Desktop Linux as easy as a smartphone – **Just in a Snap!**

An introduction into the universal packaging format

Till Kamppeter <till.kamppeter@gmail.com>

CANONICAL ubuntu

Your application everywhere, just in a Snap!

What the hell are Snaps? And why should I use them?

What the hell are Snaps?

- You are developer of an **application**?
- Already thought about how it **gets distributed** to end users?

  ⇒ **This could turn people away from Linux!**

- You provide the **source code**
  - Only **tech-savvy users** can use it directly
  - You need **goodwill of distro maintainers** to package your app
  - Distro version released ⇒ No update of your app in this distro version ⇒ User always has to update to **newest distro version**
- **You package** your app, for 10+ distros and have to test on 10+ distros
- **That is a nightmare! Isn't it?**

What the hell are Snaps?

- You have a **smartphone**? There it is much easier: **Google Play Store**, **App Store**
- And remember that Canonical developed a **smartphone OS**?
- They have **learned** from it!

⇒ And now we have ...

Snap!

What the hell are Snaps?

- A method of **OS-distribution-independent packaging**
  - **You package and test once**, put your **Snap** into the **Snap Store**, and users of **any distro** (Ubuntu, Debian, SUSE, Red Hat, Windows, ...) can use it.
  - **All libraries and other dependencies** come with your Snap
  - **User experience as with smartphone apps**
- Your app runs in a **security shell** (AppArmor, seccomp, namespaces), isolated from the host system
  - So-called **sandboxed packaging**
  - Communication to outside only via **well-defined interfaces**
  - **Snap Store has control**, has to explicitly permit "dangerous" interfaces
  - This way we can **trust third-party apps**
  - We are not dependent any more on distro maintainers for secure packages

What the hell are Snaps?

- **Don't fear the daemons, we snap them, too!**
  - Snap is universal, not only desktop apps but also daemons, system utilities, sub-systems, drivers, operating system cores, kernels, … can get snapped
    => **All-Snap operating system**, like **Ubuntu Core Desktop**
- **Packaging moves from distros to upstream**
  - 10+ distros, each packaging XXX, inventing the wheel 10+ times
  - So let upstream, XXX.org, snap it, distros take the Snap
  - Distro devs concentrate on distro core or contribute to upstream code
  - Distro version released, app updates continue from upstream
- **Immutable distros, Immutable sub-systems, Immutable apps**
  - Ubuntu Core: **Immutable core**, all-Snap distro, desktop under development
  - Snaps are **immutable apps** (or **immutable sub-systems**, like the CUPS Snap)
  - Not that immutable, many system components, like printing stack or GPU drivers in separate Snaps

Your application everywhere, just in a Snap!

Snap Packages

Snap Package Properties

- Compressed and **GPG-signed read-only squashfs images**
- Includes **metadata** in a **`*.yaml`** file
- Installed Snap has a **writable file system area** inside its confinement
- Come in **4 types** (app, os, gadget, kernel)
- Support **transactional (atomic) updates** and **rollback**
- Can handle **binary diffs** for smaller download on upgrades
- **Available on multiple distros** and supported by default on all Ubuntu installs since Ubuntu 14.04

Snap Package Security

- **Read-only** file system image (squashfs)
- **GPG signed**
- **Confinement via**:
  - **AppArmor** (File system access rules)
  - **seccomp** (System call restrictions)
  - **Namespaces** (Separate resource spaces: PIDs, users, network, ...)
- **`snapd`** and **`snap-confine`** wrap around all executables in a snap, to ensure only the allowed writable dirs can be accessed
- Details: https://developer.ubuntu.com/en/snappy/guides/security-whitepaper/

