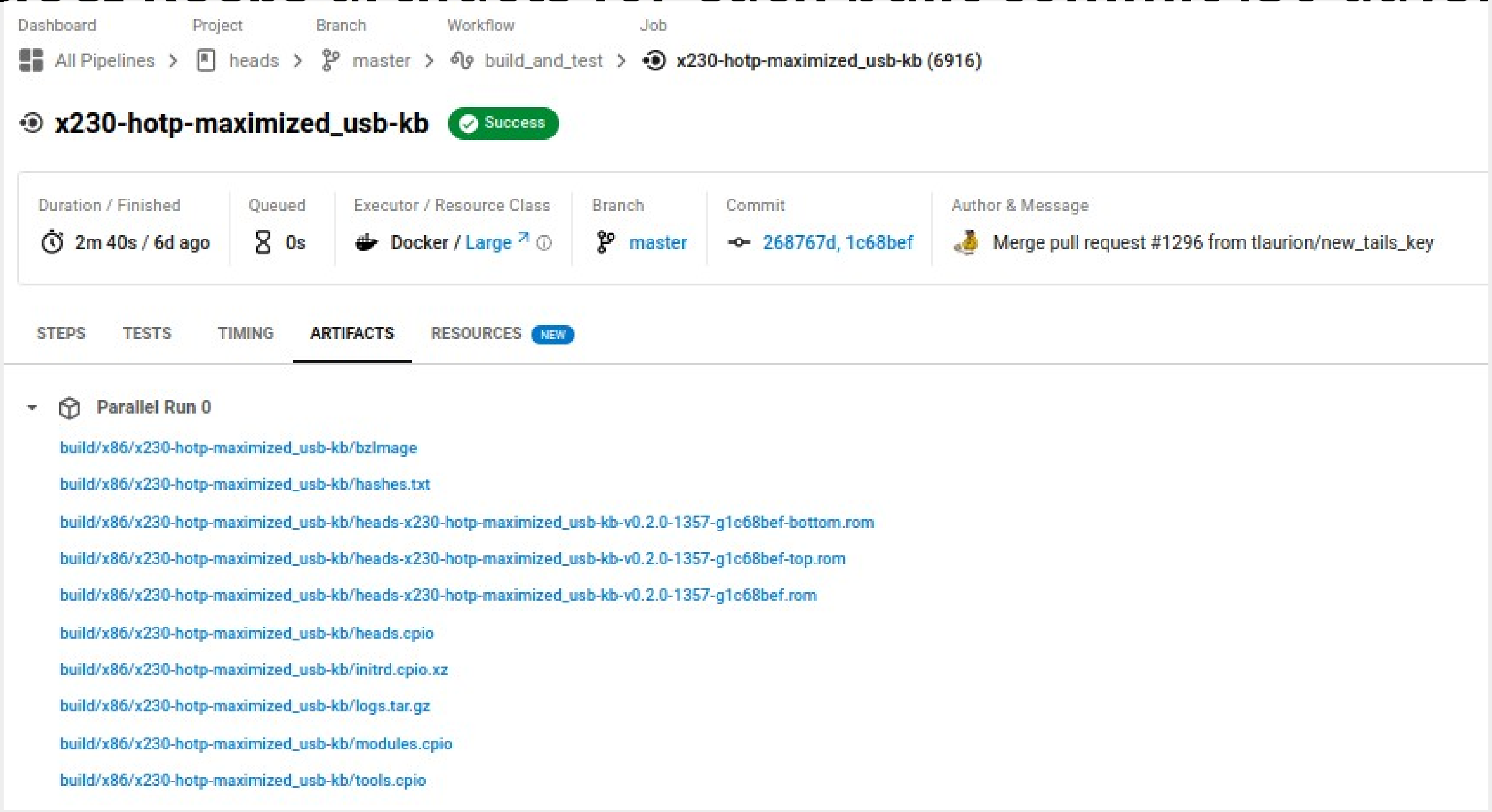


Your output is too large to display in the browser.
Only the last 400000 characters are displayed.

```
Built lenovo/x230 (ThinkPad X230)
rm x230-hotp-maximized_usb-kb/util/cbfstool/fmd_scanner.c x230-hotp-maximized_usb-kb/util/cbfstool/fmd_parser.c
make[1]: Leaving directory '/root/project/build/x86/coreboot-4.13'
********** 14:06:22-05:00 DONE coreboot
touch /root/project/build/x86/coreboot-4.13/x230-hotp-maximized_usb-kb/.build
# Use coreboot.rom, because custom output files might not be processed by cbfstool
"/root/project/build/x86/coreboot-4.13/x230-hotp-maximized_usb-kb/cbfstool" "/root/project/build/x86/coreboot-4.13/x230-hotp-maximized_usb-kb/coreboot.rom" print
FMAP REGION: COREBOOT
Name                           Offset     Type           Size   Comp
cbfs master header             0x0        cbfs header        32 none
fallback/romstage              0x80       stage           85100 none
cpu_microcode_blob.bin         0x14d80    microcode       26624 none
fallback/ramstage              0x1b600    stage           97672 none
config                         0x333c0    raw               848 none
revision                       0x33780    raw               691 none
fallback/dsdt.aml              0x33a80    raw             14615 none
vbt.bin                        0x37400    raw              1433 LZMA (4281 decompressed)
cmos_layout.bin                0x37a00    cmos_layout      1884 none
fallback/postcar               0x381c0    stage           25816 none
fallback/payload               0x3e700    simple elf    7330247 none
(empty)                        0x73c100   null          4340888 none
bootblock                      0xb5fdc0   bootblock       65536 none
********** 14:06:22-05:00 INSTALL   build/x86/coreboot-4.13/x230-hotp-maximized_usb-kb/coreboot.rom => build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef.r
if cmp --quiet "/root/project/build/x86/coreboot-4.13/x230-hotp-maximized_usb-kb/coreboot.rom" "/root/project/build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c
e2466eb66fd8a76d15abdfd6ef4d8bc14d64a7f997beb31dbec81b8bd43a049a  build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef.rom
********** 14:06:22-05:00 DD 8MB build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef-bottom.rom
dd of=/root/project/build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef-bottom.rom if=/root/project/build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-max
96b33730b63197a5a7fe9bd34795feb603f97600d4018a799b24a0a6211e8daf  /root/project/build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef-bottom.rom
********** 14:06:22-05:00 DD 4MB build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef-top.rom
dd of=/root/project/build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef-top.rom if=/root/project/build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximi
ed3eea39a0500ccf166842881b19659d2dc9982139d18ed794e0ebc2495c0829  /root/project/build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef-top.rom
e2466eb66fd8a76d15abdfd6ef4d8bc14d64a7f997beb31dbec81b8bd43a049a  /root/project/build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef.rom
```

Maximized boards

CircleCI keeps artifacts for each built commit (30 days)



Dashboard / Project / Branch / Workflow / Job

All Pipelines > heads > master > build_and_test > x230-hotp-maximized_usb-kb (6916)

x230-hotp-maximized_usb-kb Success

Duration / Finished	Queued	Executor / Resource Class	Branch	Commit	Author & Message
2m 40s / 6d ago	0s	Docker / Large	master	268767d, 1c68bef	Merge pull request #1296 from tlaurion/new_tails_key

