LINUXCON

@

THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
EUROPE

Ostree For The Uninitiated

What You Need to Get Up And Running With Ostree On Your Next Project

Presented by: Davis Roman
Bilbao, Spain
9/21/2023

#OSSummit @davisroman84

THE LINUX FOUNDATION

Who am I?

- Embedded Linux Engineer
- 16 years of experience focusing on home security products and access control systems and now I’m in the automotive space.
- Previously employed at Honeywell, Resideo and now at Lucid Motors since February.
- Originally from Queens, New York but now I’m based out of Fremont, California.
- I’ve been using ostree for many years across multiple projects.
  - One of which is a product with a deployment of 1M+ devices.

Imagine this scenario

It’s your first day at Megacorp and your boss hands you a next generation device bound to change the world.

You’re told it runs embedded Linux but before you can begin developing your cool apps on this device you’ll need to first update it to the latest and greatest. It’s been on a shelf for a couple of months so it’s likely very out of date.

Out of habit you reach for your usb thumb drive but you notice you can ssh into the egg minder.(Because why wouldn’t the egg minder support ssh right?)

You ssh into the egg minder and as you begin to type a bunch of ‘*apt get install*” commands your boss says to you “*Don’t be silly, just type the following*”

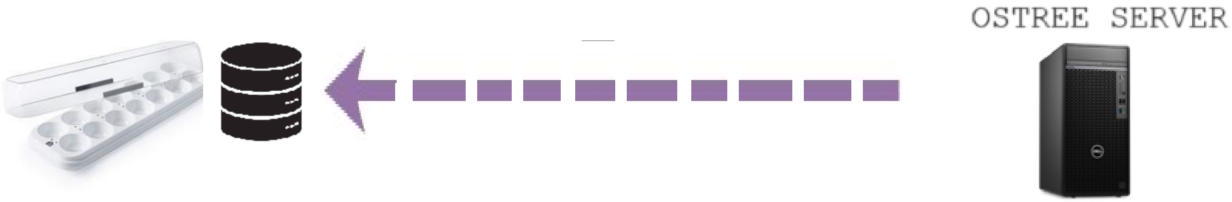
```
$ ostree admin upgrade -r
```

Done

A few moments later, the system reboots and the update is complete.

What just happened?

A bird’s eye view of ostree’s update process



1) Ostree determined the required remote and ref (think “branch”) using the origin file.
2) Summary file was pulled from the ostree server which specified the latest ref and commit.
3) A diff of the commit metadata was pulled and stored into its local ostree repository.
4) The commit was deployed
   a) Hardlink farm was created in the deploy directory (pointing back to the ostree repository)
   b) The uEnv.txt file was updated reflecting the new deployment as well as which kernel and initramfs to use.
   c) 3 way merge occured in /etc
   d) /var was selected according to stateroot
5) After the reboot, the u-boot bootloader mounted the filesystem and imported the uEnv.txt file into its environment.
6) The deployment path was passed via the kernel args and the kernel was booted with the initramfs.
7) The initramfs determined which deployment to use (via the kernel arguments) and mounted it onto the physical sysroot.
8) The init process chrooted into the deployment (becoming its new /) and proceeded with booting into userspace as ususal.
9) You got all that right?

A lot just happened so let’s take it one step at a time

Objectives

- ❑ Discuss prerequisites for understanding ostree
- ❑ Ostree vs libostree and why is it hard to search on youtube/google for ostree related content.
- ❑ package vs image vs differential updates
- ❑ Push/pull/offline/distributed architectures.
- ❑ Getting started with an ostree enabled raspberrypi4 build using the yocto project
- ❑ Pushing to an ostree server (even though it’s not natively supported).
- ❑ Command line basics
- ❑ Booting into an ostree deployment
- ❑ Yocto integration

Prerequistes

- Users (app developers / testers)
  - None.
  - Learn ostree command line basics and you’ll be fine.

- System integrators (bsp / system engineers)
  - Overlayfs - OverlayFS and its use in Yocto Project - Vyacheslav Yurkov, Precitec GmbH & Co. KG
  - Hardlinks - MicroNuggets: Hard Links versus Soft Links Explained
  - Chroots – Managing Chroot Jails in Linux – theurbanpenguin
  - Bind mounts - Symlinks or Bind Mounts – PCTLC
  - Initramfs
    - David Hand   "Linux initramfs for fun, and, uh...“
    - https://kernel.org/doc/Documentation/filesystems/ramfs-rootfs-initramfs.txt
  - u-boot - Tutorial: Introduction to the Embedded Boot Loader U-boot - Behan Webster, Converse in Code
  - Yocto
    - “Introduction to the Yocto Project and Bitbake, Part 1” by Behan Webster
    - “Introduction to the Yocto Project and Bitbake, Part 2” by Behan Webster

Ostree

- Created by Colin Walters from Red Hat
  - Originally for the Gnome continuous project
- Essentially git for filesystem trees
- Features
  - Transactional upgrades (atomic)
    - Applied fully or not at all
  - Rollback
    - Able to easily go back to the previous version
  - Allows for replicating content incrementally over HTTP.
  - Parallel installing more than 2 bootable deployments
  - Binary history on the client and server
  - Flexible support for multiple branches and repositories
  - Designed to be resistant to sudden power less
- Driven initially by server needs
- Primarily focuses on the delivery and deployment of filesystem trees

Ostree (cont)

- ❑ Originally called ostree but has since changed its name to libostree
  - ❑ To get away from putting too much emphasis on the command line tool itself.
- ❑ ‘Ostree’ refers to the command line tool while ‘libostree’ refers to the library
- ❑ Many blogs and tutorials often use the terms interchangeably (which can be confusing)
- ❑ Began in the desktop operating system space but has since been ported over to embedded Linux use cases. Most videos and online documentation are skewed towards the desktop, at least for now.

Uses of ostree in the wild

Automotive grade Linux

ChromeOS upgrader

Gnome Continuous

Flatpaks

TorizonCore

EndlessOS

- Fedora Project
  - Fedora Silverblue
  - Fedora CoreOS
  - rpm-ostree

Package vs Image vs differential updates

Package based (rpm, apt, yum, etc)

- Advantages:

- Requires low bandwidth
- Easy to use and lots of information available online

- Disadvantages:

- Every system is slightly custom since there is no strict control of what can be installed for a specific version.
- Unreliable in absence of a human operator
- Not powersafe and could corrupt the system if installation is interrupted.

Full filesystem update(Image based)

- Advantages:

- Can be tested exhaustively
- Easy to reason about
- Can be powersafe with an A/B configuration

- Disadvantages:

- Consumes a lot of network traffic
- Decreases lifetime of storage medium due to excessive writing

Atomic differential updates(ostree)

- Advantages

- Requires minimal network bandwith as only the difference is downloaded.
- Power safe due to transactional nature
- The smaller the difference, the faster the update takes

- Disadvantages:

- Requires a reboot (most folks complain about this)
- One http request for each file we don't already have
- Cannot perform cryptographic verification at the block level.

Youtube – “Colin Walters - OSTree: A middle ground between packages and images”

Objectives - recap

- ✓ ~~Discuss ostree prerequisites~~
- ✓ ~~Ostree vs libostree and why is it hard to search on youtube for ostree related content.~~
- ✓ ~~package vs image vs differential updates~~
- □ Push/pull/offline/distributed architectures.
- □ Getting started with an ostree enabled raspberrypi4 build using the yocto project.
- □ Pushing to an ostree server (even though its not natively supported).
- □ Command line basics
- □ Booting into an ostree commit
- □ Yocto integration

Ostree Architectures

- Pull
- Push
- Offline
- Distributed

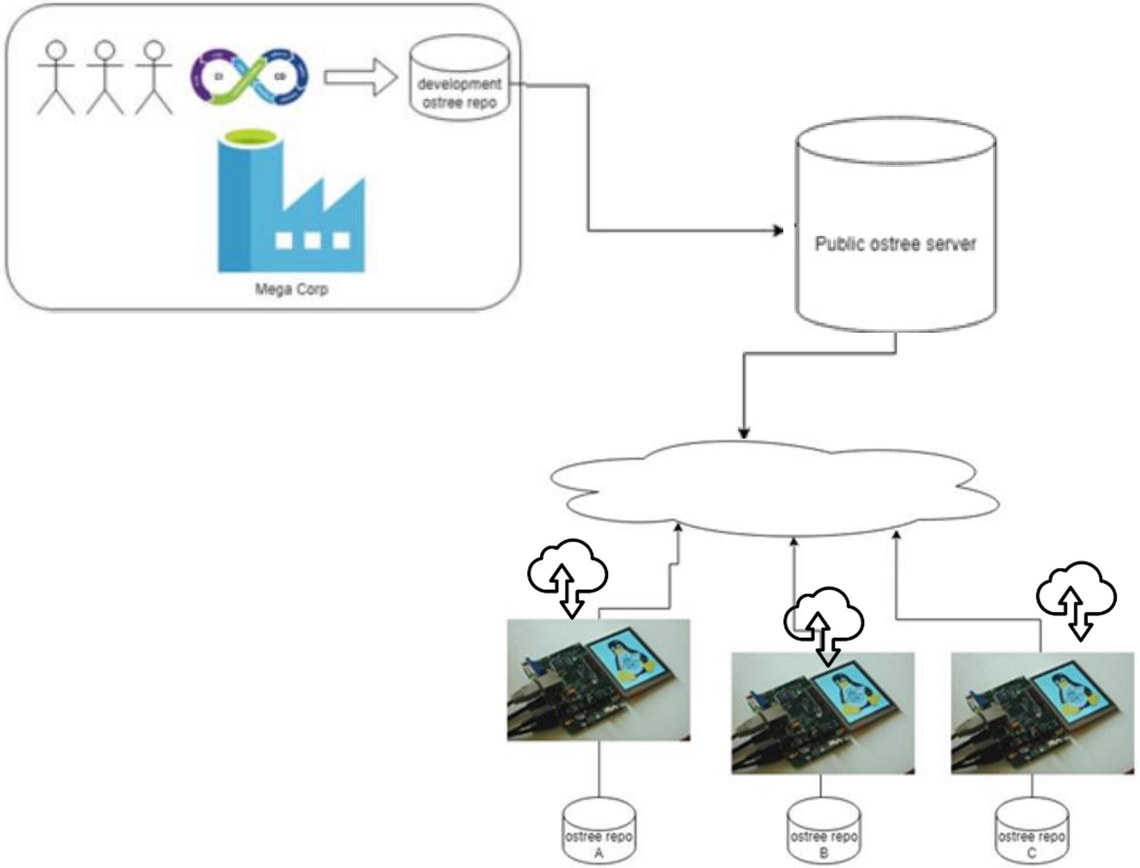
Pull

Advantages:

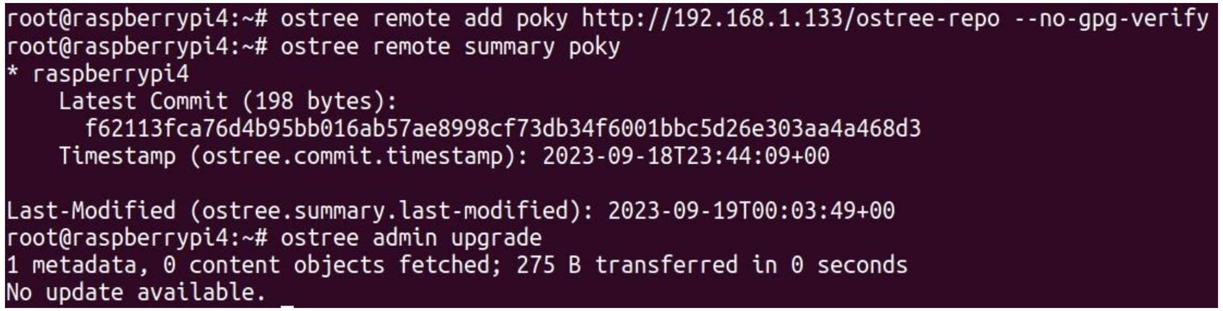
- Relies on the ostree server publishing a summary file that all clients can periodically monitor.
- If you simply care about always being on the latest, then this path is straight forward.

Disadvantages:

- Does not have strict control over which commit gets downloaded onto specific devices.



