



Accelerating Prototype to Commercial Device with Snapdragon and Ubuntu Core

Oksana Wilcox, Staff Manager
Qualcomm Technologies, Inc.

February 21, 2017

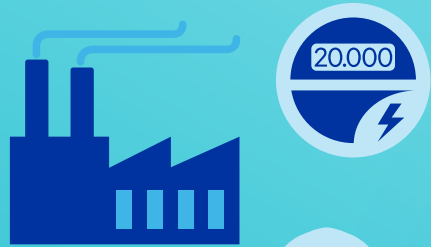
Smartphone innovations are transforming IoT

Mobile technology is the design point for other smart, connected devices



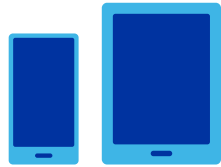
Snapdragon for embedded computing

foundation to address
broad use cases

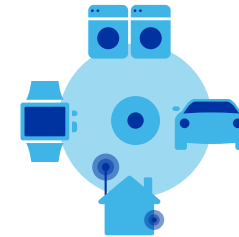


Bringing Snapdragon to embedded devices

Identifying the challenges



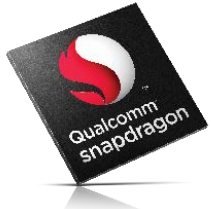
Mobile OEMs



Embedded Customers

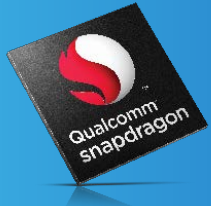
Relationship	○ High touch, 1-1	○ Low-touch, web-based
Primary fulfillment	○ Direct	○ Distribution
Minimum order	○ 10,000s	○ 100
Customers	○ High dependency, few	○ Low dependency, many
Roadmap influence	○ Strong	○ Weak
Engineering capability	○ Strong, large teams	○ Varied, small teams
Primary support	○ Direct	○ Web-based/Contract work
End-product volume	○ High	○ Low
Design type	○ Iterative	○ Clean-slate

Focused, tiered approach designed for longevity



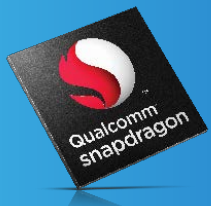
Snapdragon premium tier

Commercial modules available for Snapdragon 820



Snapdragon 600E

1.5 GHz quad-core Qualcomm® Krait™ 300 CPU



Snapdragon 410E

1.2 GHz quad-core ARM v8 Cortex-A53, 32/64-bit capable

Supported for longevity

- Snapdragon 600E and 410E are available through distribution for a minimum of 10 years from commercial sample of mobile processor in 2015

Available for chip on board design

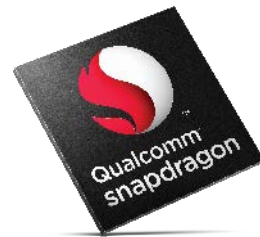
- 1st time Snapdragon processors are sold through 3rd party distribution via Arrow Electronics

Designed for performance in embedded devices

Snapdragon processors designed specifically for embedded applications

Snapdragon 410E

- **CPU:** 1.2 GHz quad-core ARM v8 Cortex-A53, 32/64-bit capable
- **Connectivity:** Integrated Wi-Fi, Bluetooth 4.xLE, and GPS
- **DSP:** Qualcomm® Hexagon™ DSP up to 500MHz
- **Graphics:** Qualcomm® Adreno™ 306 400MHz GPU with support for OpenGL ES 3.0/2.0/1.1, OpenCL 1.1e*, DirectX 9.3*
- **Interfaces:** 2x USB2.0, MIPI-CSI, MIPI-DSI, SD3.0 & eMMC v4.5 with DDR support
- **OS:** Android, Linux, Windows 10



410E
(8016E)
802.11b/g/n



Up to 13MP



600E
(8064E)
802.11a/b/g/n/ac
MU-MIMO



Up to 21MP

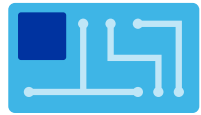
Snapdragon 600E

- **CPU:** Quad-core Qualcomm® Krait™ 300 CPU up to 1.5 GHz
- **Connectivity:** Wi-Fi, Bluetooth 4.0LE/3.x, and GPS
- **DSP:** Qualcomm® Hexagon™ DSP up to 500MHz
- **Graphics:** Qualcomm® Adreno™ 320 400+ MHz GPU with support for OpenGL ES 3.0/2.0/1.1, OpenCL 1.1e*
- **Interfaces:** SATA, PCIe 2.0, HDMI, LVDS, HSIC, 3x USB2.0, 3x MIPI-CSI, 2x MIPI-DSI, SD3.0 & eMMC v4.5 with DDR support
- **OS:** Android, Linux

