Yocto Project®
Runtime Package Management: When, Why and How
Frank Vasquez, Lunar Energy

Yocto Project Summit, May 2021
About me

• 10+ years doing embedded Linux
• networked audio, sonar and LoRa
• MELP3 co-author
• home electrification
Agenda

• What is runtime package management?
• What are the dangers?
• What are the benefits?
• Adding opkg to an image
• Serving ipk package feeds locally
• Configuring opkg client on board
When
Runtime package management

• Fetch packages from a server
• Install packages on the target
• Selecting a package manager for your distro
• Choices are dnf and opkg
• Partial image updates on a per package basis
Never in production

- Not robust
  - Extract an archive over the filesystem
  - Run scripts before and after extraction
  - Can result in inconsistent image states
- Attack vector
  - OpenWRT code-execution bug
  - HTTPS not enforced
  - downloads.openwrt.org
  - Package size vs. SHA256 hash
Only for development

- Quick experimentation
- Build and serve packages locally
- Install packages on demand
- Update packages on demand
Velocity

• No waiting for image builds
• Try before you buy
• No full image updates and reboots
• Rev packages not images
• No scping of binaries
Familiarity

- BeagleBone, Jetson, Raspberry Pi
- Headless or HDMI
- Debian or Ubuntu
- apt
- Docker
- conda-forge
How
Adding opkg to an image

- distro layer: meta-mackerel/conf/distro/mackerel.conf
  ```
  PACKAGE_CLASSES ?= "package_ipk"
  ```

- image build: build-rpi/conf/local.conf
  ```
  #PACKAGE_CLASSES ?= "package_rpm"
  EXTRA_IMAGE_FEATURES ?= "debug-tweaks ssh-server-openssh package-management"
  DISTRO = "mackerel"
  ```

- opkg client now on board
  ```
  root@raspberrypi4-64:~# which opkg
  /usr/bin/opkg
  ```
Adding packages on demand

• **Build curl package**
  
  $ source poky/oe-init-build-env build-rpi
  $ bitbake curl

• **Populate package index**
  
  $ bitbake package-index
Package feeds

• Navigate to where the package installers are
  
  $ cd build-rpi/tmp-glibc/deploy/ipk

• Package server expects four directories
  
  $ ls
  aarch64 all Packages raspberrypi4_64

• So-called *architecture* and *machine* directories
Serving packages locally

- Start HTTP package server

```bash
$ pwd
/home/frank/build-rpi/tmp-glibc/deploy/ipk

$ ip a
2: wlp59s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
   group default qlen 1000
   link/ether f4:8c:50:80:ed:9b brd ff:ff:ff:ff:ff:ff
   inet 192.168.1.69/24 brd 192.168.1.255 scope global dynamic noprefixroute
   wlp59s0

$ sudo python3 -m http.server --bind 192.168.1.69 80
[sudo] password for frank:
Serving HTTP on 192.168.1.69 port 80 (http://192.168.1.69:80/) ...
```
Configuring opkg client

- SSH into target

```
$ ssh root@raspberrypi4-64.local
```

- Edit /etc/opkg/opkg.conf

```
src/gz all http://192.168.1.69/all
src/gz aarch64 http://192.168.1.69/aarch64
src/gz raspberrypi4_64 http://192.168.1.69/raspberrypi4_64

dest root /
option lists_dir /var/lib/opkg/lists
```
Refreshing available package list

root@raspberry pi4-64:~# opkg update
Downloading http://192.168.1.69/all/Packages.gz.
Updated source 'all'.
Downloading http://192.168.1.69/aarch64/Packages.gz.
Updated source 'aarch64'.
Downloading http://192.168.1.69/raspberry pi4_64/Packages.gz.
Updated source 'raspberry pi4_64'.

Installing a package

```
root@raspberrypi4-64:~# opkg install curl
Installing libcurl4 (7.69.1) on root
Downloading http://192.168.1.69/aarch64/libcurl4_7.69.1-r0_aarch64.ipk.
Installing curl (7.69.1) on root
Downloading http://192.168.1.69/aarch64/curl_7.69.1-r0_aarch64.ipk.
Configuring libcurl4.
Configuring curl.

root@raspberrypi4-64:~# curl
curl: try 'curl --help' for more information

root@raspberrypi4-64:~# which curl
/usr/bin/curl
```
Updating packages

• Check for package updates

```bash
root@raspberrypi4-64:~# opkg list-upgradable
```

• Apply them

```bash
root@raspberrypi4-64:~# opkg upgrade
```
Connect

MELP3 available at amazon

https://packt.live/3tiDDrA

fvasquez
@st8l3ss
st8l3ss#9518