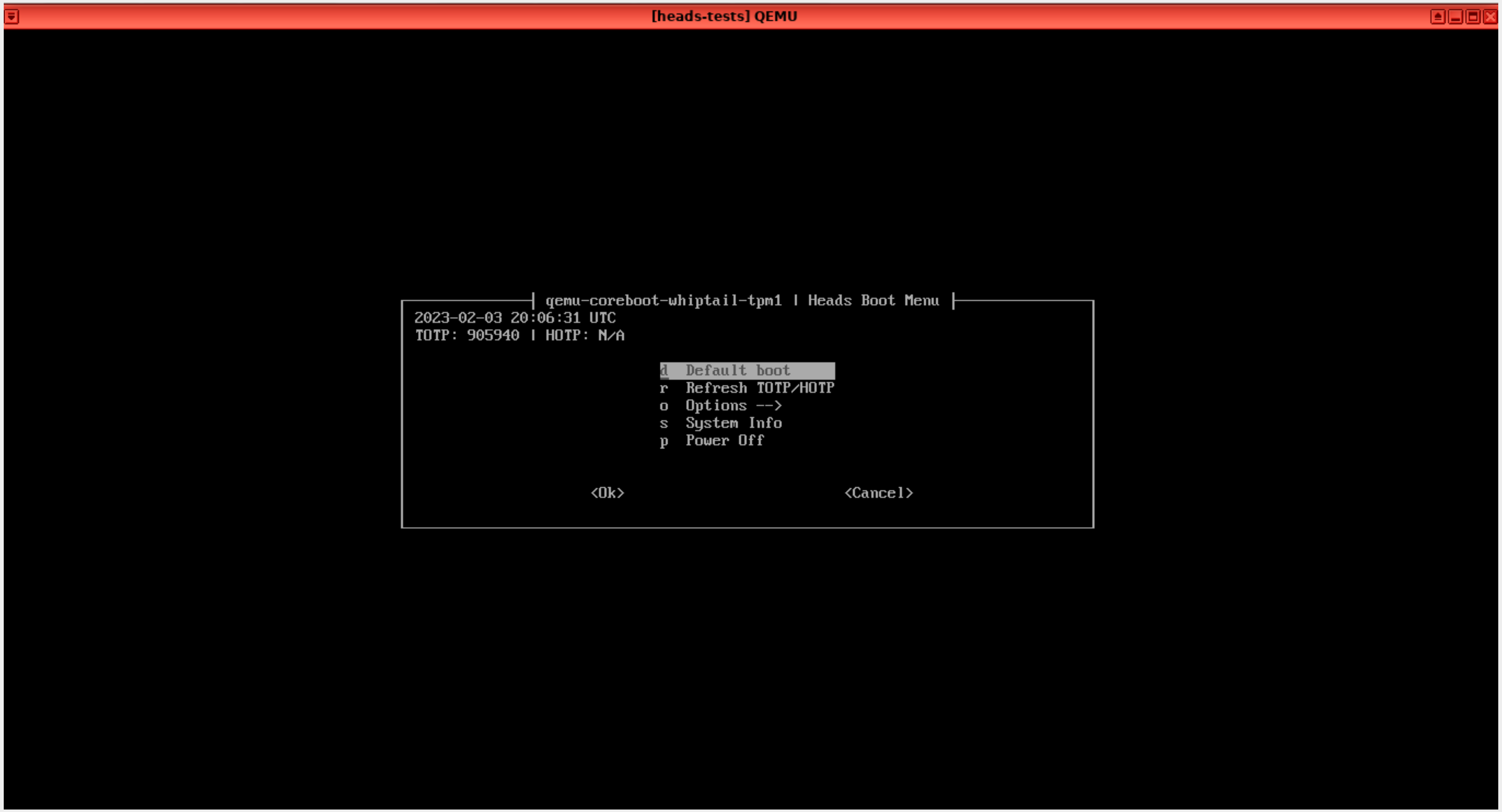
Parallel Run 0

- build/x86/x230-hotp-maximized_usb-kb/bzImage
- build/x86/x230-hotp-maximized_usb-kb/hashes.txt
- build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef-bottom.rom
- build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef-top.rom
- build/x86/x230-hotp-maximized_usb-kb/heads-x230-hotp-maximized_usb-kb-v0.2.0-1357-g1c68bef.rom
- build/x86/x230-hotp-maximized_usb-kb/heads.cpio
- build/x86/x230-hotp-maximized_usb-kb/initrd.cpio.xz
- build/x86/x230-hotp-maximized_usb-kb/logs.tar.gz
- build/x86/x230-hotp-maximized_usb-kb/modules.cpio
- build/x86/x230-hotp-maximized_usb-kb/tools.cpio

What’s new

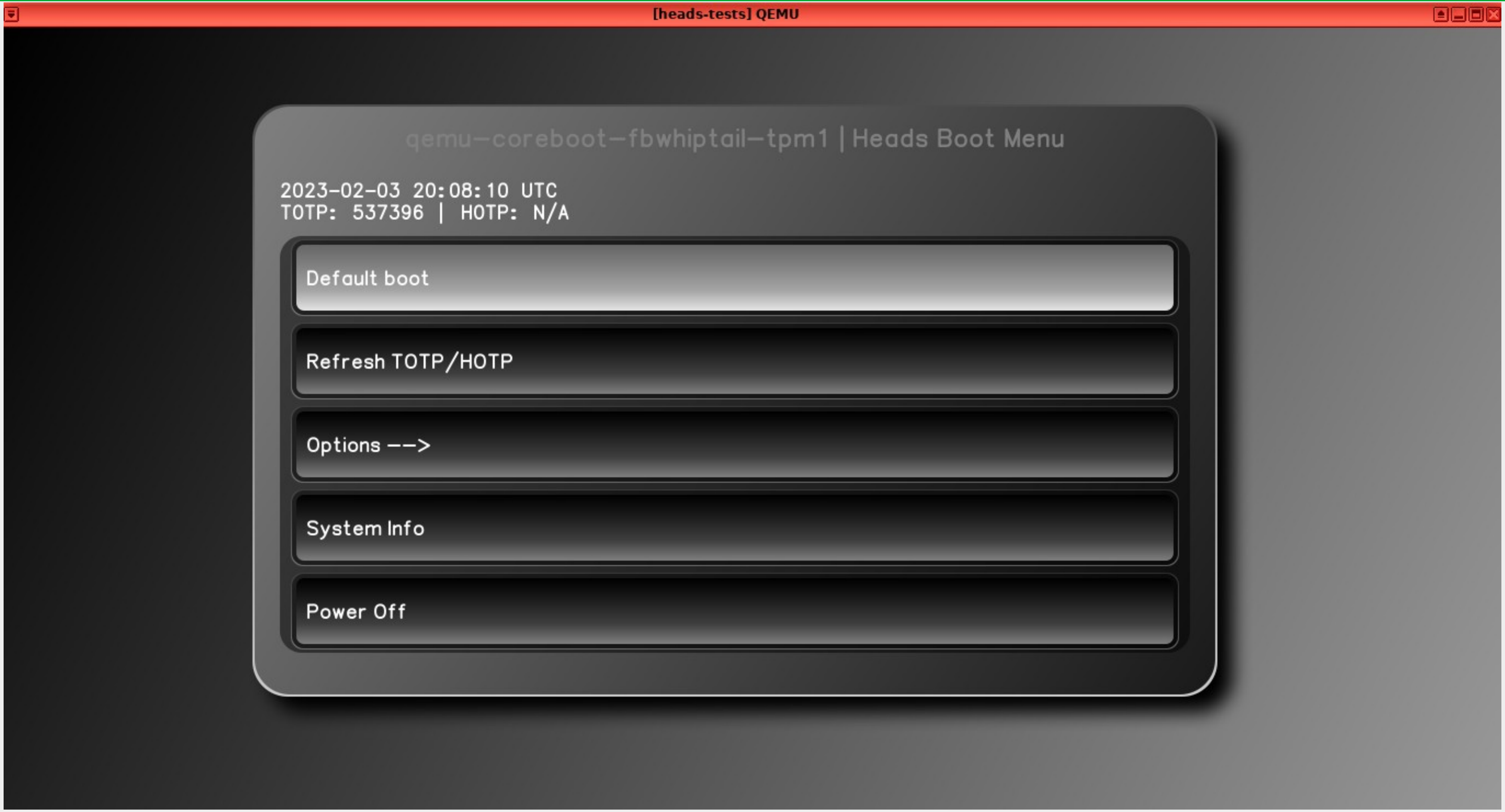
- Maximized boards vs Legacy boards, or how to dodge blob redistribution legal limitations
- **Whiptail/FBWhiptail: one graphical interface (GUI) to rule them all**
- OEM Factory reset/Re-Ownership wizard upstreamed
- QEMU/KVM board configurations with swtpm and USB Security dongle support to ease development/testing