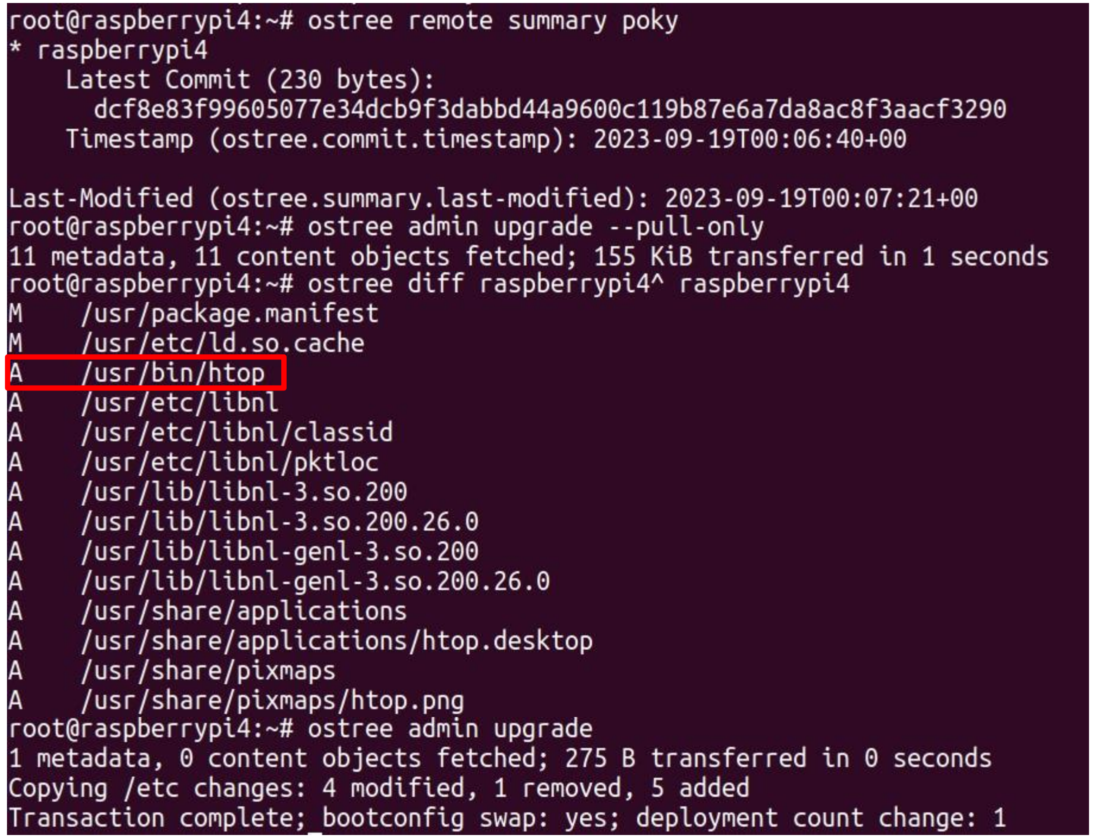
Pull - tutorial



```
root@raspberrypi4:~# ostree remote add poky http://192.168.1.133/ostree-repo --no-gpg-verify
root@raspberrypi4:~# ostree remote summary poky
* raspberrypi4
    Latest Commit (198 bytes):
      f62113fca76d4b95bb016ab57ae8998cf73db34f6001bbc5d26e303aa4a468d3
    Timestamp (ostree.commit.timestamp): 2023-09-18T23:44:09+00

Last-Modified (ostree.summary.last-modified): 2023-09-19T00:03:49+00
root@raspberrypi4:~# ostree admin upgrade
1 metadata, 0 content objects fetched; 275 B transferred in 0 seconds
No update available.
```

Now, in the background, we'll add the htop package to our Yocto build, and then push to the ostree server. (We haven't talked about how to push to an ostree server yet)

☑



```
root@raspberrypi4:~# ostree remote summary poky
* raspberrypi4
    Latest Commit (230 bytes):
      dcf8e83f99605077e34dcb9f3dabbd44a9600c119b87e6a7da8ac8f3aacf3290
    Timestamp (ostree.commit.timestamp): 2023-09-19T00:06:40+00

Last-Modified (ostree.summary.last-modified): 2023-09-19T00:07:21+00
root@raspberrypi4:~# ostree admin upgrade --pull-only
11 metadata, 11 content objects fetched; 155 KiB transferred in 1 seconds
root@raspberrypi4:~# ostree diff raspberrypi4^ raspberrypi4
M    /usr/package.manifest
M    /usr/etc/ld.so.cache
A    /usr/bin/htop
A    /usr/etc/libnl
A    /usr/etc/libnl/classid
A    /usr/etc/libnl/pktloc
A    /usr/lib/libnl-3.so.200
A    /usr/lib/libnl-3.so.200.26.0
A    /usr/lib/libnl-genl-3.so.200
A    /usr/lib/libnl-genl-3.so.200.26.0
A    /usr/share/applications
A    /usr/share/applications/htop.desktop
A    /usr/share/pixmaps
A    /usr/share/pixmaps/htop.png
root@raspberrypi4:~# ostree admin upgrade
1 metadata, 0 content objects fetched; 275 B transferred in 0 seconds
Copying /etc changes: 4 modified, 1 removed, 5 added
Transaction complete; bootconfig swap: yes; deployment count change: 1
```

Ostree Architectures

- ✓ ~~Pull~~
- ❑ Push
- ❑ Offline
- ❑ Distributed

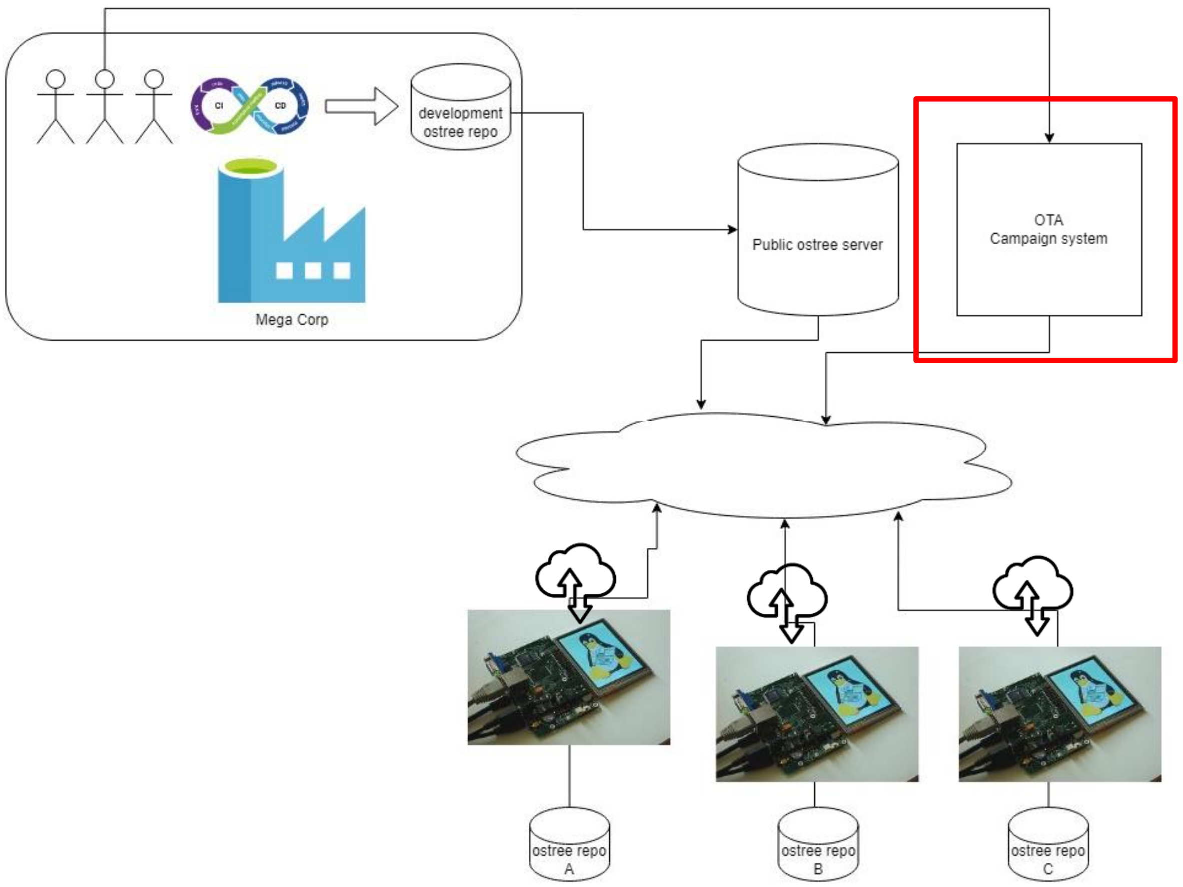
Push

Advantages:

- □ Fine grain control over which devices get specific updates.
- □ Region based upgrades
- □ Carefully control the rollout to limit exposure in case of a bad OTA.
- □ Closely monitor the progress of your update campaigns.

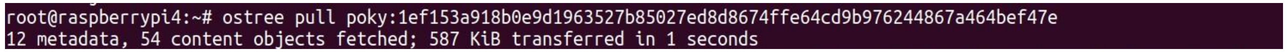
Disadvantages:

- □ More upfront cost is needed to develop and maintain the OTA campaign system although many off the shelf solutions exist such as TorizonCore.



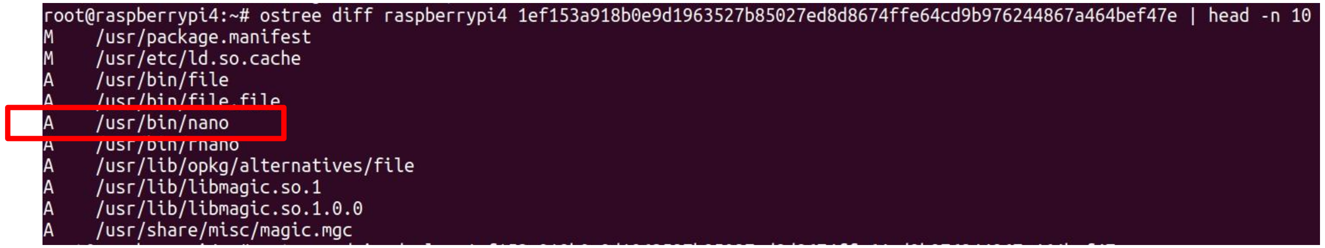
Push - tutorial

- ❑ PREPARATION WORK: In the background, we'll add the nano package to our Yocto build, and then push to the ostree server. (We still haven't talked about how to push to an ostree server yet)



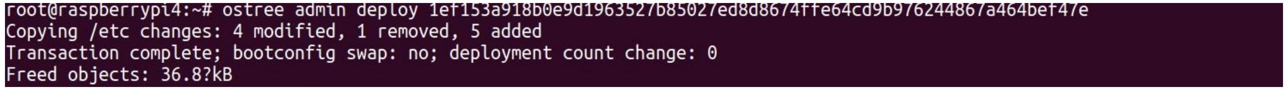
```
root@raspberrypi4:~# ostree pull poky:1ef153a918b0e9d1963527b85027ed8d8674ffe64cd9b976244867a464bef47e
12 metadata, 54 content objects fetched; 587 KiB transferred in 1 seconds
```

- ❑ Diffing the current and the new commit shows that we now have nano



```
root@raspberrypi4:~# ostree diff raspberrypi4 1ef153a918b0e9d1963527b85027ed8d8674ffe64cd9b976244867a464bef47e | head -n 10
M    /usr/package.manifest
M    /usr/etc/ld.so.cache
A    /usr/bin/file
A    /usr/bin/file.file
A    /usr/bin/nano
A    /usr/bin/rnano
A    /usr/lib/opkg/alternatives/file
A    /usr/lib/libmagic.so.1
A    /usr/lib/libmagic.so.1.0.0
A    /usr/share/misc/magic.mgc
```

- ❑ Deploy new commit



```
root@raspberrypi4:~# ostree admin deploy 1ef153a918b0e9d1963527b85027ed8d8674ffe64cd9b976244867a464bef47e
Copying /etc changes: 4 modified, 1 removed, 5 added
Transaction complete; bootconfig swap: no; deployment count change: 0
Freed objects: 36.8?kB
```

☑

Ostree Architectures

- ✓ ~~Pull~~
- ✓ ~~Push~~
- □ Offline
- □ Distributed

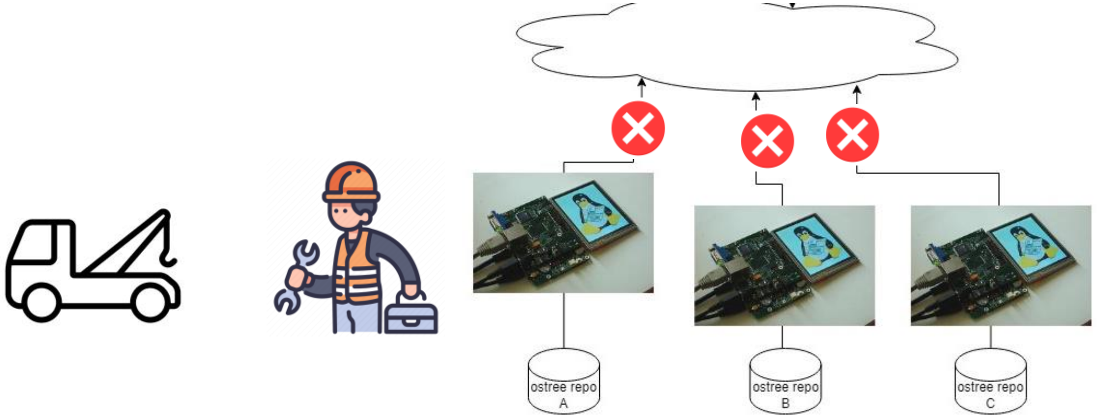
Offline

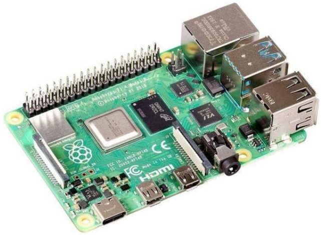
Advantages:

- Allows for airgrapped systems
- Improves ability to ensure that firmware is only used by approved individuals (provided that you trust them of course)

Disadvantages:

- Cannot push firmware over the internet
- Physical access is required but not always possible (ie: bouy in the middle of the ocean)
- Can be very expensive to visit customer sites



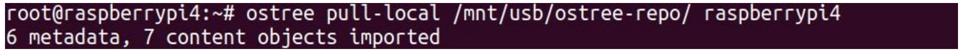


Offline - tutorial

PREPARATION WORK: We now add the jq package to our build, re-run yocto then we push the commit to the ostree server

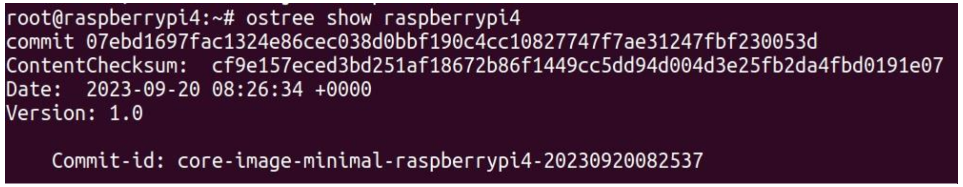
Using pull-local

❑ On the build machine, copy the ostree repo directory onto the usb drive, then insert the usb drive into the raspberry pi to pull the commit into the local repo.