*OS dependent

Building the Snapdragon embedded business

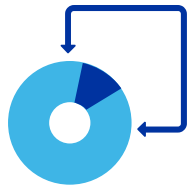
Key customer requirements



Community Board



Snapdragon
Technology Providers



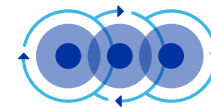
Software Ecosystem



Distribution



Extensive
Documentation



Component
Ecosystem



Longevity



Commercialization
Support

Clear path from prototype to commercialization

Ready to support consumer, commercial and industrial device needs

Develop & Prototype

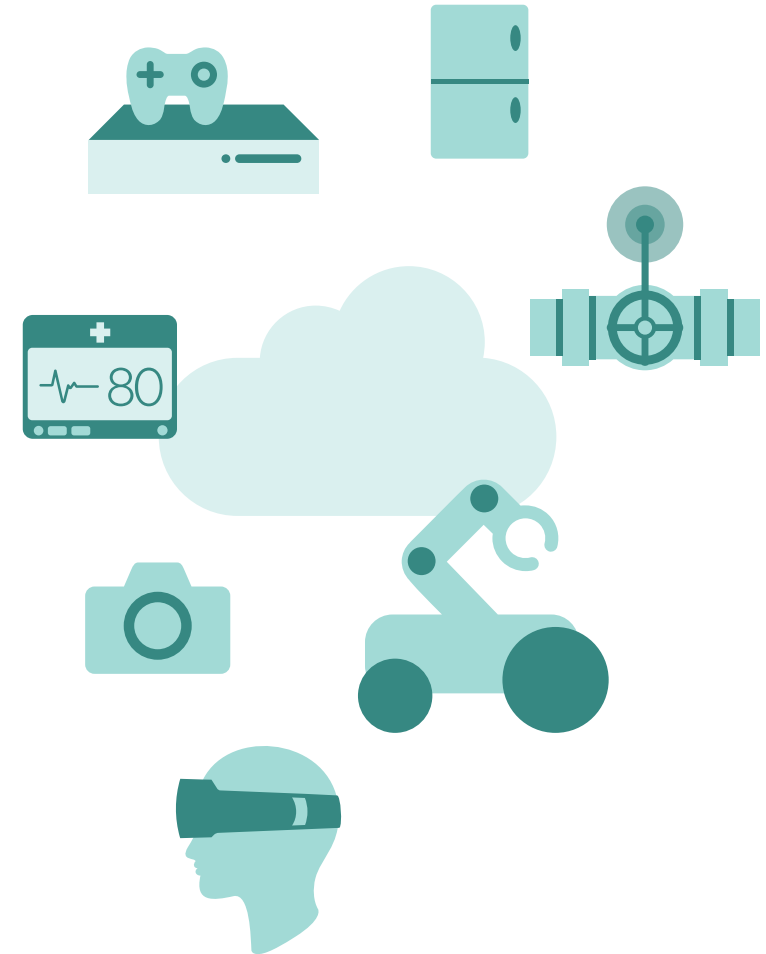
Use DragonBoard™ 410c for evaluation and prototype development

Arrow Electronics

Design & Manufacture

Flexibility to use off-the-shelf, commercial ready or custom SOMs and SBCs or discrete parts for chip on board design

Arrow Electronics
eInfochips
Inforce Computing
Intrinsyc Technologies
Variscite
and others



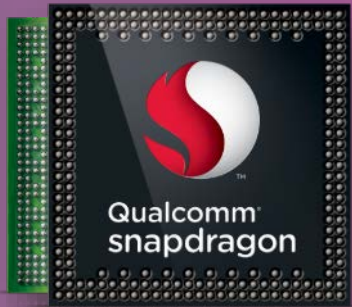
Software ecosystem

Supporting great flexibility for architecting IoT solutions



OS

- Android
- Linux
 - Debian
 - OpenEmbedded
 - Ubuntu Core
- Windows 10 IoT Core



Middleware

- AllJoyn®
- IBM Watson IoT
- ROS (Robotics Operating System)



Cloud

- AT&T M2X
- AWS IoT
- IBM Bluemix
- Microsoft Azure IoT

Community Board

Builds SW strengths and cultivates the customers of tomorrow

Benefits of community board

- Builds SW ecosystem
 - Enables community SW development
 - Middleware/Cloud service enablement platform
- Influences design in at prototype stage
 - Makers/ maker pros of today are startups of tomorrow
 - Embedded customers need evaluation platform