Whiptail : console (server/BMC)



```
┤ qemu-coreboot-whiptail-tpm1 | Heads Boot Menu ├
2023-02-03 20:06:31 UTC
TOTP: 905940 | HOTP: N/A

                         d  Default boot
                         r  Refresh TOTP/HOTP
                         o  Options -->
                         s  System Info
                         p  Power Off

              <Ok>                          <Cancel>
```

FBWhiptail : Desktop/Laptops



[heads-tests] QEMU

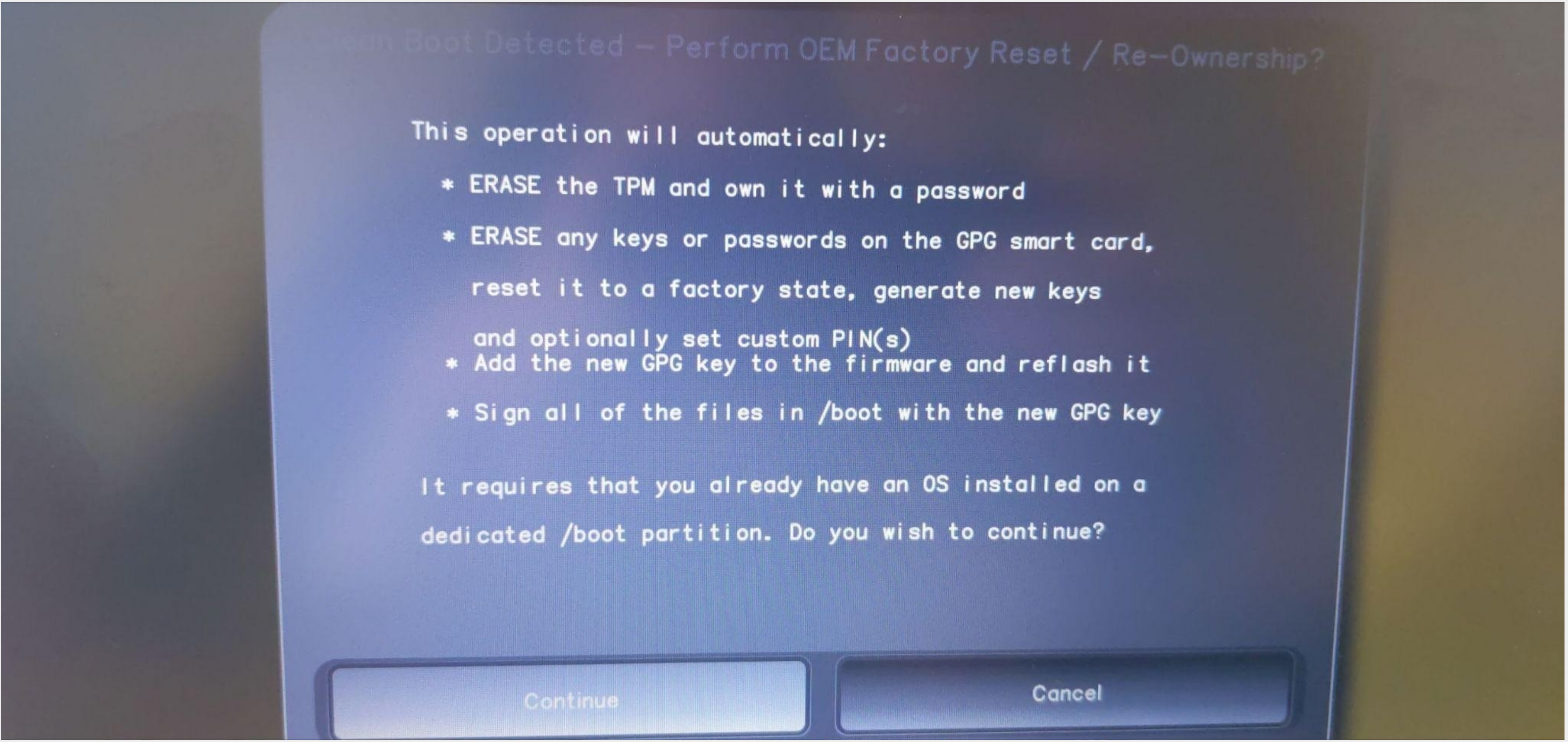
qemu-coreboot-fbwhiptail-tpm1 | Heads Boot Menu

```
2023-02-03 20:08:10 UTC
TOTP: 537396 | HOTP: N/A
```

What’s new

- Maximized boards vs Legacy boards, or how to dodge blob redistribution legal limitations
- Whiptail/FBWhiptail: one graphical interface (GUI) to rule them all
- **OEM Factory reset/Re-Ownership wizard upstreamed**
- QEMU/KVM board configurations with swtpm and USB Security dongle support to ease development/testing