Snap Package Security

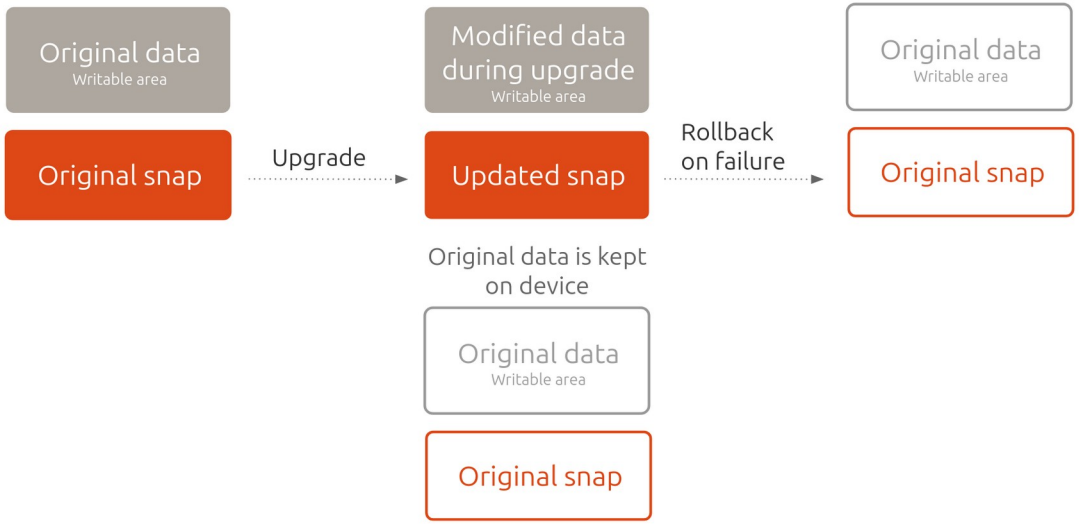
- **“root-safe”**
  - Applications can **run as root** but can not break out of the package confinement, **no need for specific user or group setup** to maintain security.
  - Example: **Daemon Snaps**
- **Interfaces**
  - **Slots**: Can provide access to features inside the Snap for other Snap packages (most slots come from core Snap)
  - **Plugs**: Can make use of features provided by slots
- **Rollback on error**
  - Transactional updates allow manual or automatic roll-back

Interfaces: Safe vs. Dangerous

- Snapped applications are **completely encapsulated** (AppArmor, seccomp, namespaces)
- By default, they cannot communicate with the host system or with other Snaps
- Communication is possible via **well-defined interfaces**: "network", "cups", "dbus", …
- A "**plug**" has to be connected with a "**slot**" of the system or of another Snap in order to communicate
  - **“Safe” interfaces**
    - Ex.: “cups” which allows listing available printers and printing
    - **are auto-connected** when installing from Snap Store
  - **“Dangerous” interfaces**
    - Ex.: “cups-control” which allows creating/removing printers, delete all jobs …
    - **need manual connection** or **permission** from Snap Store team for auto-connection

Updating Snaps

- Transactional (atomic) updates
- Current version and its writable area saved, for rollback
- Automatic rollback and reboot after kernel panic or boot failure

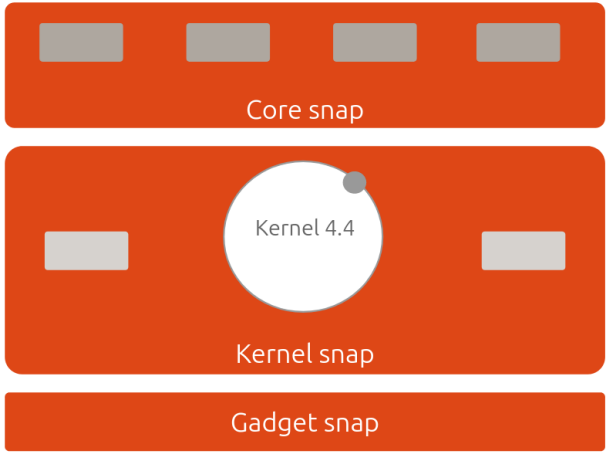


Your application everywhere, just in a Snap!

