Open Source MANO

Making OSM *“snappy”*

Adam Israel @ Canonical

Goals for making OSM *snappy*

- Fast, easy installation for end users
- Reliable path for upgrading and downgrading
- Path for testing release and pre-release components
- Improve the DevOps workflow for faster testing and releases

Why Snaps?

- Install alongside traditionally packaged software
- Ease packaging headache for developers
- Put software into the user’s hands faster
- Designed for any Linux-based Desktop, Server, Cloud, or Device
- And more...

Universal Linux Packaging

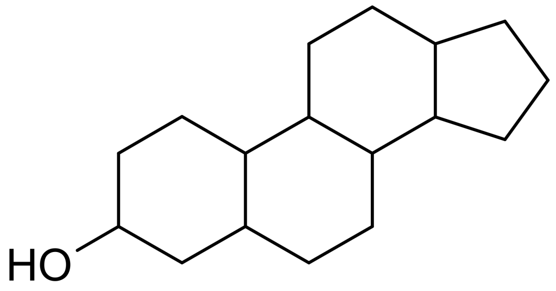
- Works on any Linux-based distribution or device
- Works on multiple architectures

Open Source MANO

Snaps are zip files on steroids

A snap contains an application, all its dependencies, and a description of how it should run safely on your system and interact with other software.

+



=

snappy

Snaps are fast to install

- Current installer takes ~1 hour (600M + dependencies) to install
- Snaps take ~= (600M including dependencies) 40 seconds to install*

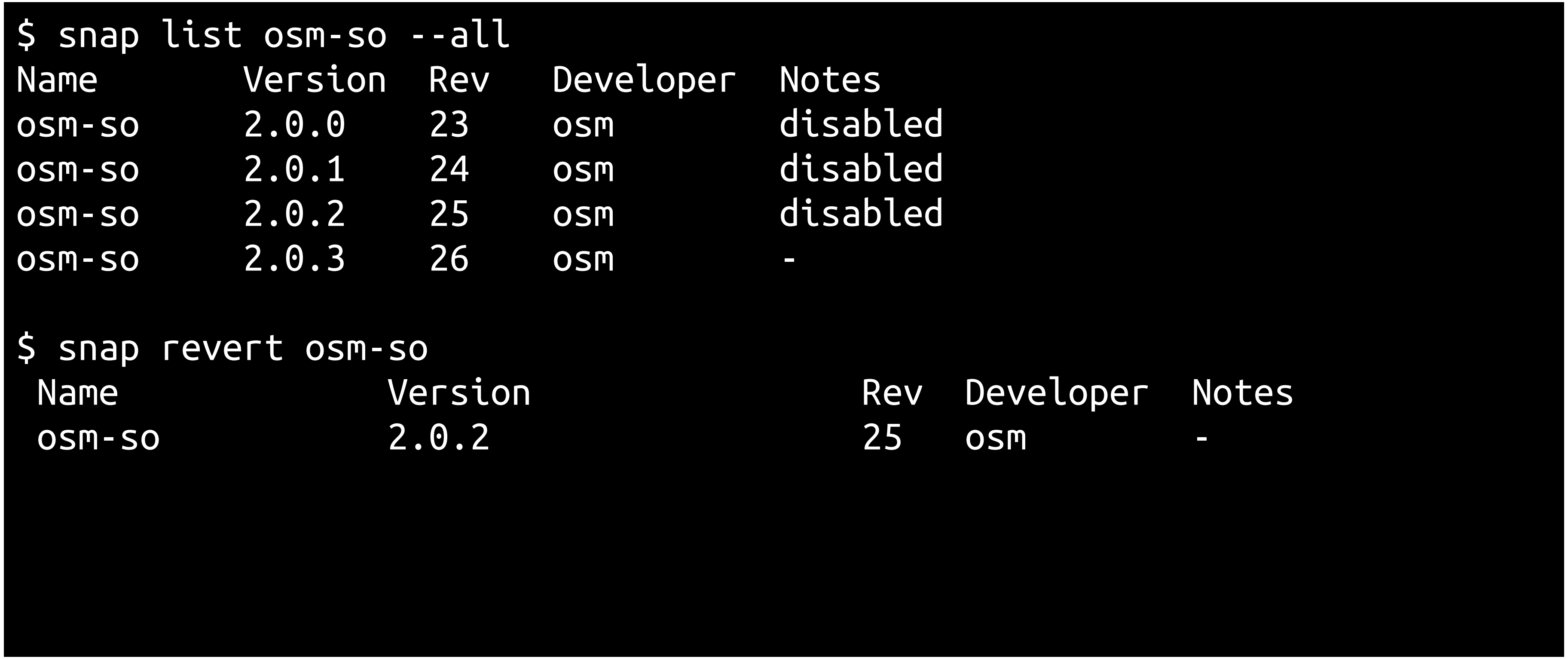
Final time will inc[illegible]stallation verification