OEM Factory Reset / Re-Ownership wizard



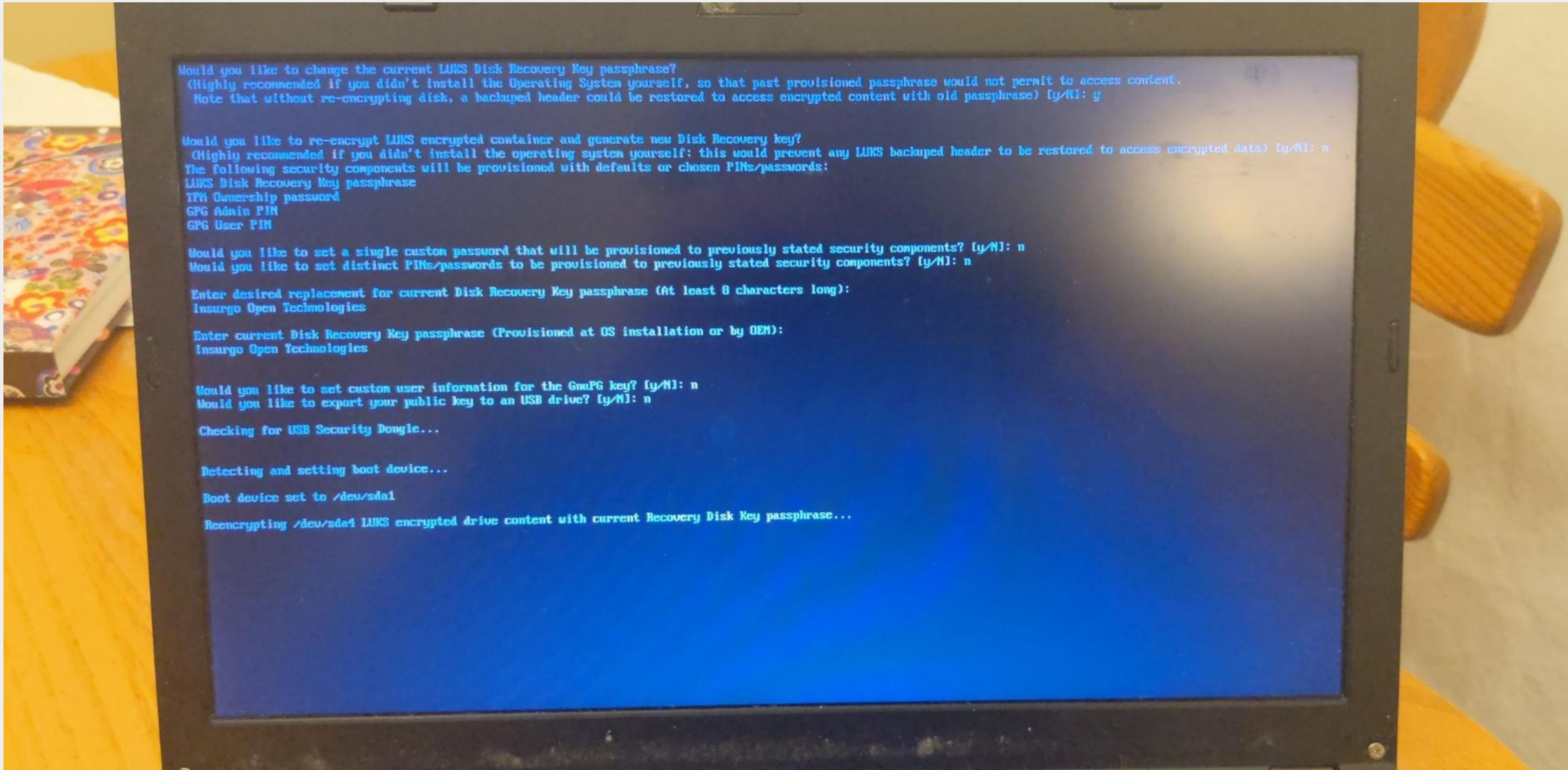
Boot Detected – Perform OEM Factory Reset / Re-Ownership?

This operation will automatically:

* ERASE the TPM and own it with a password
* ERASE any keys or passwords on the GPG smart card, reset it to a factory state, generate new keys and optionally set custom PIN(s)
* Add the new GPG key to the firmware and reflash it
* Sign all of the files in /boot with the new GPG key

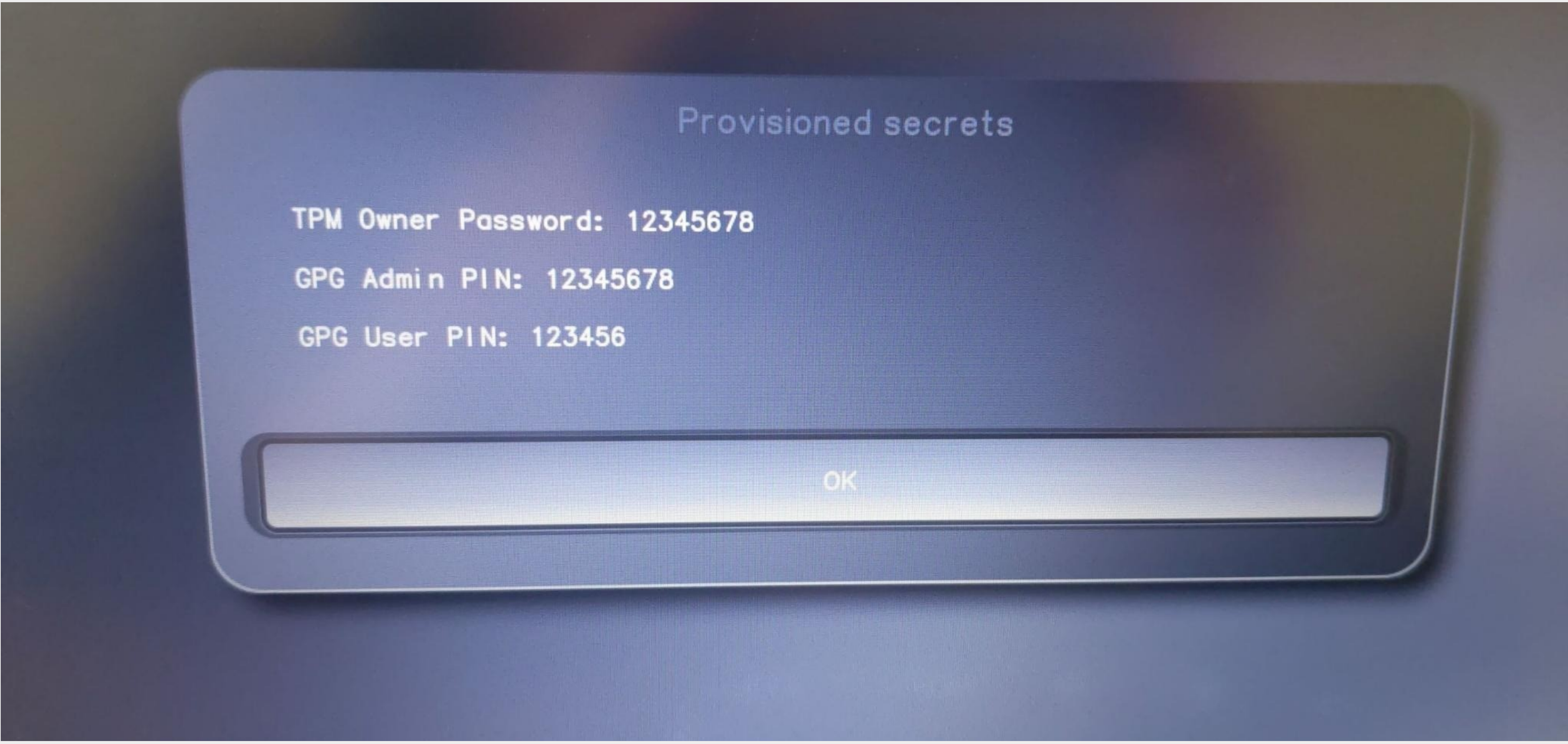
It requires that you already have an OS installed on a dedicated /boot partition. Do you wish to continue?