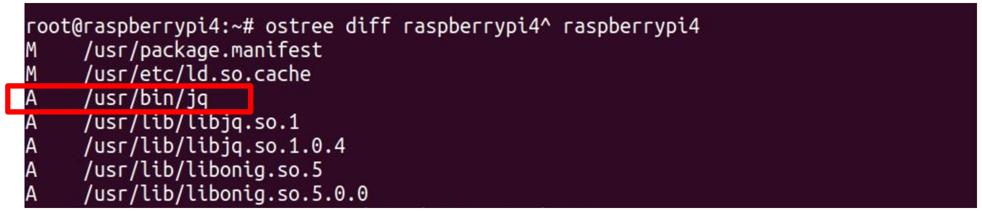
```
root@raspberrypi4:~# ostree pull-local /mnt/usb/ostree-repo/ raspberrypi4
6 metadata, 7 content objects imported
```

❑ Show new commit



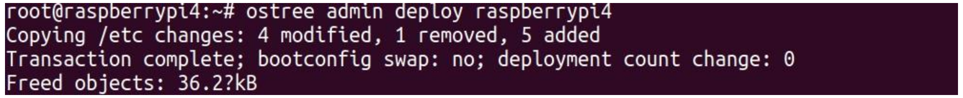
```
root@raspberrypi4:~# ostree show raspberrypi4
commit 07ebd1697fac1324e86cec038d0bbf190c4cc10827747f7ae31247fbf230053d
ContentChecksum:  cf9e157eced3bd251af18672b86f1449cc5dd94d004d3e25fb2da4fbd0191e07
Date:  2023-09-20 08:26:34 +0000
Version: 1.0

    Commit-id: core-image-minimal-raspberrypi4-20230920082537
```

❑ When comparing the previous and new commit, jq is added



```
root@raspberrypi4:~# ostree diff raspberrypi4^ raspberrypi4
M    /usr/package.manifest
M    /usr/etc/ld.so.cache
A    /usr/bin/jq
A    /usr/lib/libjq.so.1
A    /usr/lib/libjq.so.1.0.4
A    /usr/lib/libonig.so.5
A    /usr/lib/libonig.so.5.0.0
```

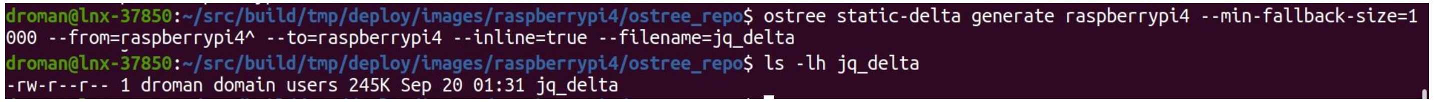
❑ Deploy new commit



```
root@raspberrypi4:~# ostree admin deploy raspberrypi4
Copying /etc changes: 4 modified, 1 removed, 5 added
Transaction complete; bootconfig swap: no; deployment count change: 0
Freed objects: 36.2?kB
```

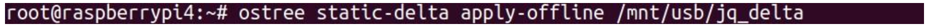
Using static deltas

❑ Generate static-delta file on build machine, then copy file onto raspberrypi and apply.



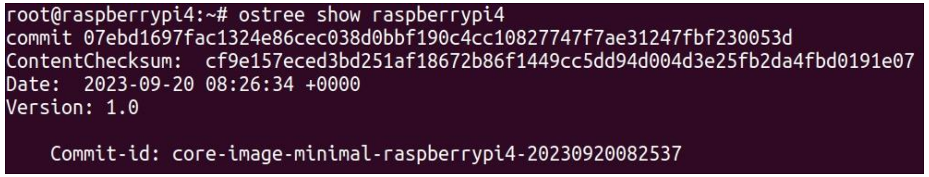
```
droman@lnx-37850:~/src/build/tmp/deploy/images/raspberrypi4/ostree_repo$ ostree static-delta generate raspberrypi4 --min-fallback-size=1
000 --from=raspberrypi4^ --to=raspberrypi4 --inline=true --filename=jq_delta
droman@lnx-37850:~/src/build/tmp/deploy/images/raspberrypi4/ostree_repo$ ls -lh jq_delta
-rw-r--r-- 1 droman domain users 245K Sep 20 01:31 jq_delta
```

❑ Copy delta file onto the raspberrypi and apply it.



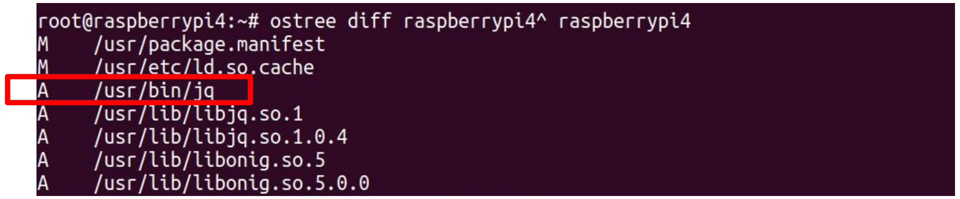
```
root@raspberrypi4:~# ostree static-delta apply-offline /mnt/usb/jq_delta
```

❑ Show new commit



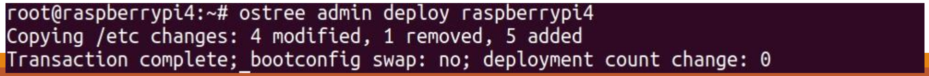
```
root@raspberrypi4:~# ostree show raspberrypi4
commit 07ebd1697fac1324e86cec038d0bbf190c4cc10827747f7ae31247fbf230053d
ContentChecksum:  cf9e157eced3bd251af18672b86f1449cc5dd94d004d3e25fb2da4fbd0191e07
Date:  2023-09-20 08:26:34 +0000
Version: 1.0

    Commit-id: core-image-minimal-raspberrypi4-20230920082537
```

❑ When comparing the previous and new commit, jq is added



```
root@raspberrypi4:~# ostree diff raspberrypi4^ raspberrypi4
M    /usr/package.manifest
M    /usr/etc/ld.so.cache
A    /usr/bin/jq
A    /usr/lib/libjq.so.1
A    /usr/lib/libjq.so.1.0.4
A    /usr/lib/libonig.so.5
A    /usr/lib/libonig.so.5.0.0
```

❑ Deploy new commit



```
root@raspberrypi4:~# ostree admin deploy raspberrypi4
Copying /etc changes: 4 modified, 1 removed, 5 added
Transaction complete; bootconfig swap: no; deployment count change: 0
```

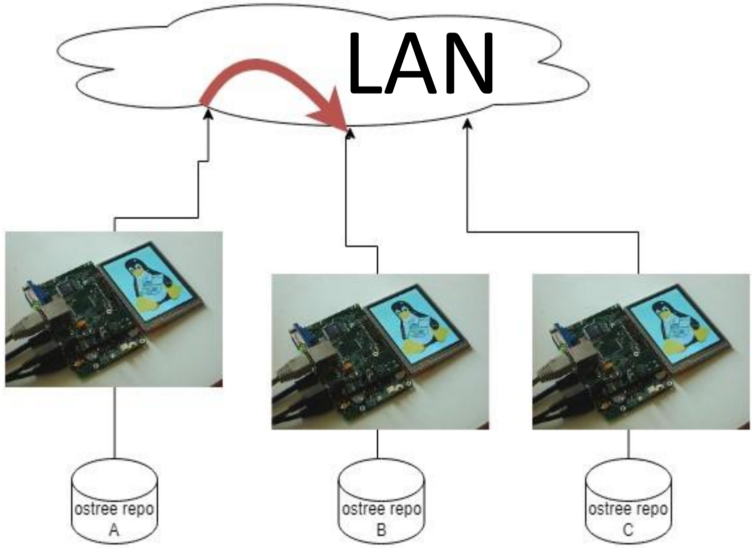
Ostree Architectures

- ✓ ~~Pull~~
- ✓ ~~Push~~
- ✓ ~~Offline~~
- ❑ Distributed

Distributed

- ❑ Peer to peer
- ❑ Uses mDNS and DNS-SD
- ❑ Collection-ids
  - ❑ Universally unique identifiers relying on reverse domain name handles
- ❑ Example use case:
  - ❑ A classroom of devices where all devices get updated as soon as a newer device appears on the network.
- ❑ Most interesting distribution scheme however I don't fully have this working yet, so I don't have a tutorial to share yet. Sorry!

- ❑ [“Peer to peer OS and flatpak updates” - All Systems Go! Conference]
- ❑ [GUADEC 2018 - Matthew Leeds - P2P Distribution of Flatpaks and OSTrees]



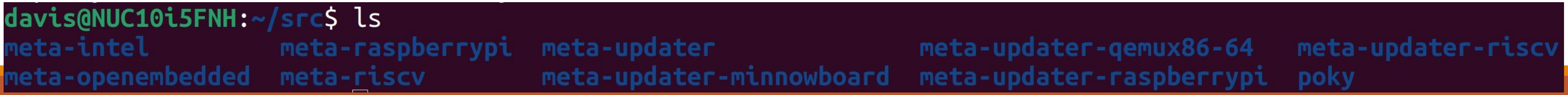
Objectives - recap

- ✓ ~~Discuss ostree prerequisites~~
- ✓ ~~Ostree vs libostree and why is it hard to search on youtube for ostree related content.~~
- ✓ ~~package vs image vs differential updates~~
- ✓ ~~Push/pull/offline/distributed architectures.~~
- □ Getting started with an ostree enabled raspberrypi4 build using the yocto project
- □ Pushing to an ostree server (even though its not natively supported).
- □ Command line basics
- □ Booting into an ostree commit
- □ Yocto integration

Create raspberry pi 4 image using meta-updater

```
Setup repo tool:
$ mkdir -p ~/.bin
$ echo export PATH=\$HOME/.bin:\$PATH >> ~/.bashrc && source ~/.bashrc
$ curl https://storage.googleapis.com/git-repo-downloads/repo > ~/.bin/repo
$ chmod +x ~/.bin/repo
```

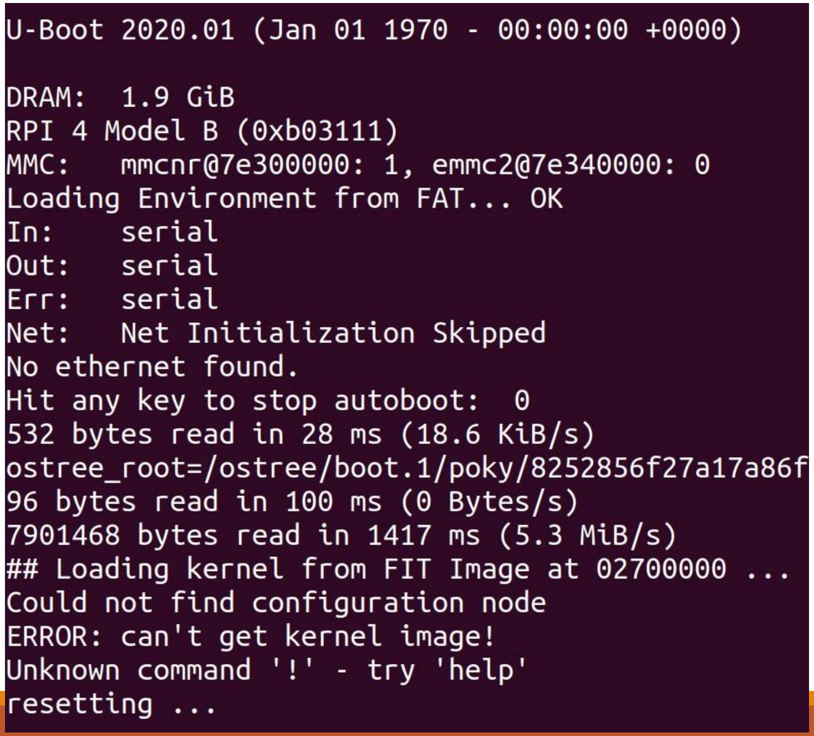
```
Setup yocto for raspberrypi4 build:
$ sudo apt-get install -y curl git chrpath diffstat gawk python3-distutils binutils g++ gcc make rpcsvc-proto
$ git config --global user.name “<Your Name>”
$ git config --global user.email “<Your Email>”
$ sudo ln -sf /usr/bin/python3 /usr/bin/python
$ mkdir src && cd src
$ repo init -u https://github.com/advancedtelematic/updater-repo.git -m dunfell.xml
$ repo sync
```



```
davis@NUC10i5FNH:~/src$ ls
meta-intel          meta-raspberrypi   meta-updater               meta-updater-qemux86-64    meta-updater-riscv
meta-openembedded   meta-riscv         meta-updater-minnowboard   meta-updater-raspberrypi   poky
```

Create raspberry pi 4 image using meta-updater (2)

Apply patch to address boot up issue with fit image configuration!