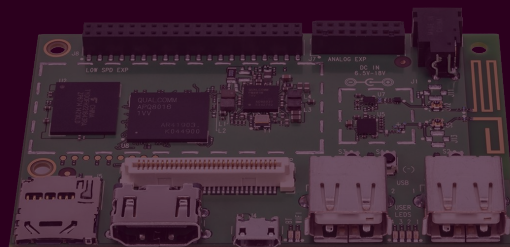
DragonBoard™ 410c - built with path to commercialization

- 96Boards open HW specification
 - Compatibility with 96Boards mezzanine products to enable easy prototyping
- Enabling mainline Linux support
- Commercial solution providers in place for industrial products/solutions



ubuntu core

From Prototype to Commercial Device with Snapdragon and **Ubuntu Core**



Kyle Fazzari

Software Engineer
kyle@canonical.com
Irc: kyrofa

Lowry Snow

Global Business Development
lowry.snow@canonical.com

Qualcomm + Canonical



- With Dragonboard 410c and Ubuntu Core, Qualcomm and Canonical are accelerating TTM for embedded computing:
 - Prebuilt kernel that enables all hardware components on the device
 - OS security and maintenance updates for the life of the Ubuntu LTS (five years through 2021)

Consultancy



Hardware & Device
Enablement



Security &
Maintenance Updates



Custom image and
support

Products



Device
Certification



Update
Control



Brand
Stores

Building your IoT Device

- Platform overview:
 - Ubuntu Core architecture
 - Roadmap
- From prototype to production:
 - An example: Nextcloud box
 - Components of an Ubuntu Core image
 - How to make your own customized flashable image
- Getting help

Platform Overview



What is Ubuntu Core?



Modular and simple architecture and packages: snaps!



Updated **transactionally**, worry free **updates** and **rollback**



Automatically **confines** applications for additional security



Provides an amazing developer experience with **snapcraft**



With a **store** to easily update devices and add/remove apps



Trusted cadence and **Long Term Support** releases

Simple packages: snaps



Package any app for a number of different Linux distributions (desktop, server, cloud or device), and deliver updates directly.



ubuntu



debian



gentoo linux

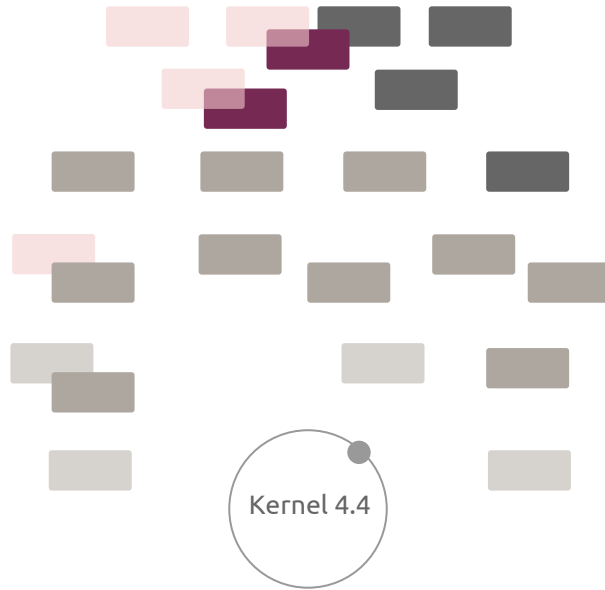


...

Modular and simple architecture



Classic



Legend:

- Application A
- Application B
- OS package
- Shared library
- Device driver

Ubuntu Core



Confined applications packaged as a snap with dependencies



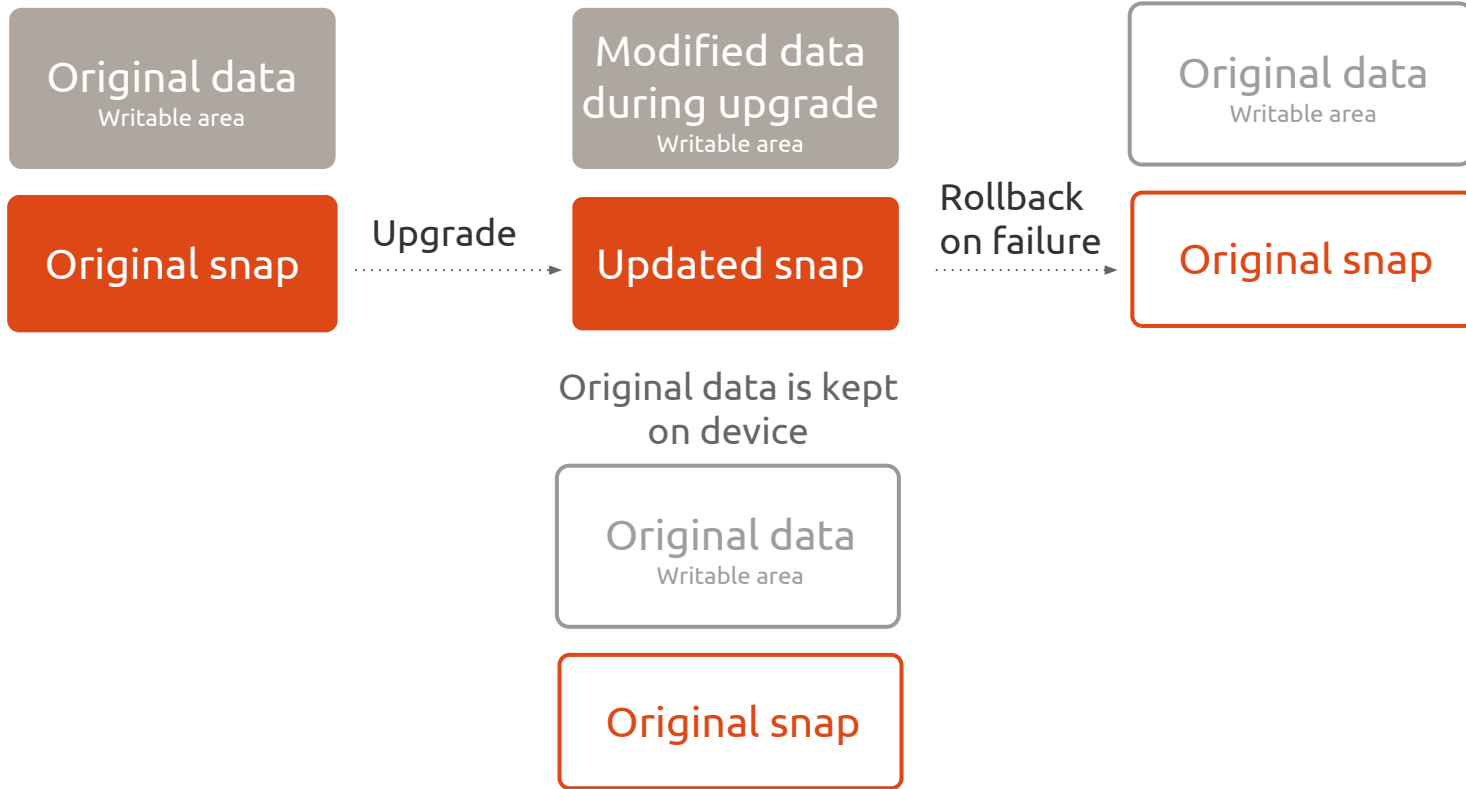
Minimal OS packaged as snap



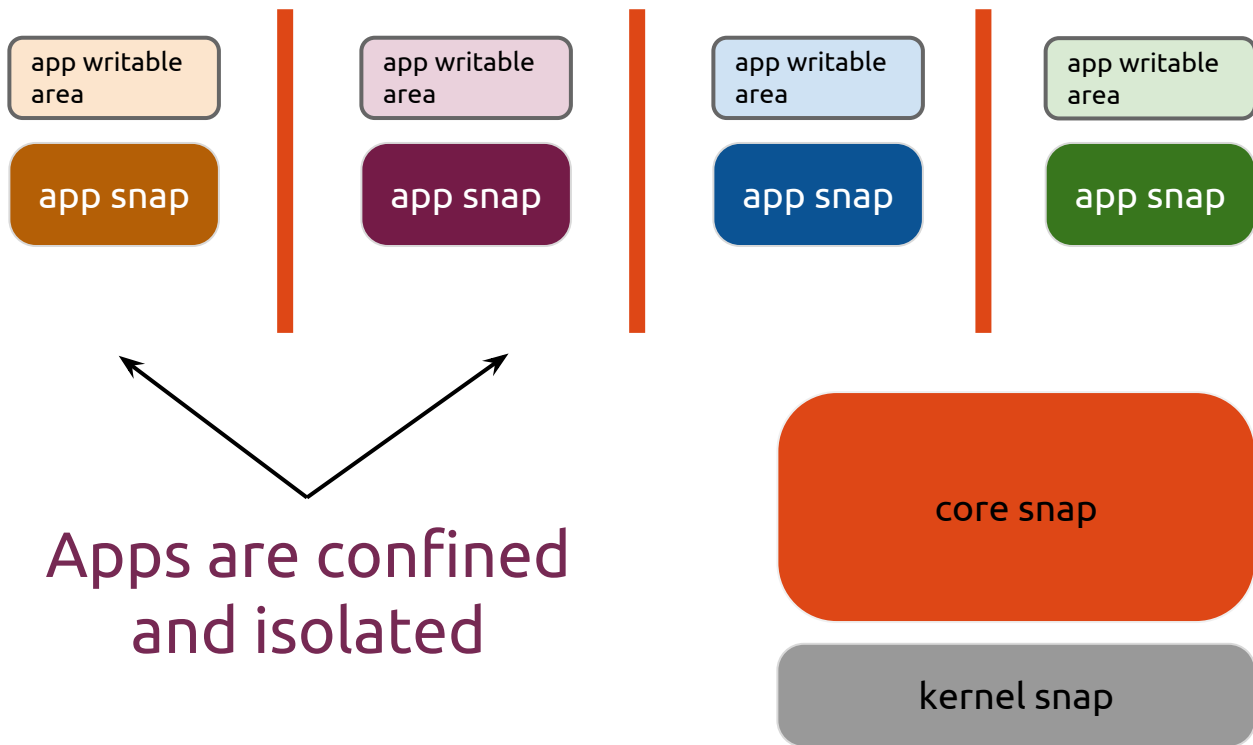
Clearly defined Kernel and device packaged as snap