OEM Factory Reset / Re-Ownership wizard



```
Would you like to change the current LUKS Disk Recovery Key passphrase?
 (Highly recommended if you didn't install the Operating System yourself, so that past provisioned passphrase would not permit to access content.
  Note that without re-encrypting disk, a backuped header could be restored to access encrypted content with old passphrase) [y/N]: y


Would you like to re-encrypt LUKS encrypted container and generate new Disk Recovery key?
 (Highly recommended if you didn't install the operating system yourself: this would prevent any LUKS backuped header to be restored to access encrypted data) [y/N]: n
The following security components will be provisioned with defaults or chosen PINs/passwords:
LUKS Disk Recovery Key passphrase
TPM Ownership password
GPG Admin PIN
GPG User PIN

Would you like to set a single custom password that will be provisioned to previously stated security components? [y/N]: n
Would you like to set distinct PINs/passwords to be provisioned to previously stated security components? [y/N]: n

Enter desired replacement for current Disk Recovery Key passphrase (At least 8 characters long):
Insurgo Open Technologies

Enter current Disk Recovery Key passphrase (Provisioned at OS installation or by OEM):
Insurgo Open Technologies


Would you like to set custom user information for the GnuPG key? [y/N]: n
Would you like to export your public key to an USB drive? [y/N]: n

Checking for USB Security Dongle...


Detecting and setting boot device...

Boot device set to /dev/sda1

Reencrypting /dev/sda4 LUKS encrypted drive content with current Recovery Disk Key passphrase...
```

OEM Factory Reset / Re-Ownership wizard



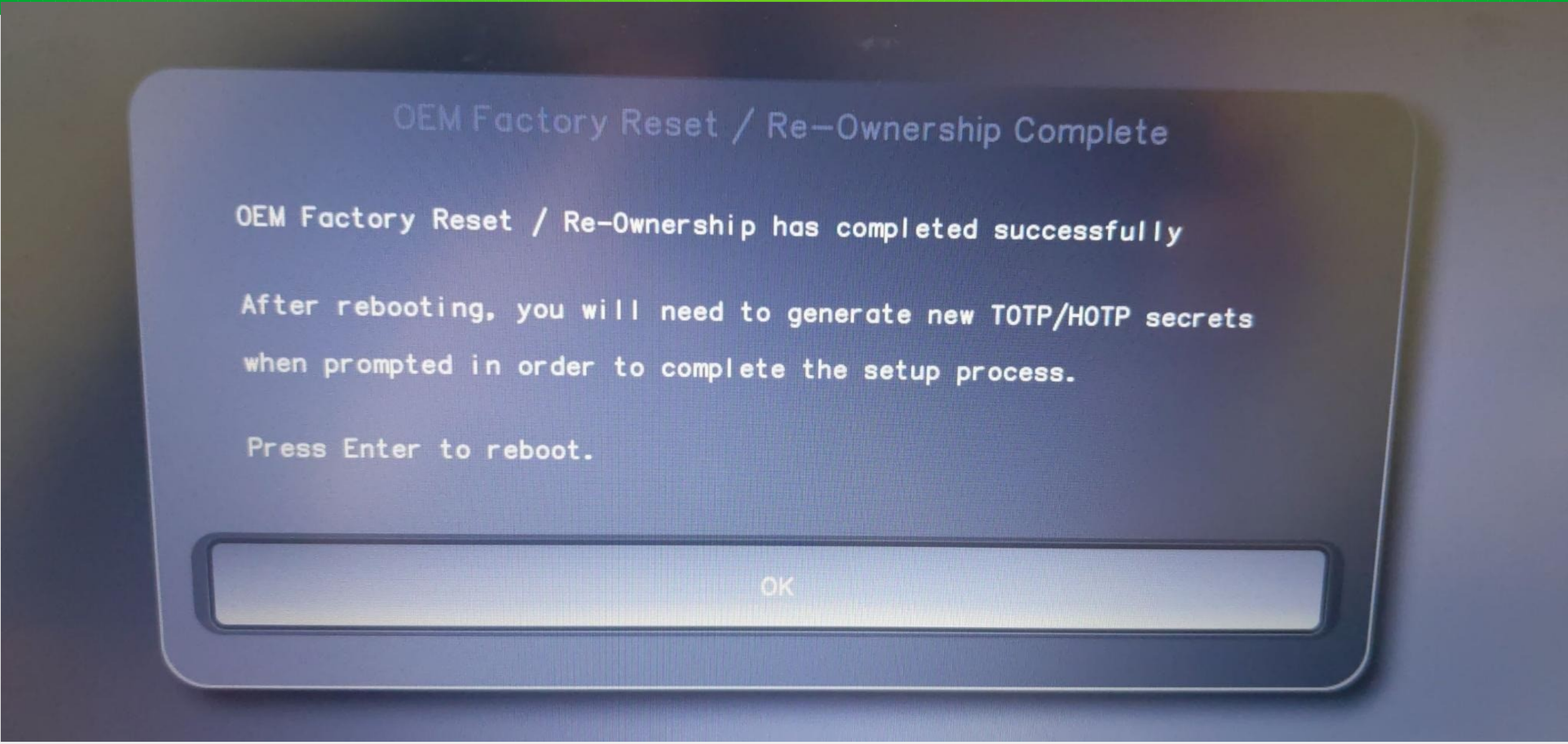
Provisioned secrets

TPM Owner Password: 12345678

GPG Admin PIN: 12345678

GPG User PIN: 123456

OEM Factory Reset / Re-Ownership wizard



OEM Factory Reset / Re–Ownership Complete

OEM Factory Reset / Re–Ownership has completed successfully

After rebooting, you will need to generate new TOTP/HOTP secrets when prompted in order to complete the setup process.

Press Enter to reboot.