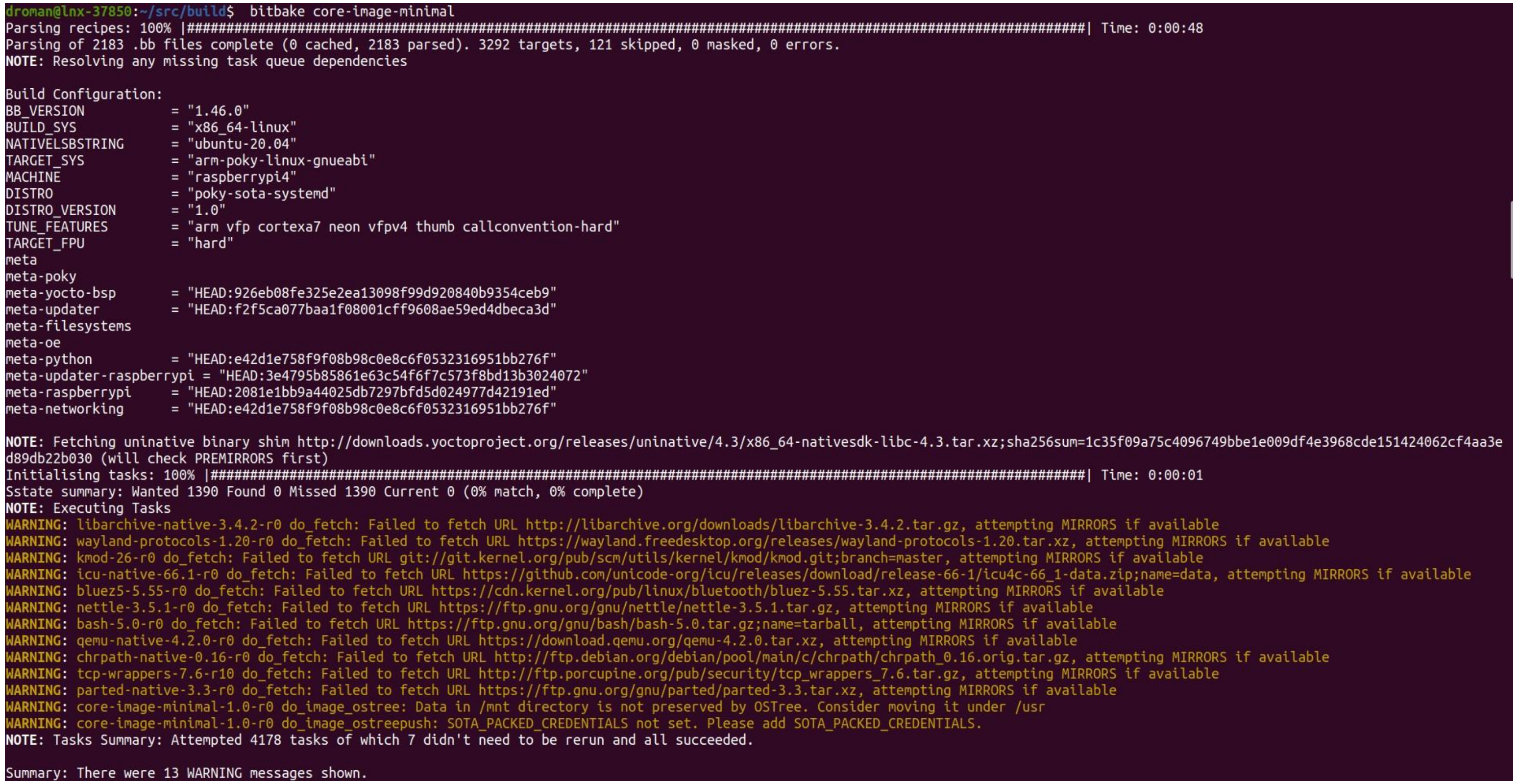
```
U-Boot 2020.01 (Jan 01 1970 - 00:00:00 +0000)

DRAM:  1.9 GiB
RPI 4 Model B (0xb03111)
MMC:   mmcnr@7e300000: 1, emmc2@7e340000: 0
Loading Environment from FAT... OK
In:    serial
Out:   serial
Err:   serial
Net:   Net Initialization Skipped
No ethernet found.
Hit any key to stop autoboot:  0
532 bytes read in 28 ms (18.6 KiB/s)
ostree_root=/ostree/boot.1/poky/8252856f27a17a86f
96 bytes read in 100 ms (0 Bytes/s)
7901468 bytes read in 1417 ms (5.3 MiB/s)
## Loading kernel from FIT Image at 02700000 ...
Could not find configuration node
ERROR: can't get kernel image!
Unknown command '!' - try 'help'
resetting ...
```

```
diff -Nupr a/meta-updater/recipes-sota/fit-conf/fit-conf.bb b/meta-updater/recipes-sota/fit-conf/fit-conf.bb
--- a/meta-updater/recipes-sota/fit-conf/fit-conf.bb        2023-09-10 10:11:12.113215781 -0700
+++ b/meta-updater/recipes-sota/fit-conf/fit-conf.bb        2023-09-10 10:10:26.248361009 -0700
@@ -9,11 +9,11 @@ do_install() {
        echo -n "fit_conf=" >${D}${libdir}/fit_conf

        if [ -n ${SOTA_MAIN_DTB} ]; then
-               echo -n "#conf@${SOTA_MAIN_DTB}" >> ${D}${libdir}/fit_conf
+               echo -n "#conf-${SOTA_MAIN_DTB}" >> ${D}${libdir}/fit_conf
        fi

        for ovrl in ${SOTA_DT_OVERLAYS}; do
-               echo -n "#conf@overlays_${ovrl}" >> ${D}${libdir}/fit_conf
+               echo -n "#conf-overlays_${ovrl}" >> ${D}${libdir}/fit_conf
        done

        for conf_frag in ${SOTA_EXTRA_CONF_FRAGS}; do
```

```
$ curl -L -s bit.ly/3rdtWPJ -o raspberrypi-bootup-fix.patch
$ patch -p1 < raspberrypi-bootup-fix.patch
```

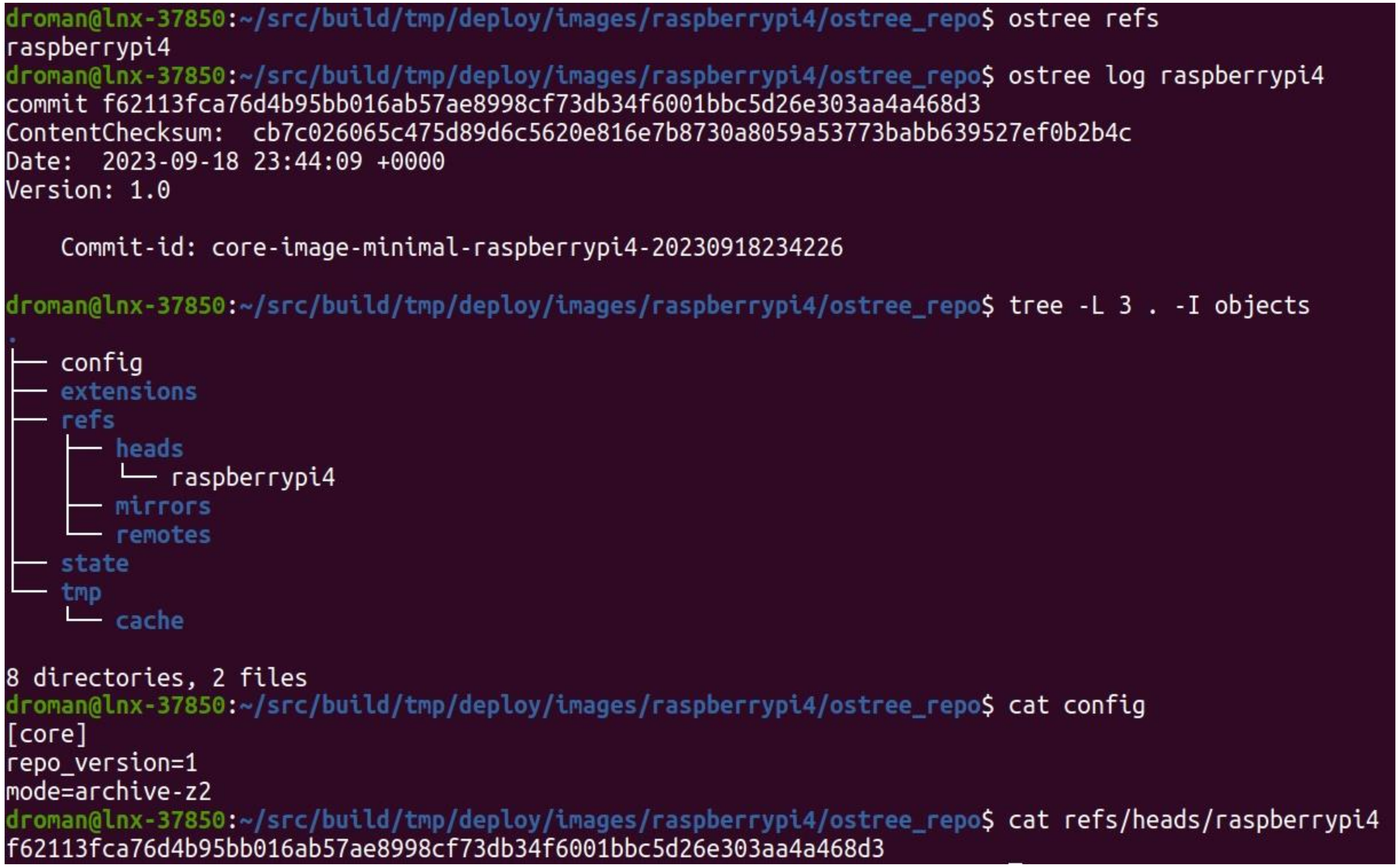
Create raspberry pi 4 image using meta-updater (3)

```
$ source meta-updater/scripts/envsetup.sh raspberrypi4
$ echo ‘OSTREE_UPDATE_SUMMARY=“1”’ >> conf/local.conf
$ bitbake core-image-minimal
```



```
droman@lnx-37850:~/src/build$  bitbake core-image-minimal
Parsing recipes: 100% |##################################################################################################################| Time: 0:00:48
Parsing of 2183 .bb files complete (0 cached, 2183 parsed). 3292 targets, 121 skipped, 0 masked, 0 errors.
NOTE: Resolving any missing task queue dependencies

Build Configuration:
BB_VERSION           = "1.46.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING      = "ubuntu-20.04"
TARGET_SYS           = "arm-poky-linux-gnueabi"
MACHINE              = "raspberrypi4"
DISTRO               = "poky-sota-systemd"
DISTRO_VERSION       = "1.0"
TUNE_FEATURES        = "arm vfp cortexa7 neon vfpv4 thumb callconvention-hard"
TARGET_FPU           = "hard"
meta
meta-poky
meta-yocto-bsp       = "HEAD:926eb08fe325e2ea13098f99d920840b9354ceb9"
meta-updater         = "HEAD:f2f5ca077baa1f08001cff9608ae59ed4dbeca3d"
meta-filesystems
meta-oe
meta-python          = "HEAD:e42d1e758f9f08b98c0e8c6f0532316951bb276f"
meta-updater-raspberrypi = "HEAD:3e4795b85861e63c54f6f7c573f8bd13b3024072"
meta-raspberrypi     = "HEAD:2081e1bb9a44025db7297bfd5d024977d42191ed"
meta-networking      = "HEAD:e42d1e758f9f08b98c0e8c6f0532316951bb276f"

NOTE: Fetching uninative binary shim http://downloads.yoctoproject.org/releases/uninative/4.3/x86_64-nativesdk-libc-4.3.tar.xz;sha256sum=1c35f09a75c4096749bbe1e009df4e3968cde151424062cf4aa3e
d89db22b030 (will check PREMIRRORS first)
Initialising tasks: 100% |###############################################################################################################| Time: 0:00:01
Sstate summary: Wanted 1390 Found 0 Missed 1390 Current 0 (0% match, 0% complete)
NOTE: Executing Tasks
WARNING: libarchive-native-3.4.2-r0 do_fetch: Failed to fetch URL http://libarchive.org/downloads/libarchive-3.4.2.tar.gz, attempting MIRRORS if available
WARNING: wayland-protocols-1.20-r0 do_fetch: Failed to fetch URL https://wayland.freedesktop.org/releases/wayland-protocols-1.20.tar.xz, attempting MIRRORS if available
WARNING: kmod-26-r0 do_fetch: Failed to fetch URL git://git.kernel.org/pub/scm/utils/kernel/kmod/kmod.git;branch=master, attempting MIRRORS if available
WARNING: icu-native-66.1-r0 do_fetch: Failed to fetch URL https://github.com/unicode-org/icu/releases/download/release-66-1/icu4c-66_1-data.zip;name=data, attempting MIRRORS if available
WARNING: bluez5-5.55-r0 do_fetch: Failed to fetch URL https://cdn.kernel.org/pub/linux/bluetooth/bluez-5.55.tar.xz, attempting MIRRORS if available
WARNING: nettle-3.5.1-r0 do_fetch: Failed to fetch URL https://ftp.gnu.org/gnu/nettle/nettle-3.5.1.tar.gz, attempting MIRRORS if available
WARNING: bash-5.0-r0 do_fetch: Failed to fetch URL https://ftp.gnu.org/gnu/bash/bash-5.0.tar.gz;name=tarball, attempting MIRRORS if available
WARNING: qemu-native-4.2.0-r0 do_fetch: Failed to fetch URL https://download.qemu.org/qemu-4.2.0.tar.xz, attempting MIRRORS if available
WARNING: chrpath-native-0.16-r0 do_fetch: Failed to fetch URL http://ftp.debian.org/debian/pool/main/c/chrpath/chrpath_0.16.orig.tar.gz, attempting MIRRORS if available
WARNING: tcp-wrappers-7.6-r10 do_fetch: Failed to fetch URL http://ftp.porcupine.org/pub/security/tcp_wrappers_7.6.tar.gz, attempting MIRRORS if available
WARNING: parted-native-3.3-r0 do_fetch: Failed to fetch URL https://ftp.gnu.org/gnu/parted/parted-3.3.tar.xz, attempting MIRRORS if available
WARNING: core-image-minimal-1.0-r0 do_image_ostree: Data in /mnt directory is not preserved by OSTree. Consider moving it under /usr
WARNING: core-image-minimal-1.0-r0 do_image_ostreepush: SOTA_PACKED_CREDENTIALS not set. Please add SOTA_PACKED_CREDENTIALS.
NOTE: Tasks Summary: Attempted 4178 tasks of which 7 didn't need to be rerun and all succeeded.

Summary: There were 13 WARNING messages shown.
```