Ubuntu Core – all-Snap OS

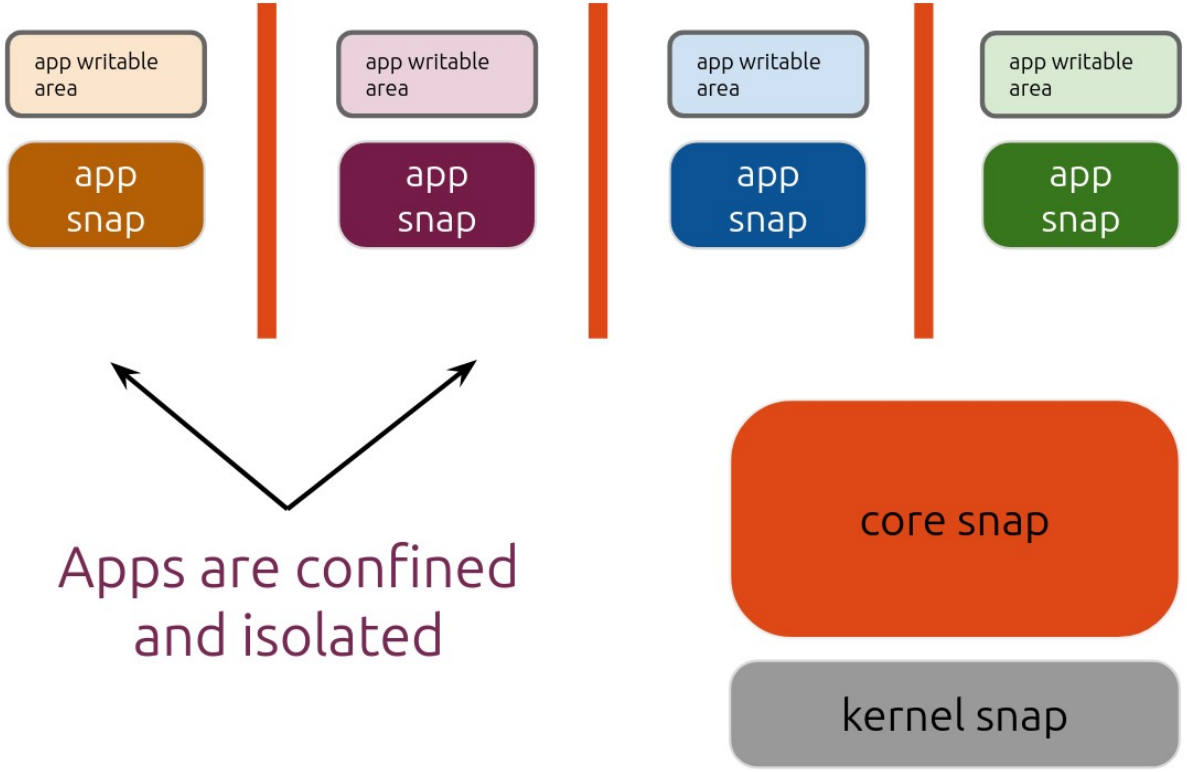
Ubuntu Core Operating System

- **Originally created for IoT …**
- **The all-Snap Ubuntu Core OS consists of**
  - **Gadget** Snap
    - Bootloader, partitioning, hardware specifics …
  - **Kernel** Snap
  - **Core** Snap
    - Minimum base operating system
    - core, core18, core20, core22, … based on Ubuntu LTS
- **Comes in one image but Snaps separately updateable**



No interdependencies between Snaps

- Every Snap can be **independently** updated (and rolled back)



The Core Snap

- On **classic** systems: Unified execution environment for all application Snaps
- On **all-Snap** systems: Core Snap is actually the **rootfs**
- **Just enough OS** to run the Snap management daemon (snapd), systemd and a minimal set of system services
- Provides the **system level Snap interfaces** (access to hardware, networking, control of the OS, modules, devices)
- Comes with a built in **configuration interface** to control certain aspects of the system via the snap set/get commands (en/disable services, set hostname, timeserver etc)

The Kernel Snap

- **Minimally patched** (the extra AppArmor bits can be found for various kernels at: http://kernel.ubuntu.com/git/jj/linux-apparmor-backports/)
- Ships with a **generic initrd** to set up writable area for the read-only core snap
- **Easily buildable** via snapcraft plugin, only needs a **`snapcraft.yaml`** in the tree
- Can be **any BSP kernel** (minimal reqs. 3.10 and the above mentioned AppArmor patch set)
- Can be **rolled back** at any time either manually or automatically on panic
- Requires certain set of **default config options** (a list can be found here: http://people.canonical.com/~ppisati/snappy_config/)

The Gadget Snap

- Defines the **partitioning of the image** and what bits get installed via the `gadget.yaml` file
- Ships the **bootloader** and bootloader config
- Can define **board-specific interfaces** and pre-connect them
- Can define **additional default snaps**

Example: A Kodi appliance image would define to install the Kodi Snap at image build time and auto-connect the slots and plugs for OpenGLES access, audio and removable media.