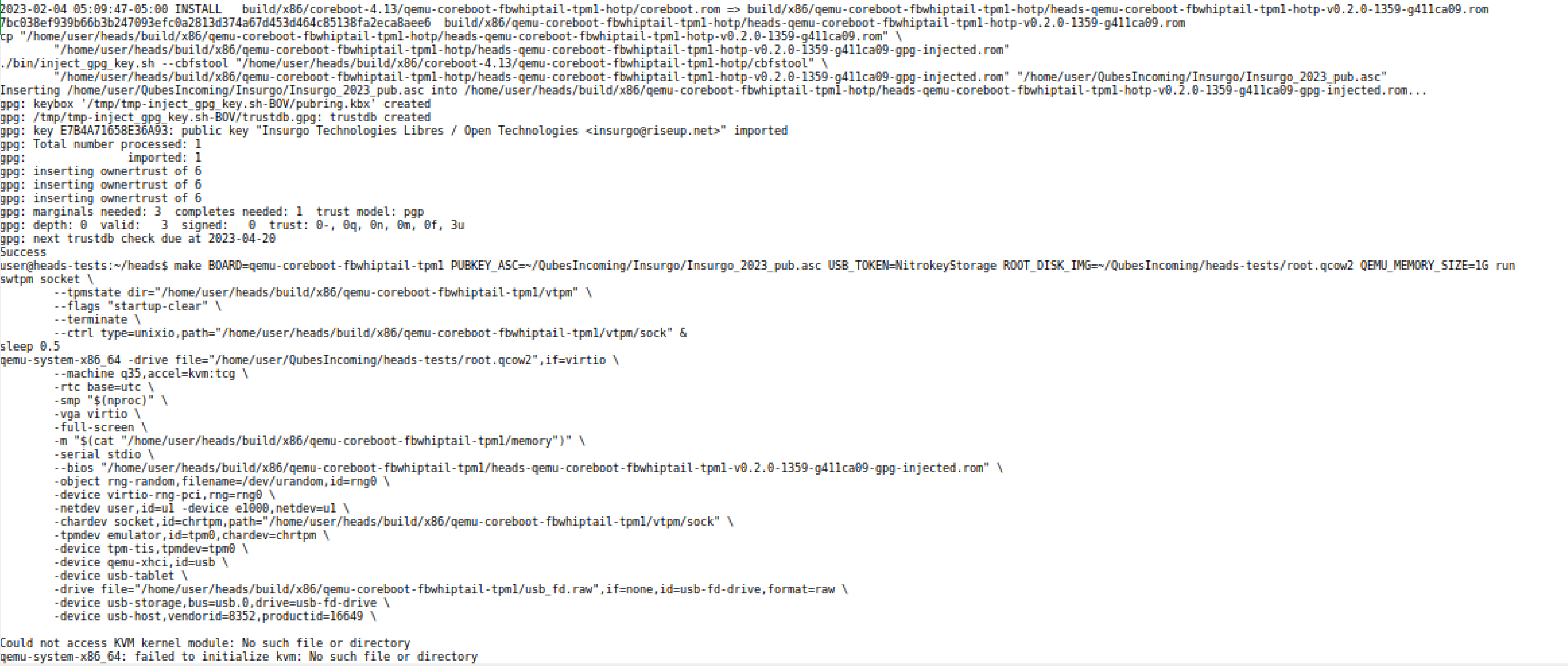
What’s new

- Maximized boards vs Legacy boards, or how to dodge blob redistribution legal limitations
- Whiptail/FBWhiptail: one graphical interface (GUI) to rule them all
- OEM Factory reset/Re-Ownership wizard upstreamed
- **QEMU/KVM board configurations with swtpm and USB Security dongle support to ease development/testing**

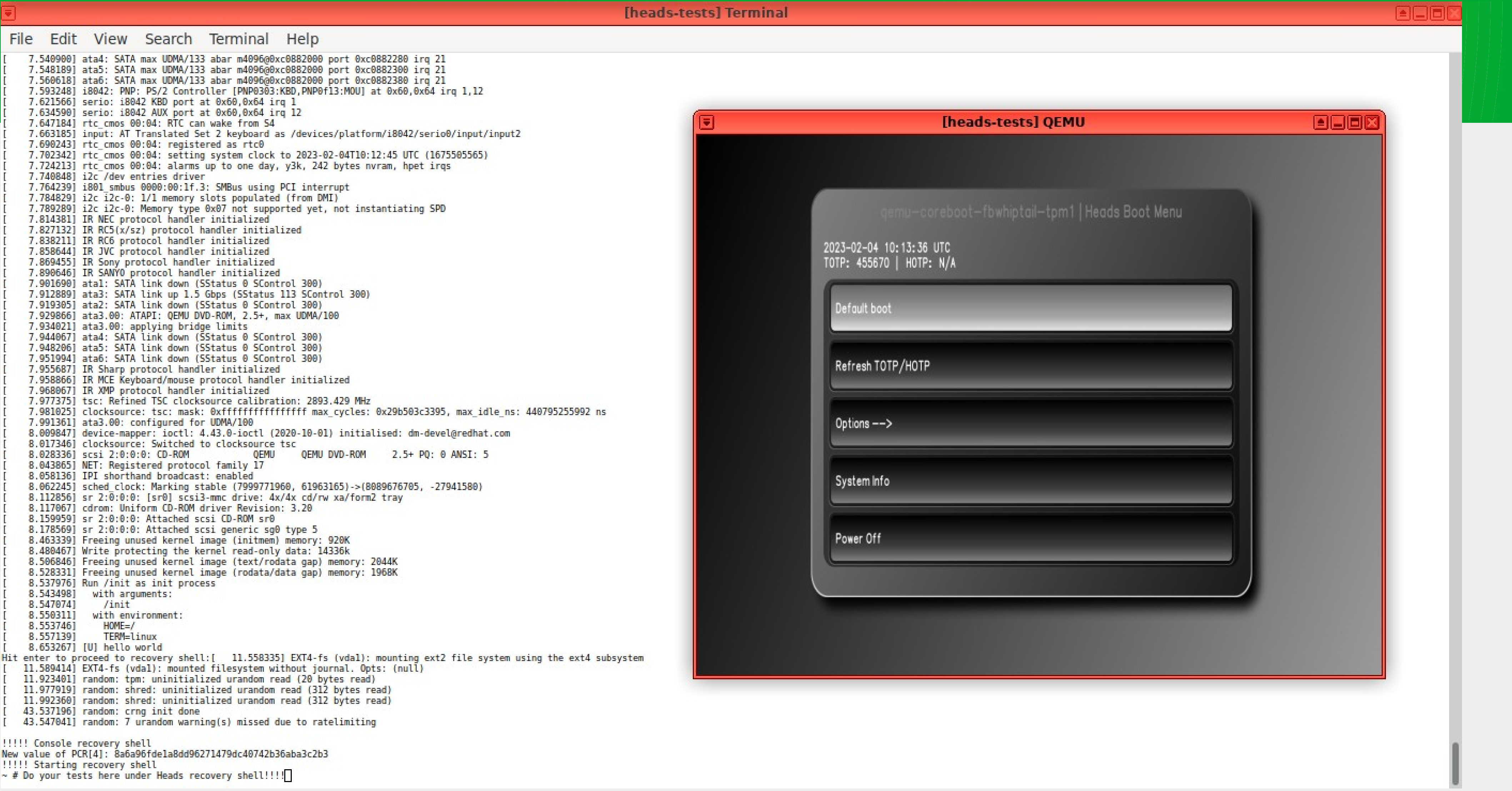
QEMU/KVM boards

- Permit easy testing and debugging (debian-12 recommended)
  - After having followed instructions: https://osresearch.net/Emulating-Heads/#comprehensive-test
    - make BOARD=qemu-coreboot-fbwhiptail-tpm2 PUBKEY_ASC=~/QubesIncoming/Insurgo/Insurgo_2023_pub.asc USB_TOKEN=NitrokeyStorage ROOT_DISK_IMG=~/QubesIncoming/heads-tests/root.qcow2 QEMU_MEMORY_SIZE=1G inject_gpg
    - make BOARD=qemu-coreboot-fbwhiptail-tpm2 PUBKEY_ASC=~/QubesIncoming/Insurgo/Insurgo_2023_pub.asc USB_TOKEN=NitrokeyStorage ROOT_DISK_IMG=~/QubesIncoming/heads-tests/root.qcow2 QEMU_MEMORY_SIZE=1G run