Snaps are fresh

- Fast, reliable, automatic transactional updates
- The end user only downloads the diff between versions, so updates are smaller

```
$ snap refresh osm-so
osm-so 64.75 MB [=====================================>___]   12s
osm-so 2.0.3 installed
```

Snaps are transactional



```
$ snap list osm-so --all
Name        Version  Rev   Developer  Notes
osm-so      2.0.0    23    osm        disabled
osm-so      2.0.1    24    osm        disabled
osm-so      2.0.2    25    osm        disabled
osm-so      2.0.3    26    osm        -

$ snap revert osm-so
 Name                Version                      Rev  Developer  Notes
 osm-so              2.0.2                        25   osm        -
```

Snaps are versioned...

- Store data common to the application and specific to the version
- "Hooks" can be used to run important steps, i.e., migrating data from a previous version

```
/var/snap/osm-so/current/  ← $SNAP_DATA is the versioned snap data
directory
/var/snap/osm-so/common/   ← $SNAP_COMMON will not be versioned on
upgrades
```

...with channels!

- Release stable, candidate, beta, and edge versions of a snap
- Community decides how close to the bleeding edge to run
- Switch between channels with ease
- Mix and match!
- Publish to Snap Store or host your own

Mix and match snap channels

```
$ snap list
 Name              Version               Rev  Developer  Notes
 osm-ro            2.0.3                 26   osm        -
 osm-so            2.0.3                 26   osm        -
 osm-ui            2.0.3                 26   osm        -

$ snap refresh osm-ro --channel=edge

$ snap revert osm-ro
 Name             Version                Rev  Developer  Notes
 osm-ro            2.0.4-b908cb          33   osm        -
 osm-so            2.0.3                 26   osm        -
 osm-ui            2.0.3                 26   osm        -
```

Snaps are easier than Debian packaging

Debian

```
debian/
├── changelog
├── compat
├── control
├── files
├── python-osm-ro/
├── python-osm-ro.postinst.debhelper
├── python-osm-ro.prerm.debhelper
├── python-osm-ro.substvars
└── rules
```

vs.

Snap

```
snap/
└── snapcraft.yaml
```

Anatomy of a snapcraft.yaml

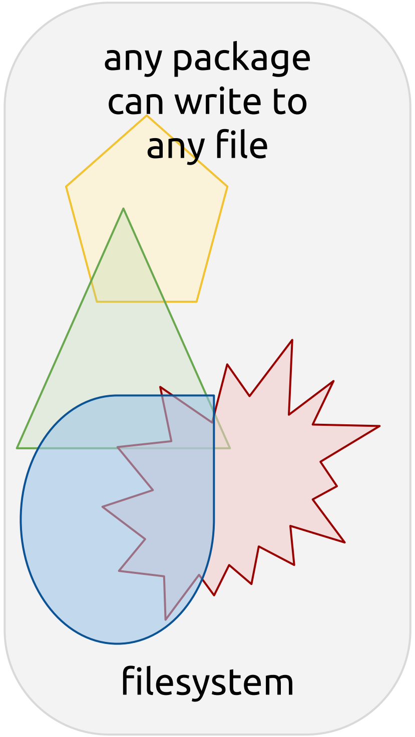
```
name: osmclient         # you probably want to 'snapcraft register <name>'
version: '0.1'          # just for humans, typically '1.2+git' or '1.3.2'
summary: A python client for osm orchestration
description: A python client for osm orchestration
grade: stable           # must be 'stable' to release into candidate/stable channels
confinement: strict     # use 'strict' once you have the right plugs and slots

apps:
  osmclient:
    command: bin/osm

parts:
  osmclient:
    source: .
    plugin: python
    python-version: python2
    stage:
      - -README.md
```

Snap can be classically confined

In classic confinement, snaps behave as a traditionally packaged application. They have full access to the system and can read or write to any file.



Snap can be strictly confined

In strict mode (the default), a snapped application can only write to its own install space and selected areas, including the libraries it installs.