Yocto’s newly generated ostree repo

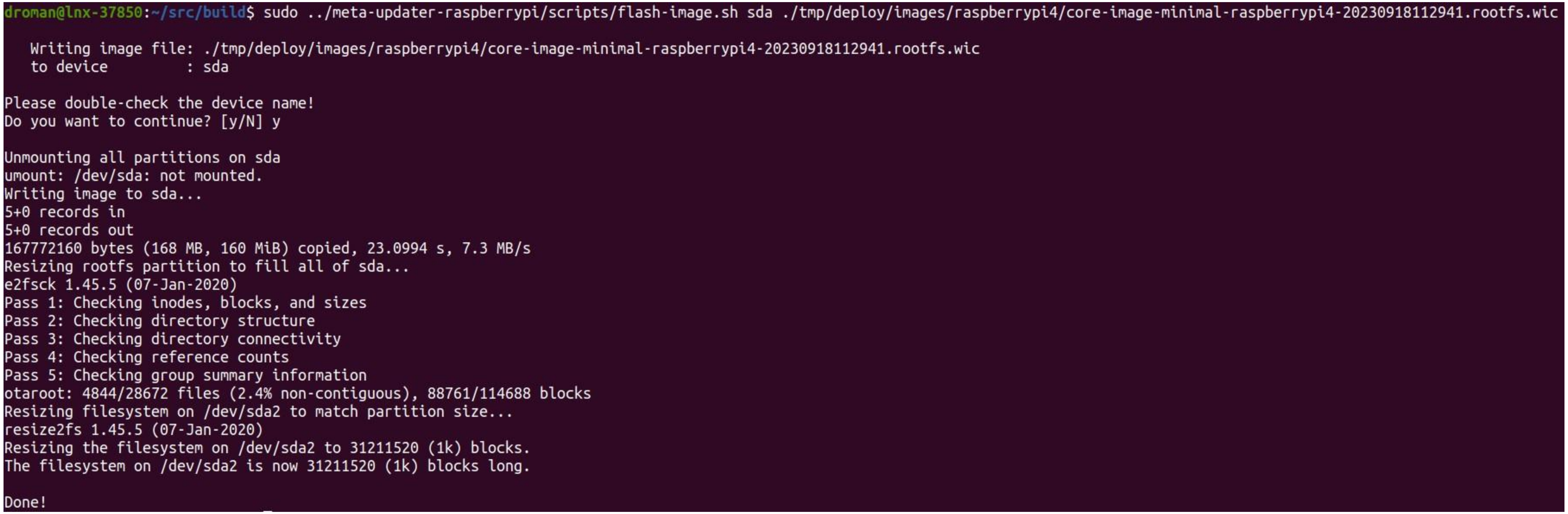


```
droman@lnx-37850:~/src/build/tmp/deploy/images/raspberrypi4/ostree_repo$ ostree refs
raspberrypi4
droman@lnx-37850:~/src/build/tmp/deploy/images/raspberrypi4/ostree_repo$ ostree log raspberrypi4
commit f62113fca76d4b95bb016ab57ae8998cf73db34f6001bbc5d26e303aa4a468d3
ContentChecksum:  cb7c026065c475d89d6c5620e816e7b8730a8059a53773babb639527ef0b2b4c
Date:  2023-09-18 23:44:09 +0000
Version: 1.0

    Commit-id: core-image-minimal-raspberrypi4-20230918234226

droman@lnx-37850:~/src/build/tmp/deploy/images/raspberrypi4/ostree_repo$ tree -L 3 . -I objects
.
├── config
├── extensions
├── refs
│   ├── heads
│   │   └── raspberrypi4
│   ├── mirrors
│   └── remotes
├── state
└── tmp
    └── cache

8 directories, 2 files
droman@lnx-37850:~/src/build/tmp/deploy/images/raspberrypi4/ostree_repo$ cat config
[core]
repo_version=1
mode=archive-z2
droman@lnx-37850:~/src/build/tmp/deploy/images/raspberrypi4/ostree_repo$ cat refs/heads/raspberrypi4
f62113fca76d4b95bb016ab57ae8998cf73db34f6001bbc5d26e303aa4a468d3
```

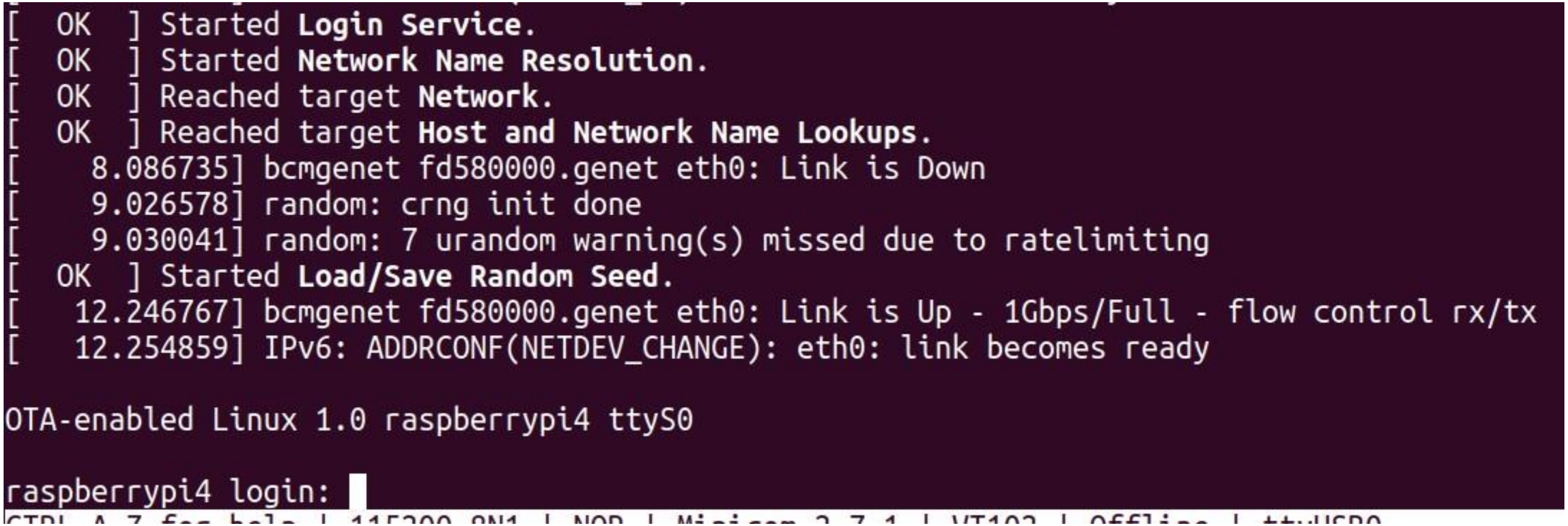
Create raspberry pi 4 sd card

Flash sd card using the following:

```
$ sudo ../meta-updater-raspberrypi/scripts/flash-image.sh <device> <path to wic file>
```



```
droman@lnx-37850:~/src/build$ sudo ../meta-updater-raspberrypi/scripts/flash-image.sh sda ./tmp/deploy/images/raspberrypi4/core-image-minimal-raspberrypi4-20230918112941.rootfs.wic

   Writing image file: ./tmp/deploy/images/raspberrypi4/core-image-minimal-raspberrypi4-20230918112941.rootfs.wic
   to device         : sda

Please double-check the device name!
Do you want to continue? [y/N] y

Unmounting all partitions on sda
umount: /dev/sda: not mounted.
Writing image to sda...
5+0 records in
5+0 records out
167772160 bytes (168 MB, 160 MiB) copied, 23.0994 s, 7.3 MB/s
Resizing rootfs partition to fill all of sda...
e2fsck 1.45.5 (07-Jan-2020)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
otaroot: 4844/28672 files (2.4% non-contiguous), 88761/114688 blocks
Resizing filesystem on /dev/sda2 to match partition size...
resize2fs 1.45.5 (07-Jan-2020)
Resizing the filesystem on /dev/sda2 to 31211520 (1k) blocks.
The filesystem on /dev/sda2 is now 31211520 (1k) blocks long.

Done!
```



```
[  OK  ] Started Login Service.
[  OK  ] Started Network Name Resolution.
[  OK  ] Reached target Network.
[  OK  ] Reached target Host and Network Name Lookups.
[    8.086735] bcmgenet fd580000.genet eth0: Link is Down
[    9.026578] random: crng init done
[    9.030041] random: 7 urandom warning(s) missed due to ratelimiting
[  OK  ] Started Load/Save Random Seed.
[   12.246767] bcmgenet fd580000.genet eth0: Link is Up - 1Gbps/Full - flow control rx/tx
[   12.254859] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready

OTA-enabled Linux 1.0 raspberrypi4 ttyS0

raspberrypi4 login: 
CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.7.1 | VT102 | Offline | ttyUSB0
```

Login: root (no password)

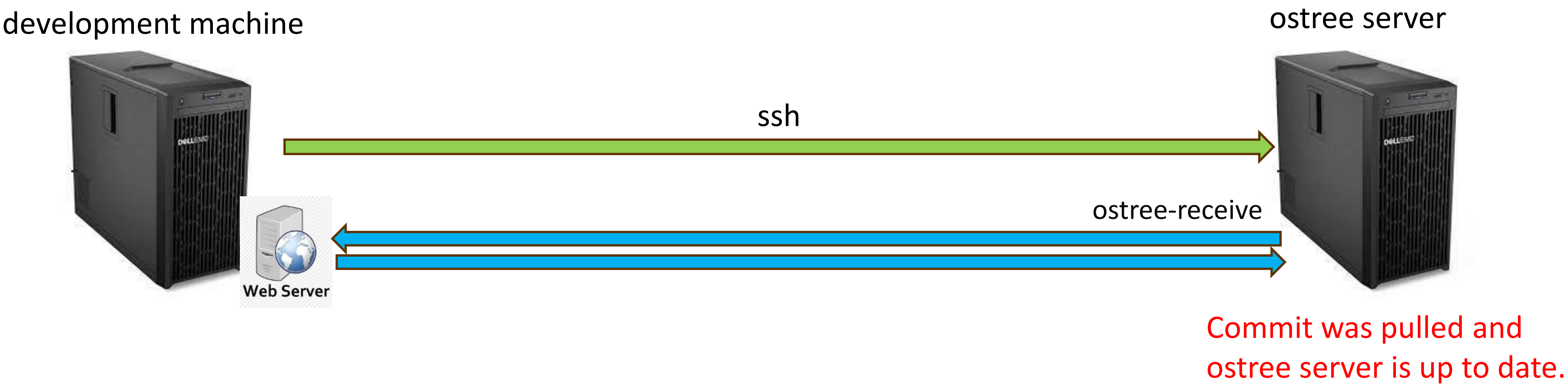
Objectives - recap

- ✓ ~~Discuss ostree prerequisites~~
- ✓ ~~Ostree vs libostree and why is it hard to search on youtube for ostree related content.~~
- ✓ ~~package vs image vs differential updates~~
- ✓ ~~Push/pull/offline/distributed architectures.~~
- ✓ ~~Getting started with an ostree enabled raspberrypi4 build using the yocto project~~
- □ Pushing to an ostree server (even though its not natively supported).
- □ Command line basics
- □ Booting into an ostree commit
- □ Yocto integration

Pushing to a remote ostree server

- During initial experimentation with ostree, I recommend that you use the same machine to host the build software as well as the ostree repository.
- Eventually you’ll want to adopt a more formal process - dev, staging and prod
- Ostree does not natively support pushing to a remote server however the ostree community has written a tool to address this gap.
- https://github.com/dbnicholson/ostree-push

Ostree-push



Commit was pulled and ostree server is up to date.

- When ostree-push is started, it starts a local HTTP server providing the contents of the local ostree repo on the development machine
- The development machine then connects to the ostree server via SSH and tunnels the HTTP server port through the SSH connection.
- The ostree-receive command is executed on the ostree server with the URL of the tunneled HTTP server.
- ostree-receive then creates a temporary remote using this URL and pulls the desired refs from the development machine.