Your application everywhere, just in a Snap!

Ubuntu Core Desktop

Ubuntu Core Desktop – Building Blocks

- **Easy to maintain** for end users, like a **smartphone**
- **Boot Base** = Core Snap
- **Additional Bases:** Extra Core Snaps needed for Apps using other coreXX base Snap
- **Ubuntu Desktop Session Snap:** Wayland, Desktop environment (GNOME)
- All building blocks **independently updateable** and **exchangeable**

Ubuntu Core Desktop

- **Principally as Ubuntu Core**, but comes with
  - **Desktop Session Snap**
  - Common **Applications**
- A **Model** defines how the image gets composed: Specialized Kernel (games, broadcasting), desktop environment (GNOME, KDE, …), Applications
- Everything **easily** exchangeable: Other desktop, gaming kernel, …
- With modularity **difficult** to break ...

Desktop Session Snap

- **Wayland** user session
- **GNOME** running under usual Snap confinement
- All of the expected **desktop services** in a confined Snap

Your application everywhere, just in a Snap!

The Making of ...

snapcraft – Let’s go snapping ...

- **snapcraft** creates Snaps, orchestrating disparate components and building systems into one cohesive **distributable package**
- It can **re-use DEB packages** from Ubuntu (of the Ubuntu LTS release the Core Snap used is based on).
- 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, autotools, etc.

snapcraft – Let’s go snapping ...

- **Single `snapcraft.yaml` file** that describes everything
- Defines apps, build process, build dependencies, runtime dependencies, interfaces
- Fully supported and integrated in **Launchpad**
- GitHub build service provided via https://build.snapcraft.io/
- **Detailed documentation** and tutorials at https://snapcraft.io/

- Or come to our **snapping workshop** after the break!

ubuntu-image – Assemble your all-Snap OS!

- The **magic tool** putting everything together
- Using a signed “assertion” file to define which Snaps end up inside the image
- Reads **`gadget.yaml`** to create **partitioning**
- Can build full disk images (i.e. SD card) or multi-partition images (i.e. to dd single img files to specific eMMC partitions on a pre-partitioned flash device)
- Available as a Snap! (**`snap install ubuntu-image`** …)
- Detailed **documentation** at: https://docs.ubuntu.com/core/en/guides/build-device/image-building

Your application everywhere, just in a Snap!

Want to know more?

More info/links:

- Snap Store and home page of Snap:
  - https://snapcraft.io
- Discuss your questions in the forums:
  - https://forum.snapcraft.io/
- Documentation:
  - https://snapcraft.io/docs

More info/links:

- Learn about immutable OS distributions:
  - https://ubuntu.com/blog/ubuntu-core-an-immutable-linux-desktop
- Ubuntu Core Desktop – Introduction
  - https://discourse.ubuntu.com/t/ubuntu-core-desktop-deep-dive/
- Ubuntu Core Desktop – GitHub
  - https://github.com/canonical/ubuntu-core-desktop/
- Ubuntu Core Desktop – Installation HOWTO
  - https://www.omgubuntu.co.uk/2023/06/try-ubuntu-snap-desktop

More info/links:

- Ubuntu blogs from Oliver Smith about **optimizing performance of Snaps**:
  - https://ubuntu.com/blog/how-are-we-improving-firefox-snap-performance-part-1
  - https://ubuntu.com/blog/how-are-we-improving-firefox-snap-performance-part-2
  - https://ubuntu.com/blog/improving-firefox-snap-performance-part-3
  - https://ubuntu.com/blog/firefox-snap-updates-and-upgrades
- Want to watch some **snappy videos**? Here we go:
  - https://www.youtube.com/watch?v=TfB6QwR2GYg
  - https://www.youtube.com/watch?v=ido6kGmSHWI