QEMU/KVM boards



```
2023-02-04 05:09:47-05:00 INSTALL   build/x86/coreboot-4.13/qemu-coreboot-fbwhiptail-tpm1-hotp/coreboot.rom => build/x86/qemu-coreboot-fbwhiptail-tpm1-hotp/heads-qemu-coreboot-fbwhiptail-tpm1-hotp-v0.2.0-1359-g411ca09.rom
7bc038ef939b66b3b247093efc0a2813d374a67d453d464c85138fa2eca8aee6  build/x86/qemu-coreboot-fbwhiptail-tpm1-hotp/heads-qemu-coreboot-fbwhiptail-tpm1-hotp-v0.2.0-1359-g411ca09.rom
cp "/home/user/heads/build/x86/qemu-coreboot-fbwhiptail-tpm1-hotp/heads-qemu-coreboot-fbwhiptail-tpm1-hotp-v0.2.0-1359-g411ca09.rom" \
        "/home/user/heads/build/x86/qemu-coreboot-fbwhiptail-tpm1-hotp/heads-qemu-coreboot-fbwhiptail-tpm1-hotp-v0.2.0-1359-g411ca09-gpg-injected.rom"
./bin/inject_gpg_key.sh --cbfstool "/home/user/heads/build/x86/coreboot-4.13/qemu-coreboot-fbwhiptail-tpm1-hotp/cbfstool" \
        "/home/user/heads/build/x86/qemu-coreboot-fbwhiptail-tpm1-hotp/heads-qemu-coreboot-fbwhiptail-tpm1-hotp-v0.2.0-1359-g411ca09-gpg-injected.rom" "/home/user/QubesIncoming/Insurgo/Insurgo_2023_pub.asc"
Inserting /home/user/QubesIncoming/Insurgo/Insurgo_2023_pub.asc into /home/user/heads/build/x86/qemu-coreboot-fbwhiptail-tpm1-hotp/heads-qemu-coreboot-fbwhiptail-tpm1-hotp-v0.2.0-1359-g411ca09-gpg-injected.rom...
gpg: keybox '/tmp/tmp-inject_gpg_key.sh-BOV/pubring.kbx' created
gpg: /tmp/tmp-inject_gpg_key.sh-BOV/trustdb.gpg: trustdb created
gpg: key E7B4A71658E36A93: public key "Insurgo Technologies Libres / Open Technologies <insurgo@riseup.net>" imported
gpg: Total number processed: 1
gpg:               imported: 1
gpg: inserting ownertrust of 6
gpg: inserting ownertrust of 6
gpg: inserting ownertrust of 6
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid:   3  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 3u
gpg: next trustdb check due at 2023-04-20
Success
user@heads-tests:~/heads$ make BOARD=qemu-coreboot-fbwhiptail-tpm1 PUBKEY_ASC=~/QubesIncoming/Insurgo/Insurgo_2023_pub.asc USB_TOKEN=NitrokeyStorage ROOT_DISK_IMG=~/QubesIncoming/heads-tests/root.qcow2 QEMU_MEMORY_SIZE=1G run
swtpm socket \
        --tpmstate dir="/home/user/heads/build/x86/qemu-coreboot-fbwhiptail-tpm1/vtpm" \
        --flags "startup-clear" \
        --terminate \
        --ctrl type=unixio,path="/home/user/heads/build/x86/qemu-coreboot-fbwhiptail-tpm1/vtpm/sock" &
sleep 0.5
qemu-system-x86_64 -drive file="/home/user/QubesIncoming/heads-tests/root.qcow2",if=virtio \
        --machine q35,accel=kvm:tcg \
        -rtc base=utc \
        -smp "$(nproc)" \
        -vga virtio \
        -full-screen \
        -m "$(cat "/home/user/heads/build/x86/qemu-coreboot-fbwhiptail-tpm1/memory")" \
        -serial stdio \
        --bios "/home/user/heads/build/x86/qemu-coreboot-fbwhiptail-tpm1/heads-qemu-coreboot-fbwhiptail-tpm1-v0.2.0-1359-g411ca09-gpg-injected.rom" \
        -object rng-random,filename=/dev/urandom,id=rng0 \
        -device virtio-rng-pci,rng=rng0 \
        -netdev user,id=u1 -device e1000,netdev=u1 \
        -chardev socket,id=chrtpm,path="/home/user/heads/build/x86/qemu-coreboot-fbwhiptail-tpm1/vtpm/sock" \
        -tpmdev emulator,id=tpm0,chardev=chrtpm \
        -device tpm-tis,tpmdev=tpm0 \
        -device qemu-xhci,id=usb \
        -device usb-tablet \
        -drive file="/home/user/heads/build/x86/qemu-coreboot-fbwhiptail-tpm1/usb_fd.raw",if=none,id=usb-fd-drive,format=raw \
        -device usb-storage,bus=usb.0,drive=usb-fd-drive \
        -device usb-host,vendorid=8352,productid=16649 \

Could not access KVM kernel module: No such file or directory
qemu-system-x86_64: failed to initialize kvm: No such file or directory
```



```
[    7.540900] ata4: SATA max UDMA/133 abar m4096@0xc0882000 port 0xc0882280 irq 21
[    7.548189] ata5: SATA max UDMA/133 abar m4096@0xc0882000 port 0xc0882300 irq 21
[    7.560618] ata6: SATA max UDMA/133 abar m4096@0xc0882000 port 0xc0882380 irq 21
[    7.593248] i8042: PNP: PS/2 Controller [PNP0303:KBD,PNP0f13:MOU] at 0x60,0x64 irq 1,12
[    7.621566] serio: i8042 KBD port at 0x60,0x64 irq 1
[    7.634590] serio: i8042 AUX port at 0x60,0x64 irq 12
[    7.647184] rtc_cmos 00:04: RTC can wake from S4
[    7.663185] input: AT Translated Set 2 keyboard as /devices/platform/i8042/serio0/input/input2
[    7.690243] rtc_cmos 00:04: registered as rtc0
[    7.702342] rtc_cmos 00:04: setting system clock to 2023-02-04T10:12:45 UTC (1675505565)
[    7.724213] rtc_cmos 00:04: alarms up to one day, y3k, 242 bytes nvram, hpet irqs
[    7.740848] i2c /dev entries driver
[    7.764239] i801_smbus 0000:00:1f.3: SMBus using PCI interrupt
[    7.784829] i2c i2c-0: 1/1 memory slots populated (from DMI)
[    7.789289] i2c i2c-0: Memory type 0x07 not supported yet, not instantiating SPD
[    7.814381] IR NEC protocol handler initialized
[    7.827132] IR RC5(x/sz) protocol handler initialized
[    7.838211] IR RC6 protocol handler initialized
[    7.858644] IR JVC protocol handler initialized
[    7.869455] IR Sony protocol handler initialized
[    7.890646] IR SANYO protocol handler initialized
[    7.901690] ata1: SATA link down (SStatus 0 SControl 300)
[    7.912889] ata3: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[    7.919305] ata2: SATA link down (SStatus 0 SControl 300)
[    7.929866] ata3.00: ATAPI: QEMU DVD-ROM, 2.5+, max UDMA/100
[    7.934021] ata3.00: applying bridge limits
[    7.944067] ata4: SATA link down (SStatus 0 SControl 300)
[    7.948206] ata5: SATA link down (SStatus 0 SControl 300)
[    7.951994] ata6: SATA link down (SStatus 0 SControl 300)
[    7.955687] IR Sharp protocol handler initialized
[    7.958866] IR MCE Keyboard/mouse protocol handler initialized
[    7.968067] IR XMP protocol handler initialized
[    7.977375] tsc: Refined TSC clocksource calibration: 2893.429 MHz
[    7.981025] clocksource: tsc: mask: 0xffffffffffffffff max_cycles: 0x29b503c3395, max_idle_ns: 440795255992 ns
[    7.991361] ata3.00: configured for UDMA/100
[    8.009847] device-mapper: ioctl: 4.43.0-ioctl (2020-10-01) initialised: dm-devel@redhat.com
[    8.017346] clocksource: Switched to clocksource tsc
[    8.028336] scsi 2:0:0:0: CD-ROM            QEMU     QEMU DVD-ROM     2.5+ PQ: 0 ANSI: 5
[    8.043865] NET: Registered protocol family 17
[    8.058136] IPI shorthand broadcast: enabled
[    8.062245] sched_clock: Marking stable (7999771960, 61963165)->(8089676705, -27941580)
[    8.112856] sr 2:0:0:0: [sr0] scsi3-mmc drive: 4x/4x cd/rw xa/form2 tray
[    8.117067] cdrom: Uniform CD-ROM driver Revision: 3.20
[    8.159959] sr 2:0:0:0: Attached scsi CD-ROM sr0
[    8.178569] sr 2:0:0:0: Attached scsi generic sg0 type 5
[    8.463339] Freeing unused kernel image (initmem) memory: 920K
[    8.480467] Write protecting the kernel read-only data: 14336k
[    8.506846] Freeing unused kernel image (text/rodata gap) memory: 2044K
[    8.528331] Freeing unused kernel image (rodata/data gap) memory: 1968K
[    8.537976] Run /init as init process
[    8.543498]   with arguments:
[    8.547074]     /init
[    8.550311]   with environment:
[    8.553746]     HOME=/
[    8.557139]     TERM=linux
[    8.653267] [U] hello world
Hit enter to proceed to recovery shell:[   11.558335] EXT4-fs (vda1): mounting ext2 file system using the ext4 subsystem
[   11.589414] EXT4-fs (vda1): mounted filesystem without journal. Opts: (null)
[   11.923401] random: tpm: uninitialized urandom read (20 bytes read)
[   11.977919] random: shred: uninitialized urandom read (312 bytes read)
[   11.992360] random: shred: uninitialized urandom read (312 bytes read)
[   43.537196] random: crng init done
[   43.547041] random: 7 urandom warning(s) missed due to ratelimiting

!!!!! Console recovery shell
New value of PCR[4]: 8a6a96fde1a8dd96271479dc40742b36aba3c2b3
!!!!! Starting recovery shell
~ # Do your tests here under Heads recovery shell!!!!
```