Transactional updates: Apps, OS and kernel



Automatically **confines** applications



Amazing developer experience: **snapcraft**

<https://snapcraft.io>

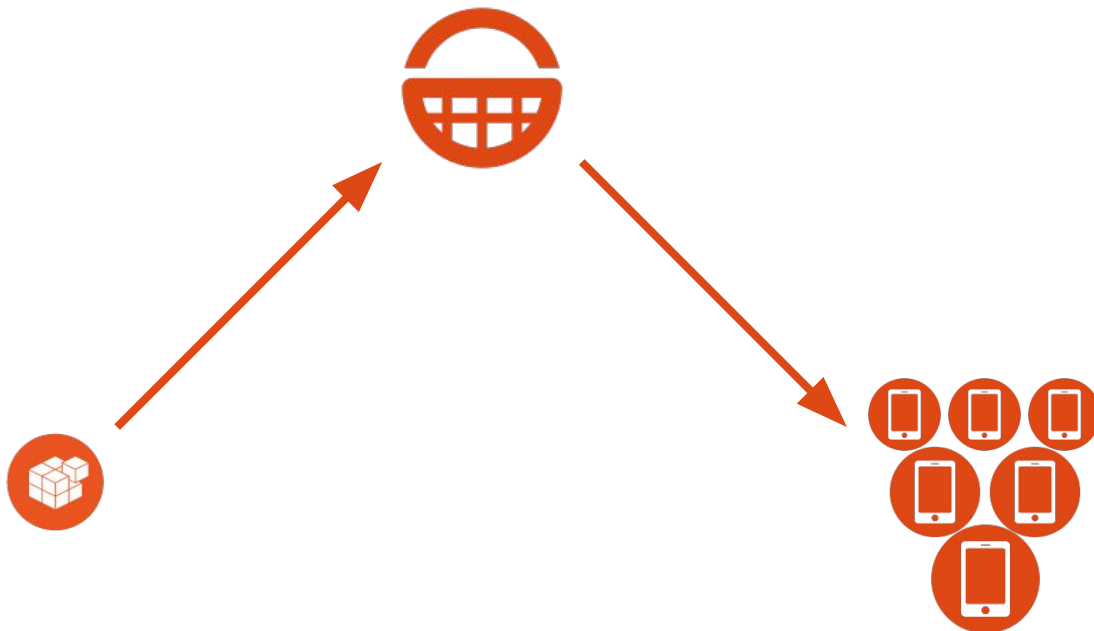


- **Snapcraft** creates snaps, orchestrating disparate components and build systems into one cohesive distributable package.
- It can re-use deb packages from Ubuntu.
- It's extensible and new plugins to leverage different technologies are being developed all the time. A few examples of its plugins are **Java, Python, Catkin (ROS), Go, CMake, qmake, make, etc.**

A **store** to manage your devices and updates

- Steps to deploy an update to all fielded devices:

1. Push an updated snap to the store.




Opportunities for new software revenue

ubuntu® Score Settings jamie Young

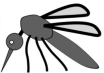






Connected grid router

Name	Connected grid router
Brand	Cisco®
Model	CGR1120
Interfaces	Bluetooth, webcam, WiFi
Docs	CGR Developer documentation
Serial	C02PQ53JFVH8
OS	Ubuntu 16
Uptime	235d : 12h : 48m : 25s