Server configuration

1. Install ostree-push

```
$ sudo apt-get install ostree openssh-server gir1.2-ostree-1.0 git python3-pip
$ git clone https://github.com/dbnicholson/ostree-push ~/ostree-push
$ cd ~/ostree-push
$ pip install .
$ sudo ln -sf ~/.local/bin/ostree-receive /usr/local/bin/ostree-receive
```

2. Initialize ostree repo

```
$ sudo install -g $USER -o $USER -d /ostree
$ ostree --repo=/ostree/repo init --mode=archive-z2
```

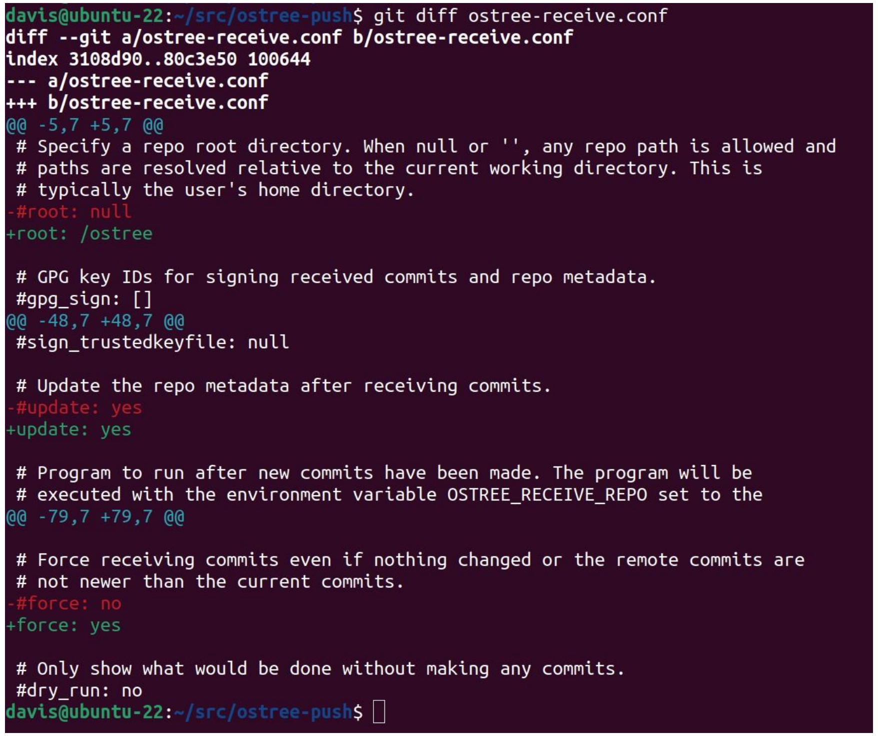
3. Install apache2 to host commits over http

```
$ sudo apt-get install apache2
$ sudo ln -sf /ostree/repo /var/www/html/ostree-repo
```

4. Apply patch and copy ostree-receive.conf file to ~/.config/ostree

```
$ mkdir -p ~/.config/ostree/
$ cp ~/ostree-push/ostree-receive.conf ~/.config/ostree/
```

Note: Installed on ubuntu 22.04.3



```
davis@ubuntu-22:~/src/ostree-push$ git diff ostree-receive.conf
diff --git a/ostree-receive.conf b/ostree-receive.conf
index 3108d90..80c3e50 100644
--- a/ostree-receive.conf
+++ b/ostree-receive.conf
@@ -5,7 +5,7 @@
 # Specify a repo root directory. When null or '', any repo path is allowed and
 # paths are resolved relative to the current working directory. This is
 # typically the user's home directory.
-#root: null
+root: /ostree

 # GPG key IDs for signing received commits and repo metadata.
 #gpg_sign: []
@@ -48,7 +48,7 @@
 #sign_trustedkeyfile: null

 # Update the repo metadata after receiving commits.
-#update: yes
+update: yes

 # Program to run after new commits have been made. The program will be
 # executed with the environment variable OSTREE_RECEIVE_REPO set to the
@@ -79,7 +79,7 @@

 # Force receiving commits even if nothing changed or the remote commits are
 # not newer than the current commits.
-#force: no
+force: yes

 # Only show what would be done without making any commits.
 #dry_run: no
davis@ubuntu-22:~/src/ostree-push$
```

Development machine configuration

1. **Setup ostree-push**

```
$ sudo apt-get install ostree gir1.2-ostree-1.0 git python3-pip
$ git clone https://github.com/dbnicholson/ostree-push ~/ostree-push
$ cd ~/ostree-push
$ pip install .
$ echo "export PATH=~/.local/bin:\$PATH" >> ~/.bashrc
$ . ~/.bashrc
```

2. **Create script, ~/ostree-push.sh, with the following content:**

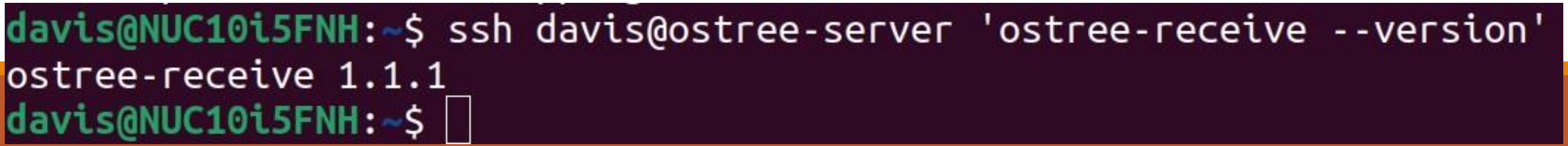
```
#!/usr/bin/env bash

repo=~/src/build/tmp/deploy/images/raspberrypi4-64/ostree_repo
ref=raspberrypi4
remote_host=<OSTREE_SERVER_IP_ADDRESS>
remote_user=<OSTREE_SERVER_USER>
destination_repo=repo

ostree-push --repo=$repo $remote_user@$remote_host:$destination_repo $ref
```

```
$ chmod +x ~/ostree-push.sh
```

3. **Setup passwordless ssh login (optional)**

```
$ ssh-keygen -t rsa
$ ssh-copyid <OSTREE_SERVER_USER>@<OSTREE_SERVER_IP_ADDRESS>
```

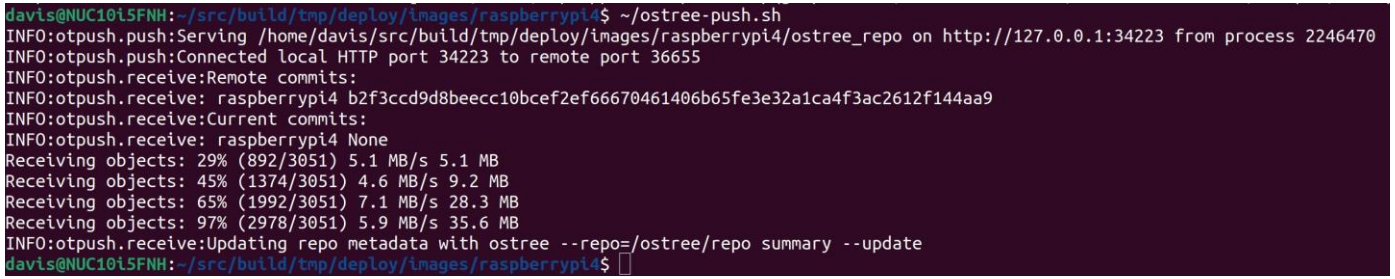
4. **Ensure you can execute 'ostree-receive --version' remotely before proceeding.** ***(Substitute 'ostree-server' with your IP)***



```
davis@NUC10i5FNH:~$ ssh davis@ostree-server 'ostree-receive --version'
ostree-receive 1.1.1
davis@NUC10i5FNH:~$
```

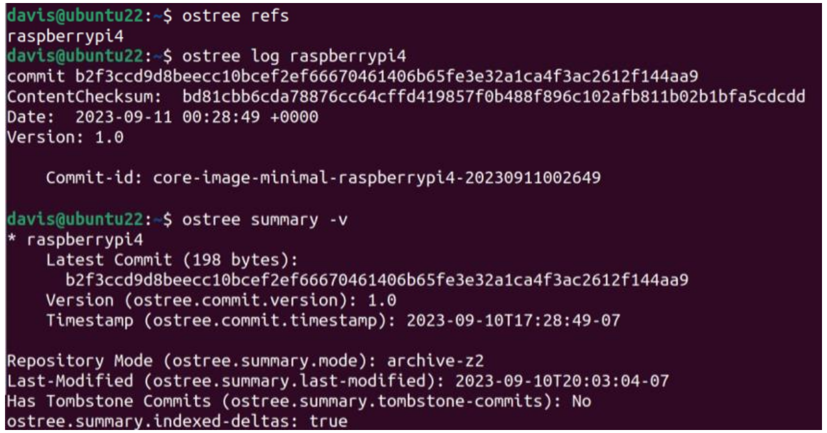
Push commit to ostree server

- Run ostree-push.sh on development machine

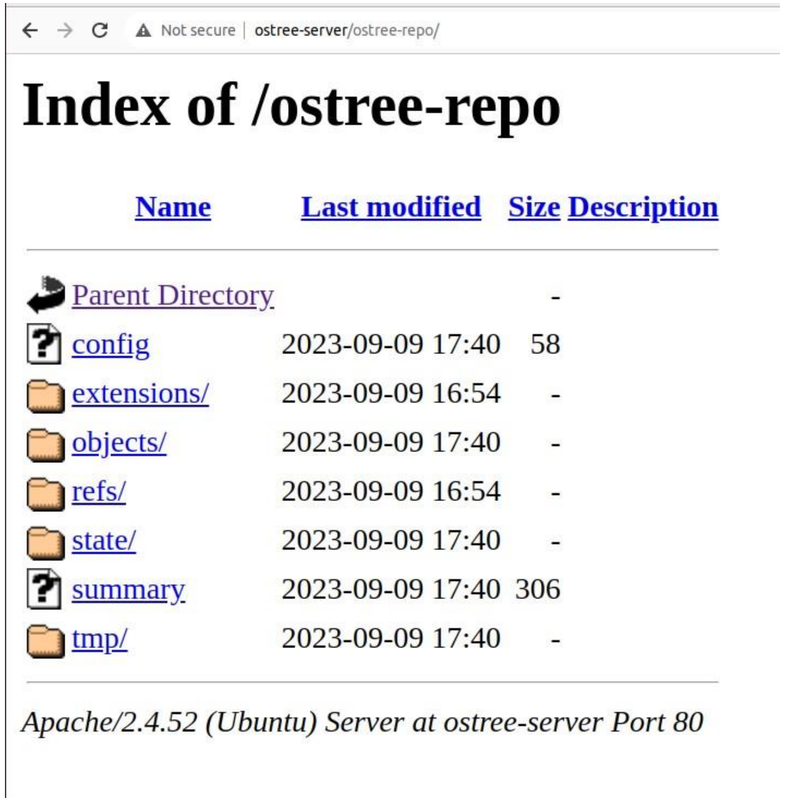


```
davis@NUC10i5FNH:~/src/build/tmp/deploy/images/raspberrypi4$ ~/ostree-push.sh
INFO:otpush.push:Serving /home/davis/src/build/tmp/deploy/images/raspberrypi4/ostree_repo on http://127.0.0.1:34223 from process 2246470
INFO:otpush.push:Connected local HTTP port 34223 to remote port 36655
INFO:otpush.receive:Remote commits:
INFO:otpush.receive: raspberrypi4 b2f3ccd9d8beecc10bcef2ef66670461406b65fe3e32a1ca4f3ac2612f144aa9
INFO:otpush.receive:Current commits:
INFO:otpush.receive: raspberrypi4 None
Receiving objects: 29% (892/3051) 5.1 MB/s 5.1 MB
Receiving objects: 45% (1374/3051) 4.6 MB/s 9.2 MB
Receiving objects: 65% (1992/3051) 7.1 MB/s 28.3 MB
Receiving objects: 97% (2978/3051) 5.9 MB/s 35.6 MB
INFO:otpush.receive:Updating repo metadata with ostree --repo=/ostree/repo summary --update
davis@NUC10i5FNH:~/src/build/tmp/deploy/images/raspberrypi4$ 
```

- Check commits on ostree server



```
davis@ubuntu22:~$ ostree refs
raspberrypi4
davis@ubuntu22:~$ ostree log raspberrypi4
commit b2f3ccd9d8beecc10bcef2ef66670461406b65fe3e32a1ca4f3ac2612f144aa9
ContentChecksum:  bd81cbb6cda78876cc64cffd419857f0b488f896c102afb811b02b1bfa5cdcdd
Date:  2023-09-11 00:28:49 +0000
Version: 1.0

    Commit-id: core-image-minimal-raspberrypi4-20230911002649

davis@ubuntu22:~$ ostree summary -v
* raspberrypi4
    Latest Commit (198 bytes):
      b2f3ccd9d8beecc10bcef2ef66670461406b65fe3e32a1ca4f3ac2612f144aa9
    Version (ostree.commit.version): 1.0
    Timestamp (ostree.commit.timestamp): 2023-09-10T17:28:49-07

Repository Mode (ostree.summary.mode): archive-z2
Last-Modified (ostree.summary.last-modified): 2023-09-10T20:03:04-07
Has Tombstone Commits (ostree.summary.tombstone-commits): No
ostree.summary.indexed-deltas: true
```



Index of /ostree-repo

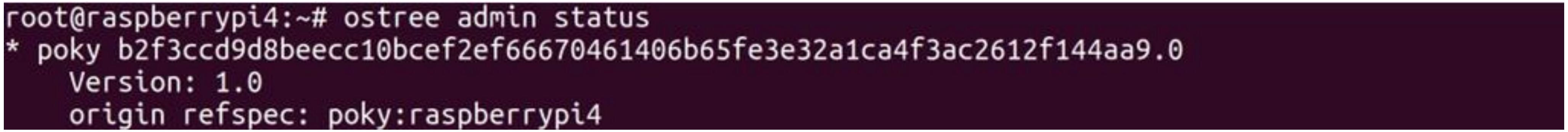
Name	Last modified	Size	Description
Parent Directory		-	
config	2023-09-09 17:40	58	
extensions/	2023-09-09 16:54	-	
objects/	2023-09-09 17:40	-	
refs/	2023-09-09 16:54	-	
state/	2023-09-09 17:40	-	
summary	2023-09-09 17:40	306	
tmp/	2023-09-09 17:40	-	