Strict confinement gives you the following readable and/or writable paths:

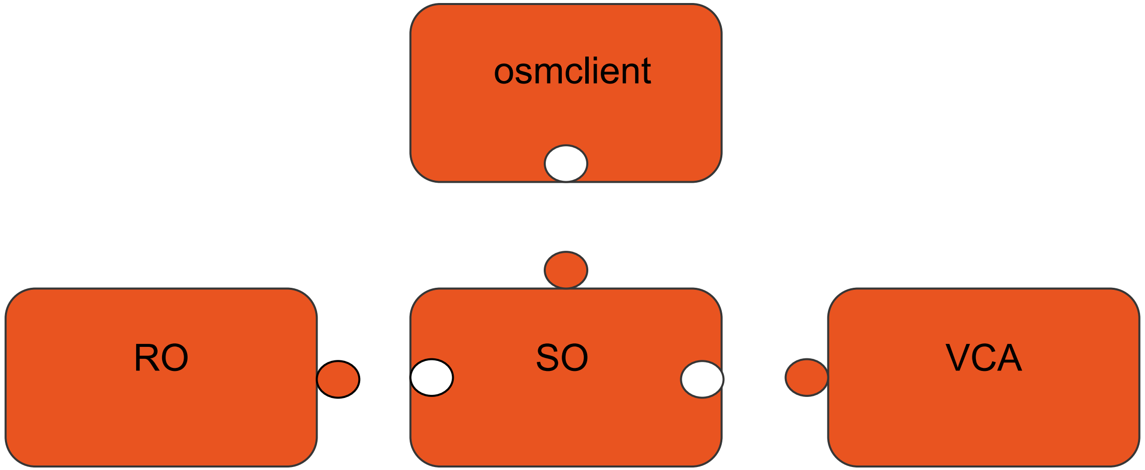
- /snap/<snap>/<revision> (read-only, snap install path)
- /var/snap/<snap>/<revision> (read/write, per-revision data)
- /var/snap/<snap>/common (read/write, common data)
- /home/$USER/snap/<snap>/<revision> (read/write, per-revision user data)
- /home/$USER/snap/<snap>/common (read/write, common user data)

Snaps are secure

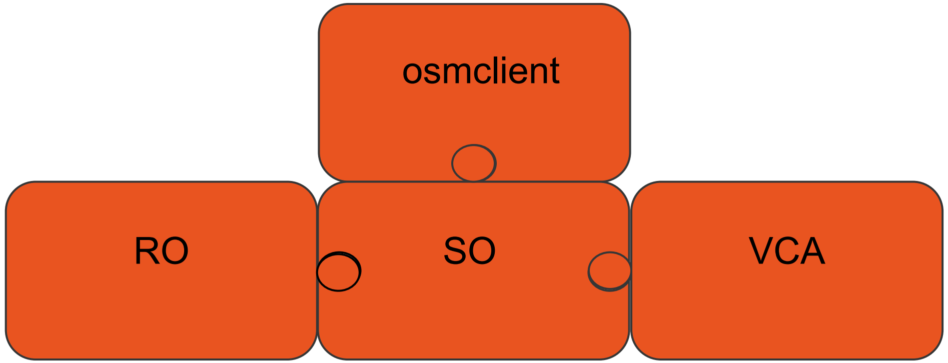
Strictly confined snaps are secured via a sophisticated kernel mechanism so they can only access data they've been explicitly been given access to.

A snap declares a “slot” for data it provides and a “plug” for data it consumes, and an “interface” connects snaps together.