What’s Next

- TPM2 support on QEMU/KVM and SWTPM (skeleton: https://github.com/osresearch/heads/pull/1292)
- A better build system to guarantee reproducible builds based on NixOS if everything goes well (**PoCs started**)
- Clean room, in ram GPG key generation with backup/restore/USB thumb drive emergency usage capabilities (No more USB Security dongle strong requirement to use Heads while still highly recommended).
  - Authenticated Heads recovery shell, USB boot and more! (under design!)
- Finally: flash write protection options!
  - Platform chipset locking (only Heads can flash firmware) (PoC: https://github.com/osresearch/heads/pull/326#issuecomment-1019684512)
  - SPI Write protection, permitting to write protect coreboot’s bootblock region (requires external flashing when coreboot version bumps happen under Heads. For the most paranoid only!)
    - 3mdeb dasharo/flashrom merge soon : https://github.com/osresearch/heads/pull/1251
- International keyboard support (PoC started: https://github.com/osresearch/heads/issues/555)
- On demand MAC randomization inside of Heads, overwriting GBE region inside of firmware. Persistence across firmware upgrades. (PoC started: https://github.com/osresearch/heads/pull/1195)
- Even more space for Maximized roms! (additional ~0.3-0.6mb): (PoC : https://github.com/osresearch/heads/pull/1298

References / Links

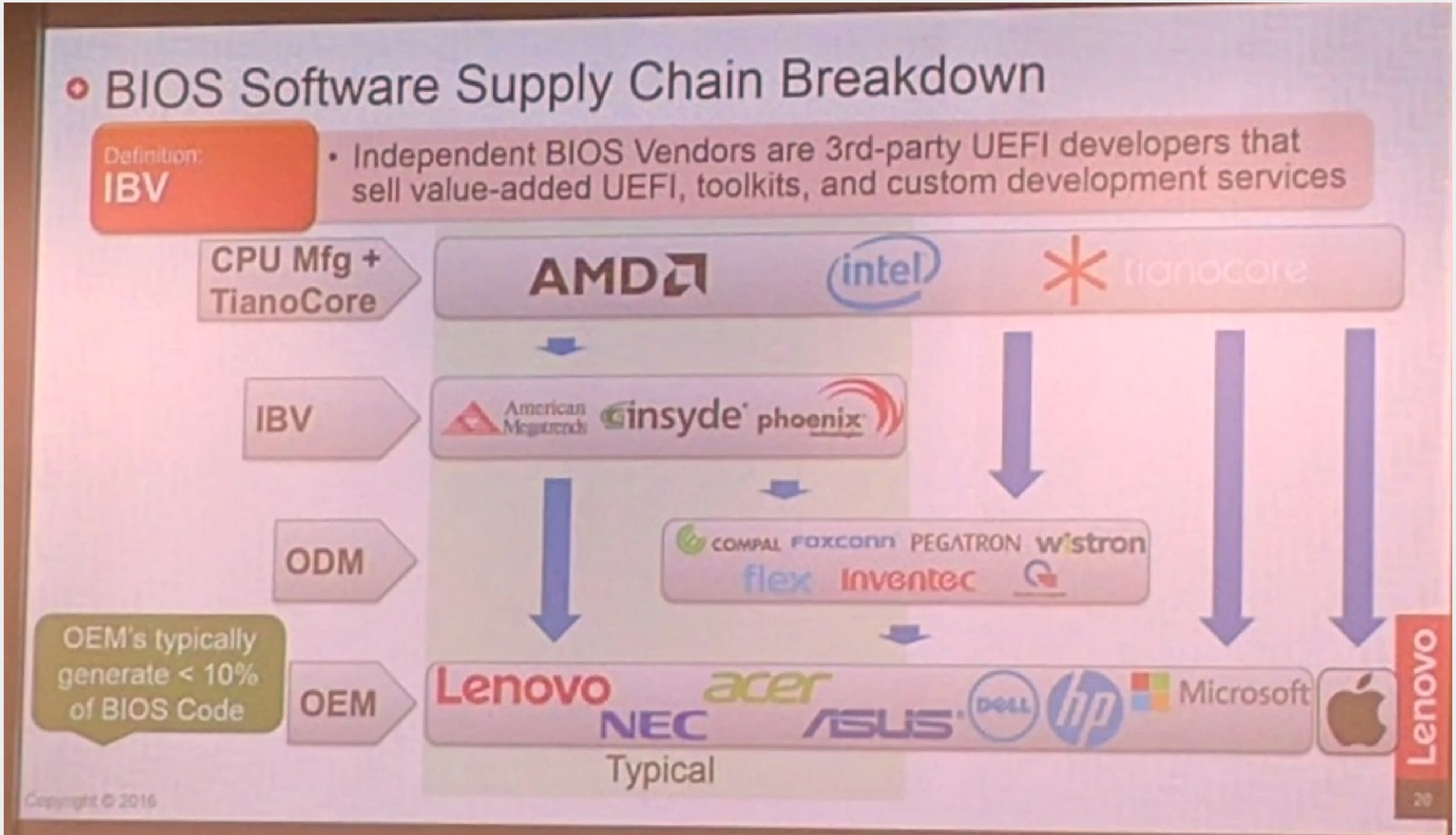
References:

- Differences between linuxboot, Heads NERF
- Heads conference (Hudson, 33c3, 2016)
- Linuxboot conference (Hudson, 34c3, 2017)
- Heads: a call for collaboration (Laurion, FOSDEM, 2020)
- Coreboot measured boot, SRTM mode (coreboot doc)
- Heads current measured boot scheme (Heads doc)

Project homes

- Heads searchable documentation
- Heads project's home (GitHub code/features/issues)
- Heads documentation's home (GitHub documentation/issues)
- Heads community direct link

Questions/Comments?


BIOS Software Supply Chain Breakdown

Definition: IBV

- Independent BIOS Vendors are 3rd-party UEFI developers that sell value-added UEFI, toolkits, and custom development services