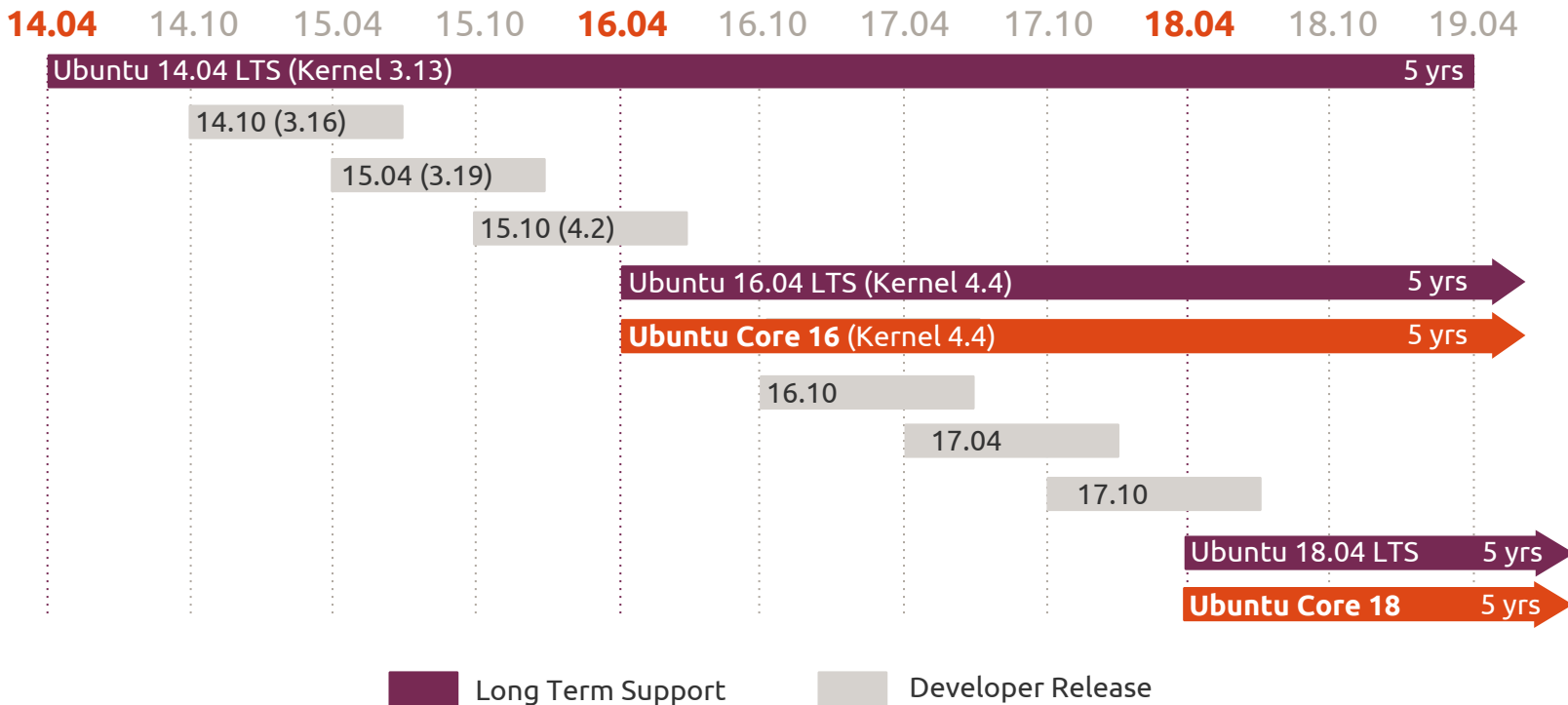
Installed snaps

[Add snaps](#)

 mosquitto Benjamin Cabé Active <input checked="" type="checkbox"/>	 nextcloud Nextcloud GmbH Active <input checked="" type="checkbox"/>	 canonical-pc Canonical Gadget <input type="checkbox"/>	 canonical-pc-linux Canonical Kernel <input type="checkbox"/>
 ubuntu-core Canonical Operating system <input type="checkbox"/>	 snapweb Canonical Device Manager <input type="checkbox"/>	 Add more snaps Browse more snaps from the Store	



Trusted by Linux developers



Prototype to production

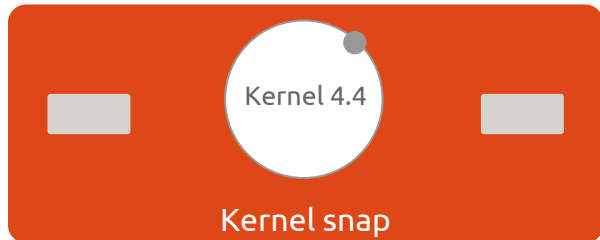


An example: Nextcloud box

From prototype to commercial device



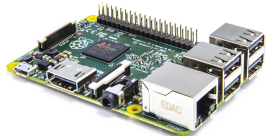
Core snap



Kernel snap



Gadget snap



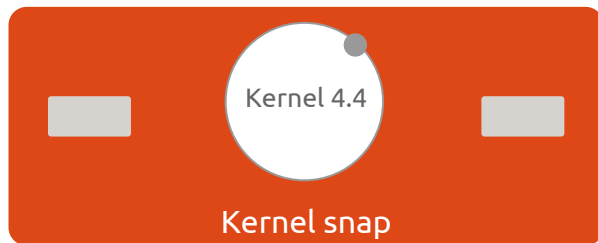
The **gadget** snap: make it boot

Gadget snap

- Bootloader
- Filesystem layout specification
- Default configuration
- **Using a DragonBoard? This is already provided and maintained for you**
 - (you can still make your own if you want)



