

# Linux Distro Tools for Building Container Images

Nisha Kumar (@nishakmr)

Joshua Lock (@hi\_joshuagl)

Open Source Software Engineers / OSTC

@vmwopensource

November 2019

# Agenda

## Motivations and context

Motivations for exploring other container image build tools

Context and level setting

## Investigation

Linux distros and tools

Experiments and analysis

## Summary

Takeaways

Actionable learning

# Motivations and context

Why are we looking at distro tools?

# Motivations



Compliance



Security



Discipline

# Level setting

Containers ~= packaging format



# Desirable Properties of a Container Image

...or package format

## Repeatable

At any given time, we can reliably create an equivalent image from source

## Identifiable

We can reason about their contents, their license compliance and any known vulnerabilities

## Recent

Contents are not old and vulnerable

# Common practices which violate the desirable properties

Performing non-deterministic operations

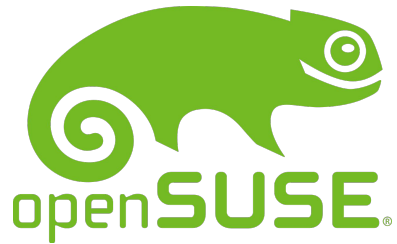
Inserting untraceable files into the filesystem

Encoding build-time dependent state in the image

Using an old base OS

# These problems have been solved before

Why reinvent the wheel?





# Tool Usability Continuum

...or DevOps

Development

Deployment

Easy to use



Comprehensive

Available  
Content



Secure  
Content

Fast



Reliable

# Investigation

Using distribution tools to build container images

# Linux distributions and tools

## Choosing is hard

### Debian and DebOS

- Debian – the universal OS
- DebOS – Debian image generator created for Embedded Linux use-cases

### Buildroot

- Embedded Linux image creator

### Yocto Project

- Embedded Linux distribution builder

### Guix SD

- Functional distro built on Guix package manager, inspired by Nix
- Scheme extensions provide DSLs for packaging and configuration

# Distribution tool showdown

Trying to use the tools and compare the outputs

Define experiments to understand how useful these tools are for container image creators

Compare

- Ease of use
- Minimal reliance on external infrastructure
- Output image size
- Engineering effort
- Quality/presence of SBoM
- Ease of update







This Photo by Unknown Author is licensed under [CC BY-SA-NC](#)

# Showdown: base OS for Go applications





## Base OS with Go toolchain

Goal: create a container image with Go toolchain

DebOS: 6/10

- Ease of use 
- Image size 
- Engineering effort 
- SBoM 

Buildroot: 5/10





- Ease of use 
- Image size 
- Engineering effort 
- SBoM 

# Showdown: base OS for Go applications





## Base OS with Go toolchain

Goal: create a container image with Go toolchain

Yocto Project: 4/10

- Ease of use 
- Image size 
- Engineering effort 
- SBoM 

Guix: 3/10

- Ease of use 
- Image size 
- Engineering effort 
- SBoM 

# Summary

Closing remarks

# Takeaways

This is only a partially solved problem

Tools go halfway, reasonable companion but not completely there:

- Distros have traditionally focused on different deployment targets (you'll see references to “boards”)
- They are improving their tooling for the cloud (a “cloud” board?)
- Compelling reasons to help them (test, document, code, etc – Open Source)



# Actionable learning

What can we do today for my Dockerfile built containers?

Be aware of the issues

Google Cloud's "[Best practices for building containers](#)"

Introspect your containers:

Dive - to dig into the layers of a container

- <https://github.com/wagoodman/dive>

Container-diff - to understand what changed between container versions/additions

- <https://github.com/GoogleContainerTools/container-diff>

Tern - to understand the license compliance obligations of your image

- Watch out for pinned Dockerfile feature ([github.com/vmware/tern/issues/454](https://github.com/vmware/tern/issues/454))
- <https://github.com/vmware/tern>

# Resources

DebOS: <https://github.com/go-debos/debos>

Debian packaging: <https://wiki.debian.org/Packaging/Intro?action=show&redirect=IntroDebianPackaging>

Buildroot user manual: <https://buildroot.org/downloads/manual/manual.html>

Guix user manual: [https://guix.gnu.org/manual/en/html\\_node/](https://guix.gnu.org/manual/en/html_node/)

Yocto Project Overview and Concepts manual: <https://www.yoctoproject.org/docs/3.0/overview-manual/overview-manual.html>

Yocto Project Development Tasks manual: <https://www.yoctoproject.org/docs/3.0/dev-manual/dev-manual.html>

Yocto Project Reference manual: <https://www.yoctoproject.org/docs/3.0/ref-manual/ref-manual.html>

Yocto Project Mega Manual: <https://www.yoctoproject.org/docs/3.0/mega-manual/mega-manual.html>



# Thank You