Snapping it together



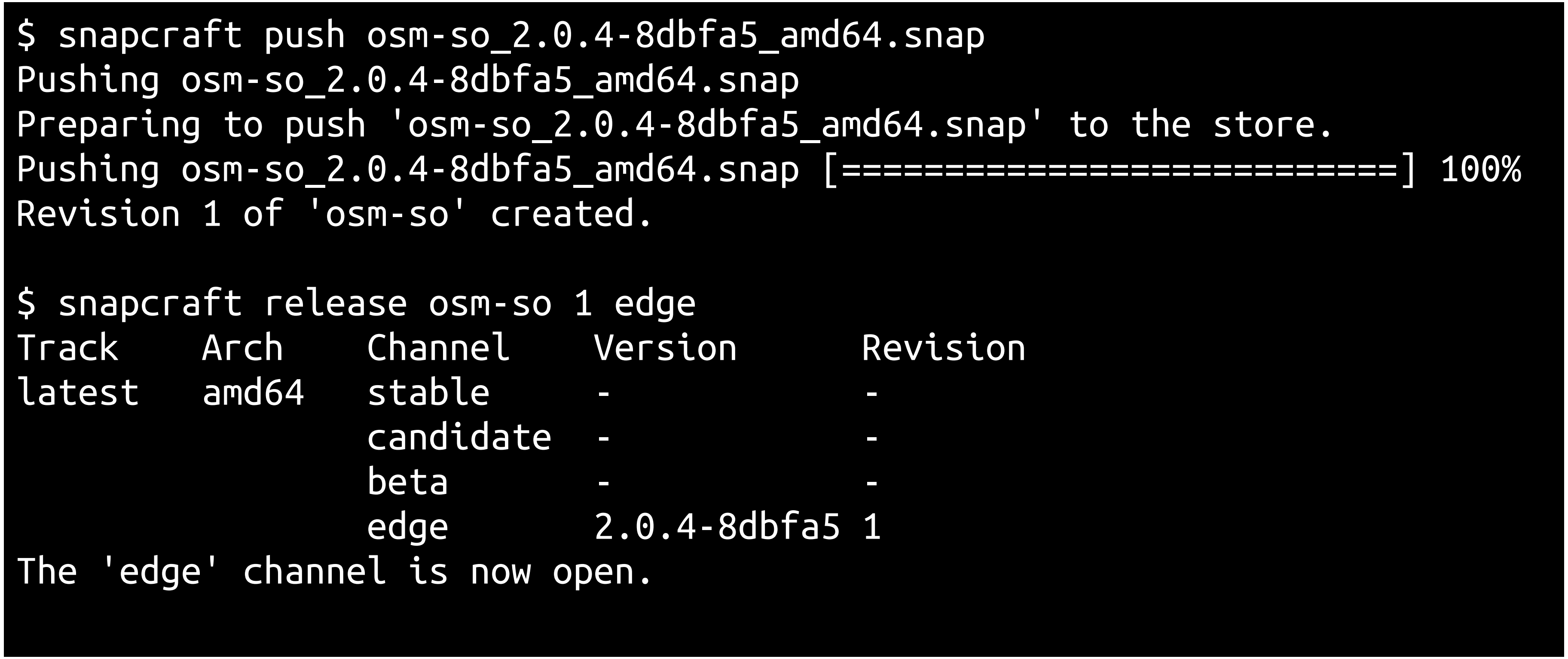
Snap!



Snaps are good for devops

- Creating a fresh environment to test in is faster
- Test commits against stable components
- Easier packaging format reduces time commitment to maintain
- Easier to distribute the binary artifacts

Easy to release new builds



```
$ snapcraft push osm-so_2.0.4-8dbfa5_amd64.snap
Pushing osm-so_2.0.4-8dbfa5_amd64.snap
Preparing to push 'osm-so_2.0.4-8dbfa5_amd64.snap' to the store.
Pushing osm-so_2.0.4-8dbfa5_amd64.snap [===========================] 100%
Revision 1 of 'osm-so' created.

$ snapcraft release osm-so 1 edge
Track    Arch    Channel    Version      Revision
latest   amd64   stable     -            -
                 candidate  -            -
                 beta       -            -
                 edge       2.0.4-8dbfa5 1
The 'edge' channel is now open.
```

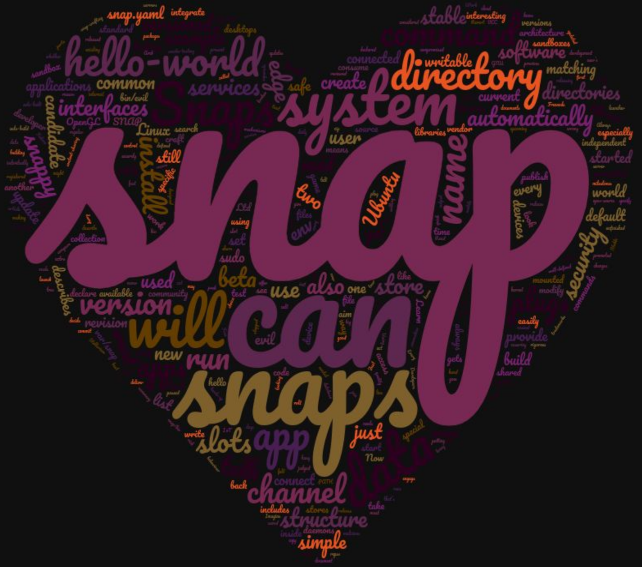
Flashback

Snaps are:

- Self-contained, including dependencies
- Fast to install
- Transactionally updated, automatically
- Versioned
- Distributable via channel
- Easier to build than traditional packaging
- Contained, classically or strictly
- Secure

Open Source MANO

THANK YOU



https://snapcraft.io/