Apache/2.4.52 (Ubuntu) Server at ostree-server Port 80

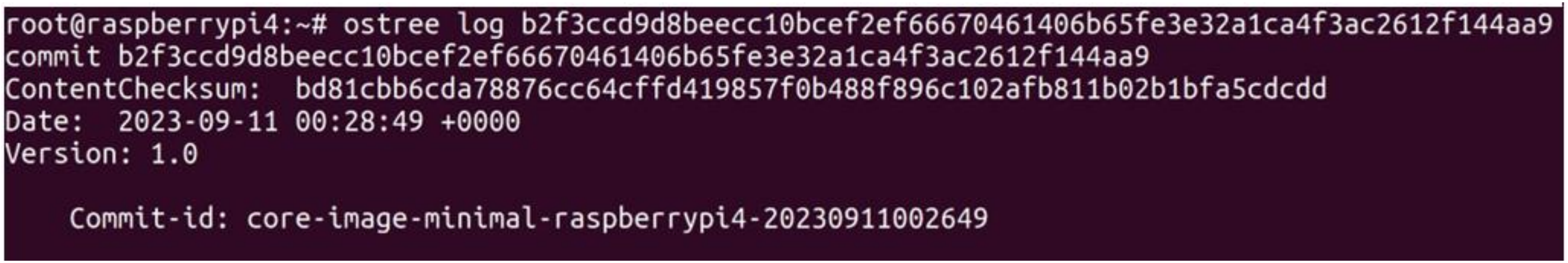
Objectives - recap

- ✓ ~~Discuss ostree prerequisites~~
- ✓ ~~Ostree vs libostree and why is it hard to search on youtube for ostree related content.~~
- ✓ ~~package vs image vs differential updates~~
- ✓ ~~Push/pull/offline/distributed architectures.~~
- ✓ ~~Get started with an ostree enabled raspberrypi4 build using the yocto project.~~
- ✓ ~~Pushing to an ostree server (even though its not natively supported).~~
- ❑ Command line basics
- ❑ Booting into an ostree commit
- ❑ Yocto integration

Check if system ostree enabled

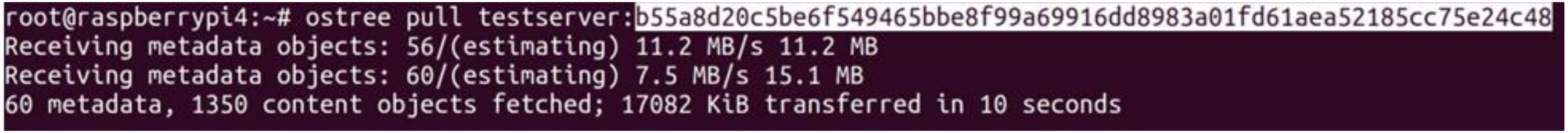


```
root@raspberrypi4:~# ostree admin status
* poky b2f3ccd9d8beecc10bcef2ef66670461406b65fe3e32a1ca4f3ac2612f144aa9.0
    Version: 1.0
    origin refspec: poky:raspberrypi4
```



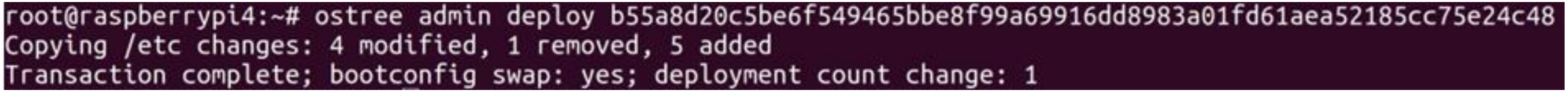
```
root@raspberrypi4:~# ostree log b2f3ccd9d8beecc10bcef2ef66670461406b65fe3e32a1ca4f3ac2612f144aa9
commit b2f3ccd9d8beecc10bcef2ef66670461406b65fe3e32a1ca4f3ac2612f144aa9
ContentChecksum:  bd81cbb6cda78876cc64cffd419857f0b488f896c102afb811b02b1bfa5cdcdd
Date:  2023-09-11 00:28:49 +0000
Version: 1.0

    Commit-id: core-image-minimal-raspberrypi4-20230911002649
```

Pulling specific commits from ostree server



```
root@raspberrypi4:~# ostree pull testserver:b55a8d20c5be6f549465bbe8f99a69916dd8983a01fd61aea52185cc75e24c48
Receiving metadata objects: 56/(estimating) 11.2 MB/s 11.2 MB
Receiving metadata objects: 60/(estimating) 7.5 MB/s 15.1 MB
60 metadata, 1350 content objects fetched; 17082 KiB transferred in 10 seconds
```

Deploying into a commit



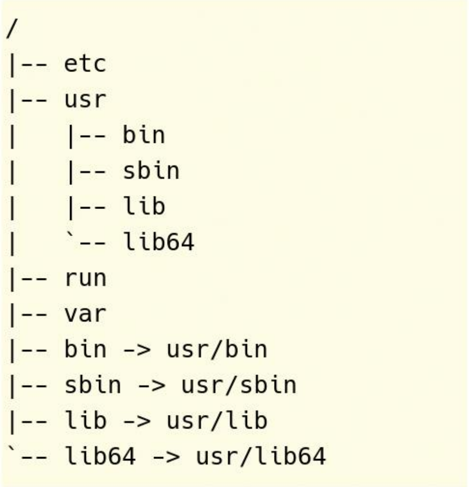
```
root@raspberrypi4:~# ostree admin deploy b55a8d20c5be6f549465bbe8f99a69916dd8983a01fd61aea52185cc75e24c48
Copying /etc changes: 4 modified, 1 removed, 5 added
Transaction complete; bootconfig swap: yes; deployment count change: 1
```

Cleanup

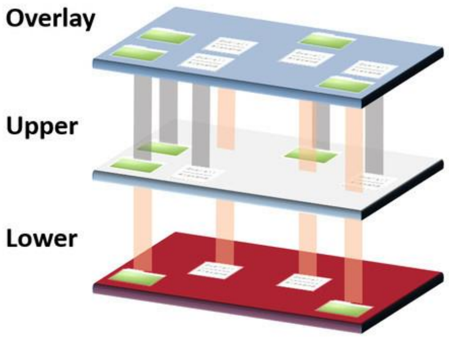
- Over time, the client’s ostree repository grows and can eventually fill up the filesystem.
- It’s recommended to periodically run ‘`ostree admin cleanup`’
  - Removes commits which are not associated with deployments.

Can changes be made to /usr?

- /usr is read-only by design
- usr-merge design pattern is used
  - Majority of filesystem is read-only
  - *“openSUSE Conference 2022 - usrmerge and beyond”*
    - https://www.youtube.com/watch?v=GRdbI7fwIo0
- `ostree admin unlock`
  - Applies an overlayfs over /usr allowing for modifications
  - Removed after a reboot.
  - Good for quick tests
- `ostree admin unlock --hotfix`
  - Applies an overlayfs over /usr allowing for modifications
  - Persistent across reboots
  - Useful during application development
  - After deploying a new commit, overlayfs is thrown away!

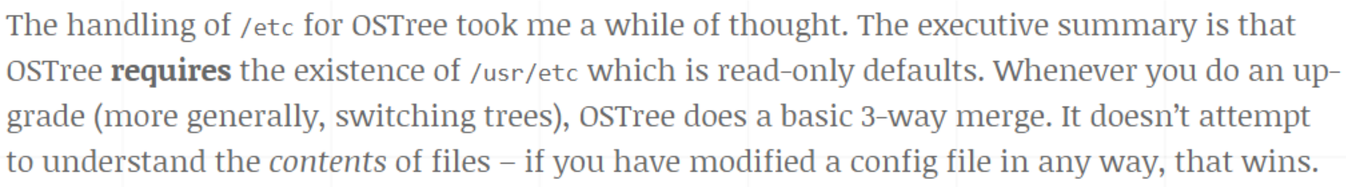


```
/
|-- etc
|-- usr
|    |-- bin
|    |-- sbin
|    |-- lib
|    `-- lib64
|-- run
|-- var
|-- bin -> usr/bin
|-- sbin -> usr/sbin
|-- lib -> usr/lib
`-- lib64 -> usr/lib64
```



Can changes be made to /etc?

- Technically yes, but here be dragons
- ostree uses the ‘3 way merge scheme’
  - In my experience, this is a big source of frustration for ostree beginners.
- TLDR
  - If a file */etc/foobar* is locally modified and later */etc/foobar* is updated in a newer build, your locally modified version of */etc/foobar* is kept.
    - However, the newer read-only default version of *foobar* is stored in */usr/etc*
- `ostree admin config-diff`
  - ostree can inspect /etc and inform us about what has been added, modified , and deleted relative to its current commit.
  - If /etc/foobar shows up as modified then use ‘*`cp /usr/etc/foobar /etc`*’
- From Colin Walter’s blog post – *why ostree require’s “/usr/etc”:*
  - https://blog.verbum.org/2014/01/24/why-ostree-requires-usretc/



The handling of `/etc` for OSTree took me a while of thought. The executive summary is that OSTree **requires** the existence of `/usr/etc` which is read-only defaults. Whenever you do an upgrade (more generally, switching trees), OSTree does a basic 3-way merge. It doesn’t attempt to understand the *contents* of files – if you have modified a config file in any way, that wins.

Does ostree protect against corruption due to power-loss?

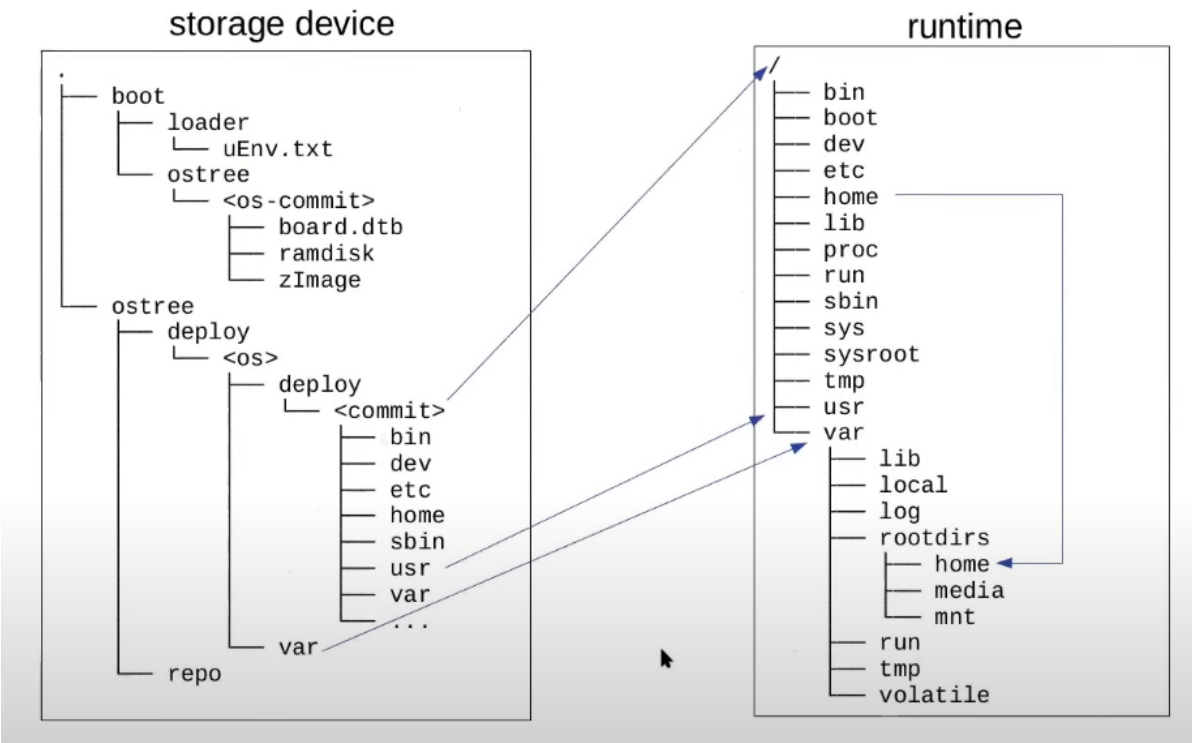
- ❑ Short answer: YES!
- ❑ Long answer:
  - ❑ Watch Drew Moseley’s webinar on “Implementing Power-safe Atomic Over the Air Updates”
  - ❑ https://www.youtube.com/watch?v=sdox0gwakX8

I know someone will ask so I figured I’d drop it here. ☺

Objectives - recap

- ✓ ~~Discuss ostree prerequisites~~
- ✓ ~~Ostree vs libostree and why is it hard to search on youtube for ostree related content.~~
- ✓ ~~package vs image vs differential updates~~
- ✓ ~~Push/pull/offline/distributed architectures.~~
- ✓ ~~Getting started with an ostree enabled raspberrypi4 build using the yocto project.~~
- ✓ ~~Pushing to an ostree server (even though its not natively supported).~~
- ✓ ~~Command line basics~~
- □ Booting into an ostree commit
- □ Yocto integration

How do we boot into an ostree commit?

