Snappy Ubuntu Core

James Henstridge
james@jamesh.id.au

Timeline

- 2012: Ubuntu ported to Nexus 7
  - Standard Ubuntu desktop
- 2013: Ubuntu Touch project launched
  - Touch oriented UI
  - Read-only image based updates, Click packages
  - Hardware released in 2015
- 2014: Snappy Ubuntu Core
  - Strips UI from Touch to provide a core

Snappy Overview

- Small base image (“Ubuntu Core”)
- Does not replace traditional Ubuntu
- Read-only root file system, allowing for transactional updates
- Applications isolated from base system and each other
  - applications confined with AppArmor
  - stored separate from base image
- Can be augmented with “frameworks” to provide additional functionality to apps

Possible Use Cases

- Cloud servers
- Routers
- Internet of Things devices
- ...

Installation

- Instructions on developer site:
  - https://developer.ubuntu.com/en/snappy/start/
- Images available for:
  - x86 (bare metal, cloud, VM)
  - Beaglebone Black
  - **Raspberry Pi 2**

Partition Layout (R-Pi 2)

mmcblk0p1	64 MB	system-boot
mmcblk0p2	1 GB	system-a
mmcblk0p3	1 GB	system-b
mmcblk0p4	(remainder)	writable

Read-only Root Image

```
$ cat /etc/fstab
# Auto-generated by /init
# DO NOT EDIT THIS FILE BY HAND - YOUR CHANGES WILL BE OVERWRITTEN
# (See writable-paths(5) for details)
/dev/root / rootfs defaults,ro 0 0
/writable/user-data /home none bind 0 0
/writable/system-data/apps /apps none bind 0 0
/writable/system-data/oem /oem none bind 0 0
tmpfs /tmp tmpfs defaults 0 0
tmpfs /mnt tmpfs defaults 0 0
/writable/system-data/var/lib/apps /var/lib/apps none bind 0 0
/writable/system-data/var/lib/cloud /var/lib/cloud none bind 0 0
...
```

Transactional Updates

- System image is read only, so in known state
- Apply binary patch to contents of system-a partition and write to system-b
- Boot with system-b as root.

Applications

- Installed to `/apps/$package/$version`
  - Contents of this directory are under control of package
- System wide writable data in `/var/lib/apps/$package/$version`
- Per user data in `~/apps/$package/$version`
- Commands from package made available on `$PATH`
  - Using wrapper to enforce confinement
- Can install services managed by systemd

Frameworks

- Packaged the same way as applications
- Provide extra functionality for apps
- Can provide security policy fragments for dependent apps

Creating Snappy Packages

- Use Snapcraft
  - https://developer.ubuntu.com/en/snappy/snapcraft/
- Package metadata and build description expressed as YAML
- Supports common build systems (make, autotools, Python pip, Go, etc)
- Does not currently support cross compilation

Simple Python Snapcraft Project

https://github.com/jhenstridge/plug-snappy-example-python

- `snapcraft.yaml`
- `readme.md`
- `icon.png`
- `setup.py`
- `mycat.py`

snapcraft.yaml

```
name: plug-example-python
version: 1
vendor: James Henstridge <james@jamesh.id.au>
summary: Example python package
description: Example python package
icon: icon.png
binaries:
  mycat:
    exec: usr/bin/mycat.py

parts:
  mycat:
    type: python3-project
    source: .
```

mycat.py (trivial Python application)

```
import sys

def main(argv):
    try:
        with open(argv[1], 'r') as fp:
            buf = fp.read(4096)
            while buf:
                sys.stdout.write(buf)
                buf = fp.read(4096)
    except Exception as exc:
        print("Error:", str(exc))
        return 1

if __name__ == "__main__":
    sys.exit(main(sys.argv))
```

Building and Deploying

- Running “snapcraft” downloads dependencies and assembles package
  - Must be done on same arch as target
- Copy `plug-example-python_1_armhf.snap` to target
- Install package:

```
sudo snappy install --allow-untrusted \
    plug-example-python_1_armhf.snap
```

Building for ARM

- qemu: emulated ARM system (slow)
- Ubuntu on another ARM device (e.g. Chromebook with Crouton)
- Run traditional Ubuntu inside container on R-Pi:

```
sudo snappy install lxd
lxc remote add images images.linuxcontainers.org
lxc launch images:ubuntu/vivid/armhf dev
lxc exec dev bash
```

Security

- No access to data owned by other apps:

```
$ plug-example-python.mycat \
    apps/lxd/0.19-1/.config/lxc/client.key
Error: [Errno 2] No such file or directory:
'apps/lxd/0.19-1/.config/lxc/client.key'
```

- Access to hardware must be granted post-install. For example:

```
sudo snappy hw-assign my-webcam-app /dev/video0
```

Services

```
services:
  - name: webserver
    description: "..."
    start: ./path-to-webserver
    caps:
      - networking
      - network-service
```

Resources:

- Snappy developer documentation:
  - https://developer.ubuntu.com/en/snappy/
- Python test example:
  - https://github.com/jhenstridge/plug-snappy-example-python/

Demo