The **kernel** snap: make it Linux

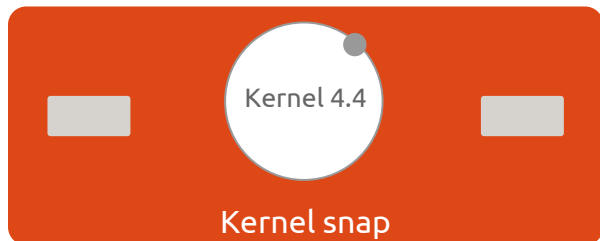


- 4.4-based kernel
- Device drivers
- **Using a DragonBoard? This is already provided and maintained for you**
 - (you can still make your own if you want)

Gadget snap



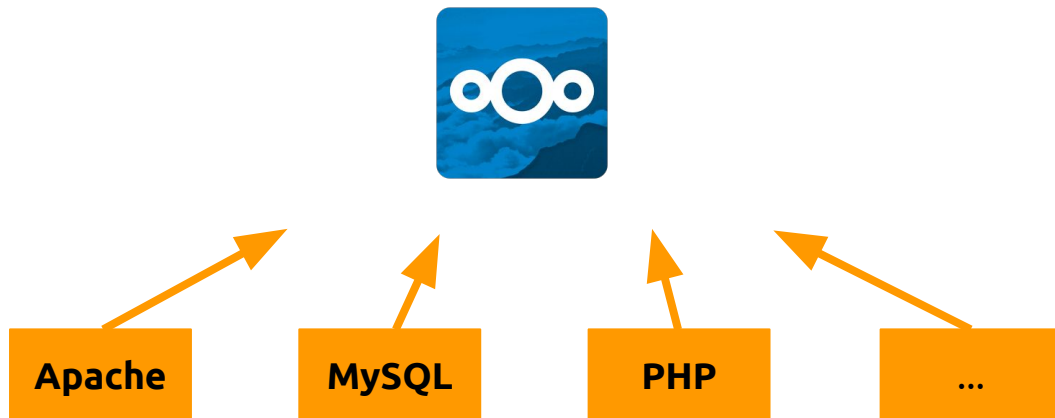
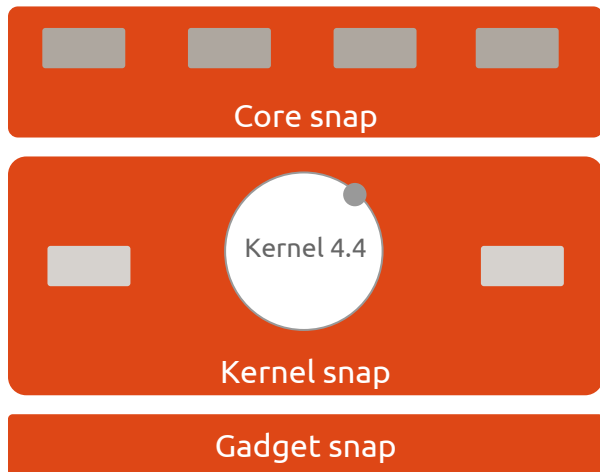
The **core** snap: make it Ubuntu



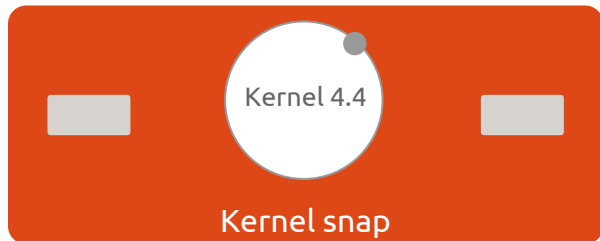
- Execution environment for app snaps
- Init system
- Basic services (networking, etc.)
- Basic files and libraries (libc, etc.)
- **Using a DragonBoard? This is already provided and maintained for you**
 - (you can still make your own if you want)

The **application** snaps: make it awesome

Use **Snapcraft** to assemble your snap from existing projects, leveraging different technologies.



Put it together: build a flashable image



You'll need:

- Gadget snap
- Kernel snap
- Core snap
- Whatever you want to add on top (in our example, Nextcloud)
- Store account
- ubuntu-image

Create your store account

Ubuntu Core will verify the image it's booting actually came from you. In order for that to happen, you need to create a signing key and register it with the store.