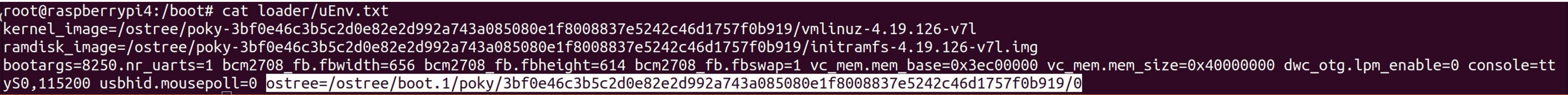


storage device

```
.
├── boot
│   ├── loader
│   │   └── uEnv.txt
│   └── ostree
│       └── <os-commit>
│           ├── board.dtb
│           ├── ramdisk
│           └── zImage
└── ostree
    ├── deploy
    │   └── <os>
    │       ├── deploy
    │       │   └── <commit>
    │       │       ├── bin
    │       │       ├── dev
    │       │       ├── etc
    │       │       ├── home
    │       │       ├── sbin
    │       │       ├── usr
    │       │       ├── var
    │       │       └── ...
    │       └── var
    └── repo
```

runtime

```
/
├── bin
├── boot
├── dev
├── etc
├── home
├── lib
├── proc
├── run
├── sbin
├── sys
├── sysroot
├── tmp
├── usr
└── var
    ├── lib
    ├── local
    ├── log
    ├── rootdirs
    │   ├── home
    │   ├── media
    │   └── mnt
    ├── run
    ├── tmp
    └── volatile
```



```
root@raspberrypi4:/boot# cat loader/uEnv.txt
kernel_image=/ostree/poky-3bf0e46c3b5c2d0e82e2d992a743a085080e1f8008837e5242c46d1757f0b919/vmlinuz-4.19.126-v7l
ramdisk_image=/ostree/poky-3bf0e46c3b5c2d0e82e2d992a743a085080e1f8008837e5242c46d1757f0b919/initramfs-4.19.126-v7l.img
bootargs=8250.nr_uarts=1 bcm2708_fb.fbwidth=656 bcm2708_fb.fbheight=614 bcm2708_fb.fbswap=1 vc_mem.mem_base=0x3ec00000 vc_mem.mem_size=0x40000000 dwc_otg.lpm_enable=0 console=tt
yS0,115200 usbhid.mousepoll=0 ostree=/ostree/boot.1/poky/3bf0e46c3b5c2d0e82e2d992a743a085080e1f8008837e5242c46d1757f0b919/0
```

initramfs

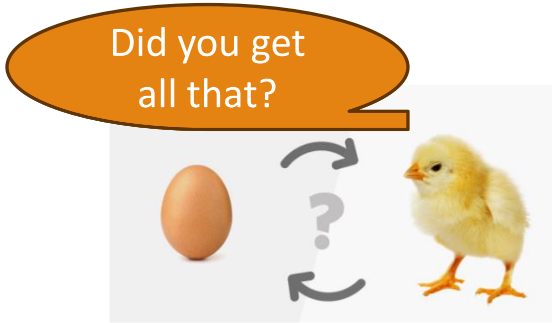
- The initramfs helps with ostree’s chicken and egg problem.

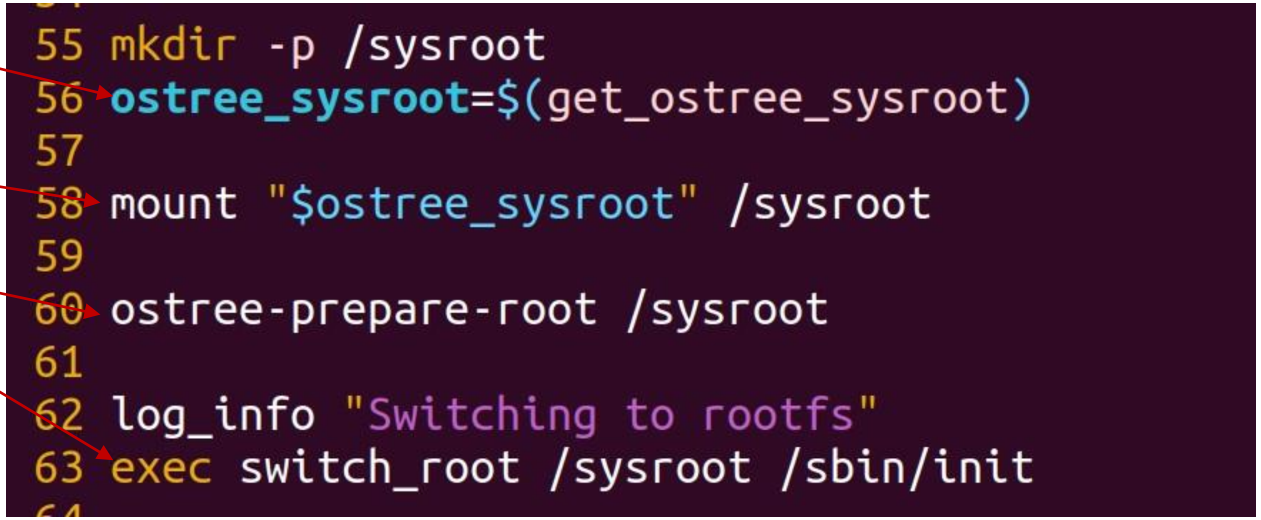
- The kernel cannot start the init process until it switches into the deployment and it cannot know what deploymet to use until it mounts the filesystem and to mount, it needs to know which deployment to use and so on and so on.

- On boot up, the bootloader, uboot, instructs the kernel to start the initramfs which contains a tiny filesystem with an init script whose job it is to parse the kernel command line to figure out which deployment to use.

- The initramfs mounts the “physical sysroot” onto /sysroot

- The init process performs a switch_root using /sysroot. From this point forward, /sbin/init (or even systemd) continues with the deployment as its new root.





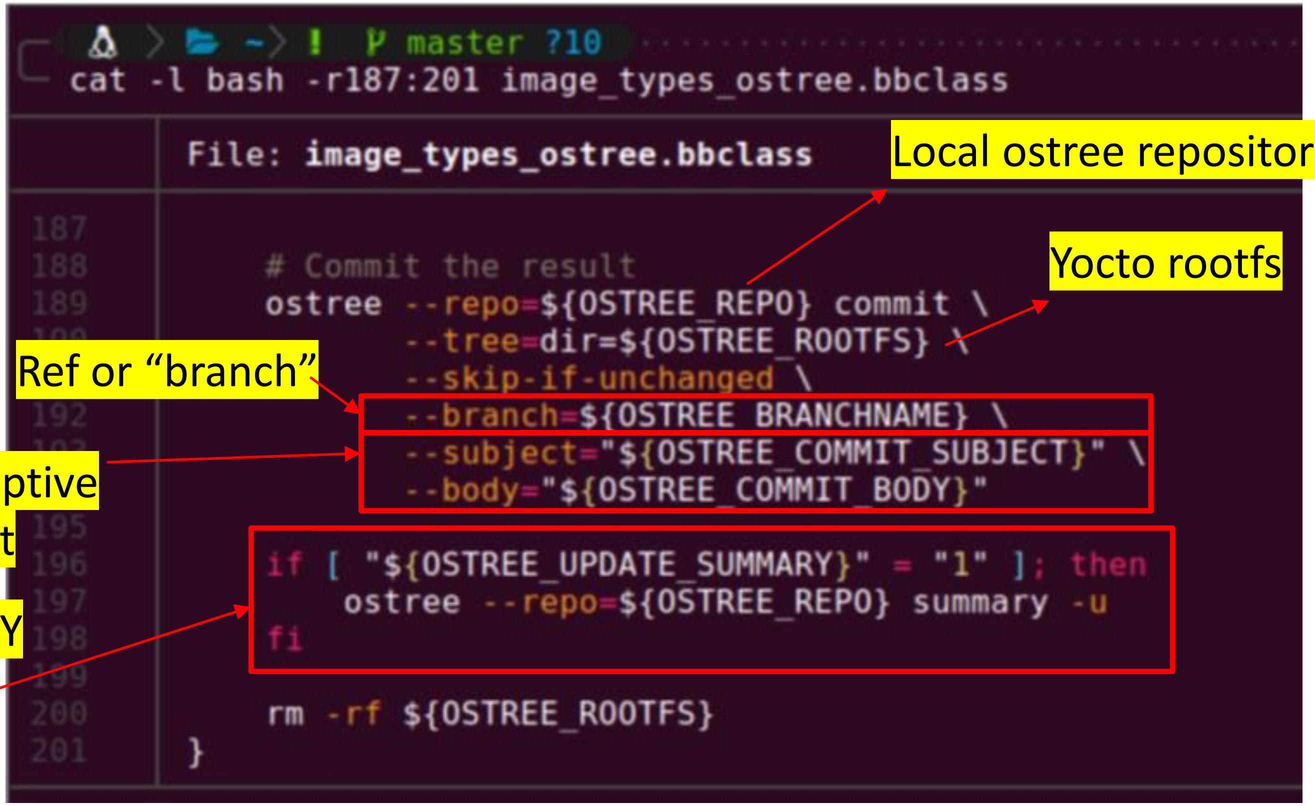
```
mkdir -p /sysroot
ostree_sysroot=$(get_ostree_sysroot)

mount "$ostree_sysroot" /sysroot

ostree-prepare-root /sysroot

log_info "Switching to rootfs"
exec switch_root /sysroot /sbin/init
```

Objectives - recap

- ✓ ~~Discuss ostree prerequisites~~
- ✓ ~~Ostree vs libostree and why is it hard to search on youtube for ostree related content.~~
- ✓ ~~package vs image vs differential updates~~
- ✓ ~~Push/pull/offline/distributed architectures.~~
- ✓ ~~Get started with an ostree enabled raspberrypi4 build using the yocto project.~~
- ✓ ~~Pushing to an ostree server (even though its not natively supported).~~
- ✓ ~~Command line basics~~
- ✓ ~~Booting into an ostree commit~~
- □ Yocto integration

Yocto integration

- ❑ Heavy lifting is done by meta-updater
- ❑ Builds target and native ostree binaries
- ❑ After building initramfs, image_types_ostree.bbclass commits yocto's final rootfs into the buildsystem's ostree repository



```
master ?10
cat -l bash -r187:201 image_types_ostree.bbclass
```

```
File: image_types_ostree.bbclass

    # Commit the result
    ostree --repo=${OSTREE_REPO} commit \
           --tree=dir=${OSTREE_ROOTFS} \
           --skip-if-unchanged \
           --branch=${OSTREE_BRANCHNAME} \
           --subject="${OSTREE_COMMIT_SUBJECT}" \
           --body="${OSTREE_COMMIT_BODY}"

    if [ "${OSTREE_UPDATE_SUMMARY}" = "1" ]; then
        ostree --repo=${OSTREE_REPO} summary -u
    fi

    rm -rf ${OSTREE_ROOTFS}
}
```

Local ostree repository

Yocto rootfs

Ref or "branch"

Can be as descriptive as you want

Checks that OSTREE_UPDATE_SUMMARY IS SET IN ORDER TO AUTOMATICALLY UPDATE SUMMARY FILE

Yocto Integration (cont)

- ❑ Prepare physical sysroot
- ❑ Prepare deployment sysroot (aka hardlink farm)
  - ❑Located in `/ostree/deploy/`**`$STATEROOT`**`/deploy/`**`$COMMIT`**
- ❑ Make bootloader and initramfs work together to boot the deployment
  - ❑Involves setting up uEnv.txt so that uboot knows which deployment to pass on the kernel command line
- ❑ Make sure you control mutable state in your system.
  - ❑ Operating system and application binaries under **/usr** must always be R/O
  - ❑ Application runtime data under **/var** should be R/W
    - ❑Initial setup of files and directories in **/var** can be created using systemd-tmpfiles during first boot
    - ❑Templated config files can live in **/usr** but must be copied to **/var** during initial boot. Apps should use the copy in **/var**
- ❑ First 3 steps are already done by meta-updater

Final thoughts and recommendations

- ❑ Ostree is a powerful and flexible update technology
- ❑ Periodically monitor the size of your ostree repo otherwise it'll eventually use up all your disk space
  - ❑ '`ostree admin cleanup`' is your friend
- ❑ Periodically monitor the amount of memory used
  - ❑ Especially if your rootfs is larger than your available memory. This may cause you to anger the "*out of memory killer*" at times.
- ❑ Use '`ostree admin unlock`' instead of just remounting /usr as R/W whenever possible so you avoid possibly corrupting your filesystem
- ❑ Whether to use a the pull, push or offline scheme is use case dependent.

If you forget all else, just remember this.

Mutability is the root of all evil. Choose immutable filesystems whenever possible.

References

- Advanced Telematics – meta-updater layer documentation
  - https://github.com/advancedtelematic/meta-updater
- Official ostree documentation:
  - https://ostreedev.github.io/ostree/
- "Empowering the next billion with OSTree, Flatpak, NDN, and the cloud" by Srdjan Grubor
  - https://www.youtube.com/watch?v=KFwBu-_k3rM
- StarlingX Using OSTree Atomic Updates to Drastically Reduce Outage Times for Upgrades
  - https://www.youtube.com/watch?v=GXCz6vDzM8U
- Software updates with OSTree Why and how
  - https://www.youtube.com/watch?v=81Jp4Jp_pNc
- Open Software Updates for IoT - Phil Wise, ATS Advanced Telematic Systems GmbH
  - https://www.youtube.com/watch?v=slq0Hhu5Bx4
- OSTree CLI for OS management
  - https://www.youtube.com/watch?v=B0xvrXkEwr4
- “Designing OSTree based embedded Linux systems with the Yocto Project” by Sergio Prado
  - https://www.youtube.com/watch?v=4aAnXAU-oX8

Questions?

Thank you!

LINUXCON

@

THE LINUX FOUNDATION

OPEN SOURCE SUMMIT

EUROPE

THE LINUX FOUNDATION