1. Visit <https://myapps.developer.ubuntu.com> and create your account.
2. Record your **account ID**. You'll need it in a minute.

Install ubuntu-image (and related tools)

```
$ sudo apt install ubuntu-image snapcraft snapd
```

- snapcraft/snapd: needed to generate, register, and use a signing key
- ubuntu-image: actually generate a flashable image

Create your key

In order to build an image, you need to be able to “assert” that this is your image with a key, then snapd has something to verify when it boots.

1. Generate a key that will be linked to your Ubuntu Store account:

```
$ snapcraft create-key my-key-name
```

2. Check that everything went OK and you have your key ready:

```
$ snapcraft list-keys
Name          SHA3-384 fingerprint
- my-key-name  Qjdfpj0EAW<snip>kkiZ41H4CR0y (not registered)
```

3. Register your new key with the store:

```
$ snapcraft register-key
Registering key ...
Done. The key “my-key-name” (Qjdfp<snip>H4CR0y) may be used to sign your assertions.
```

Create your model definition

dragon-model.json:

```
{
  "type": "model",
  "series": "16",
  "model": "nextcloud-dragon",
  "architecture": "arm64",
  "gadget": "dragonboard",
  "kernel": "dragonboard-kernel",
  "authority-id": "<account id>",
  "brand-id": "<account id>",
  "timestamp": "<timestamp>",
  "required-snaps": ["nextcloud"]
}
```

- What is your model's name? (nextcloud-dragon)
- Which Ubuntu Core series are you targeting? (16)
- What architecture is this image for? (arm64)
- Which gadget snap is being used? (dragonboard)
- Which kernel snap is being used? (dragonboard-linux)
- Who is defining this model? (your store account ID)
- When was this model defined?
 - `$ date -Iseconds --utc`
- What extra snaps are contained within this image? (nextcloud)

Create your model **assertion**

Now it's time to use our key to sign this model definition (thereby turning it into an **assertion**).

```
$ cat dragon-model.json | snap sign -k my-key-name > dragon.model
```

```
You need a passphrase to unlock the secret key for
```

```
user: "my-key-name"
```

```
4096-bit RSA key, ID 0B79B865, created 2016-01-01
```

```
...
```

After giving your passphrase, a dragon.model file is created containing the assertion.

Build and flash your image!

Once we have a signed model assertion, build the image using `ubuntu-image`, requesting that the **stable** channel be used when fetching the image's various snaps:

```
$ sudo ubuntu-image -c stable -o dragon.img dragon.model
```

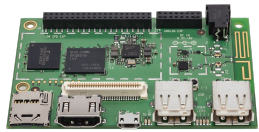
After a few minutes, you'll end up with **dragon.img**, which is a bootable Ubuntu Core image containing the components specified in your model definition (in our example, this includes Nextcloud). Flash it to an SD card, put it in a DragonBoard, and boot!

```
$ sudo dd if=dragon.img of=/dev/sdXX bs=32M; sync;
```

Let's recap: 1, 2, 3!

1

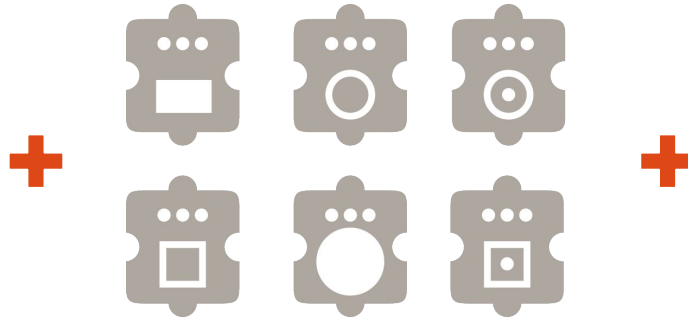
Pick the board



Qualcomm DragonBoard

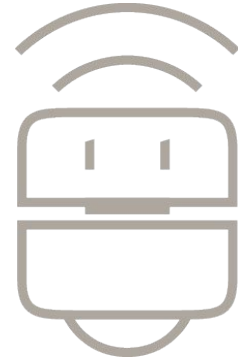
2

Develop & upload Snaps



3

Design a cool box



Getting help



Getting help and getting involved

- **snapcraft.io**
 - Snapcraft documentation and walkthroughs
- **tutorials.ubuntu.com**
 - Various Snapcraft and Ubuntu Core tutorials in a codelabs format
- **Ask a question on Ask Ubuntu (askubuntu.com)**
 - If you're stuck on a problem, someone else has probably encountered it too and they can help you. Take a look at the "snap" or "ubuntu-core" tags.
- **Join our real time chat**
 - IRC: #snappy on freenode.net
 - Rocket: <https://rocket.ubuntu.com/channel/snapcraft>