datto

Building Linux images

Markus Rex
MAY 2017

Agenda

- Use cases for image building
- Why automation?
- Background in process
- Tools overview
- Examples

Use cases

- OS images are needed everywhere, old to new:
  - Virtual appliances
  - Docker containers
  - Physical appliances (like Backup devices or network routers or Alexa)
  - OpenStack
  - Public Cloud images
  - Vagrant … and plenty more!

⇒ All need **reliable**, **repeatable** and **automated** creation of OS images for a wide variety of platforms

Why automation

One word: QA

Background

Stages of Image creation

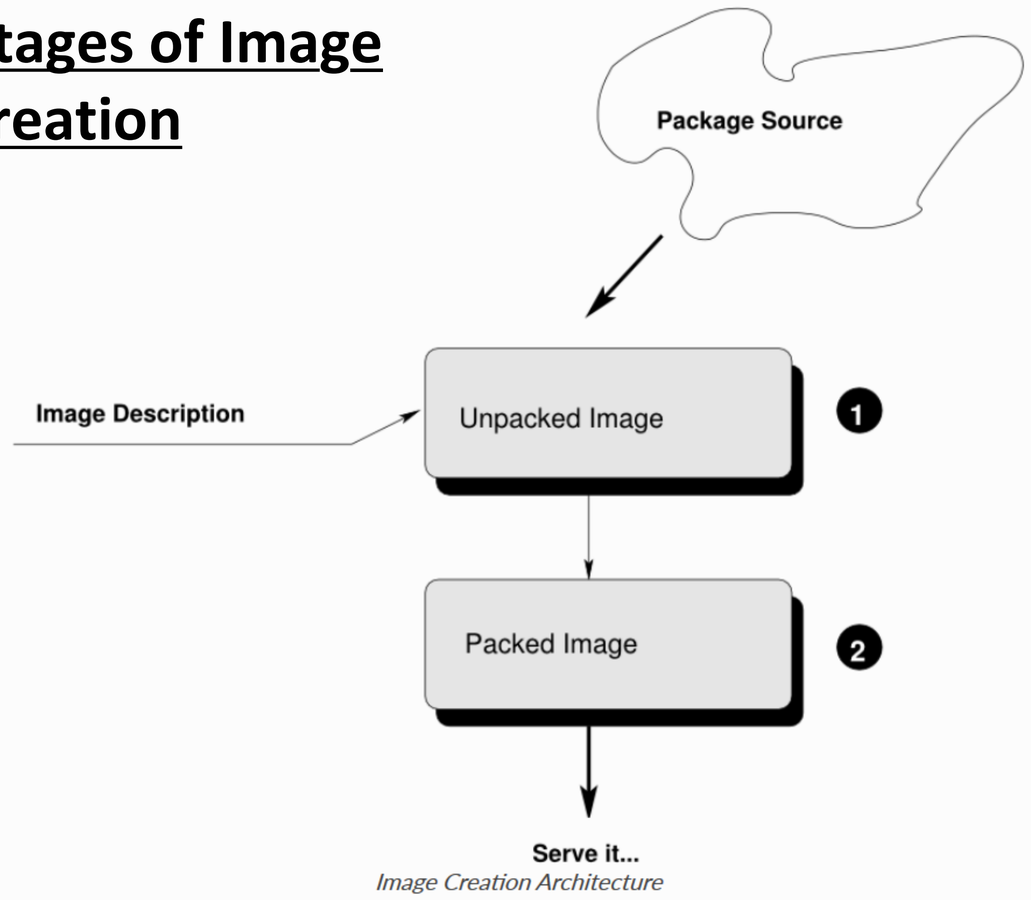

Image Creation Architecture

- ❶ Before assembly, modification *might* be needed, definitively after assembly
- ❷ Here the start/boot setup happens, plus stripping, pre-configuring, deployment automation etc.
- Rarely any defaults fit the specific need, customization needs are HUGE
- There are many ways to make an OS image start or boot, a nightmare!

Tools

Many different choices – most at v0.9 level and stale. This is dry stuff and people leave…

Good options are

- Packer (HashiCorp) – cloud-centric, tight post-build mgmt-tool integr.
- KIWI (SUSE) – XML config-file, OBS integration, super-granular
- Virt-builder (RH) – all CLI, fewer platforms, good with ISOs
- Diskimage-builder (OpenStack) – more cloud focus, less automation

Examples

Virt-builder options

```
virt-builder os-version
   [-o|--output DISKIMAGE] [--size SIZE] [--format raw|qcow2]
   [--arch ARCHITECTURE] [--attach ISOFILE]
   [--append-line FILE:LINE] [--chmod PERMISSIONS:FILE]
   [--commands-from-file FILENAME] [--copy SOURCE:DEST]
   [--copy-in LOCALPATH:REMOTEDIR] [--delete PATH] [--edit FILE:EXPR]
   [--firstboot SCRIPT] [--firstboot-command 'CMD+ARGS']
   [--firstboot-install PKG,PKG..] [--hostname HOSTNAME]
   [--install PKG,PKG..] [--link TARGET:LINK[:LINK..]] [--mkdir DIR]
   [--move SOURCE:DEST] [--password USER:SELECTOR]
   [--root-password SELECTOR] [--run SCRIPT]
   [--run-command 'CMD+ARGS'] [--scrub FILE] [--sm-attach SELECTOR]
   [--sm-register] [--sm-remove] [--sm-unregister]
   [--ssh-inject USER[:SELECTOR]] [--truncate FILE]
   [--truncate-recursive PATH] [--timezone TIMEZONE] [--touch FILE]
   [--uninstall PKG,PKG..] [--update] [--upload FILE:DEST]
   [--write FILE:CONTENT] [--no-logfile]
   [--password-crypto md5|sha256|sha512] [--selinux-relabel]
   [--sm-credentials SELECTOR]


virt-builder -l|--list [--long] [--list-format short|long|json] [os-version]

virt-builder --notes os-version

virt-builder --print-cache

virt-builder --cache-all-templates

virt-builder --delete-cache

virt-builder --get-kernel DISKIMAGE
   [--format raw|qcow2] [--output OUTPUTDIR]
```

KIWI config file

```
14          <file name="/lib/i686/nosegneg"/>
15          <file name="/lib/kbd"/>
16      </strip>
17      <preferences>
18          <version>1.2.0</version>
19          <packagemanager>yum</packagemanager>
20          <bootsplash-theme>charge</bootsplash-theme>
21          <locale>en_US</locale>
22          <keytable>us.map.gz</keytable>
23          <timezone>UTC</timezone>
24          <hwclock>utc</hwclock>
25          <rpm-check-signatures>false</rpm-check-signatures>
26          <type image="vmx" primary="true" boot="vmxboot/rhel-07.0" filesystem="ext3" kernelcmdline="rhgb" bootloader="grub2"/>
27          <type image="oem" boot="oemboot/rhel-07.0" filesystem="ext3" installiso="true" bootloader="grub2" initrd_system="dracut">
28              <oemconfig>
29                  <oem-systemsize>2048</oem-systemsize>
30                  <oem-swap>true</oem-swap>
31                  <oem-swapsize>200</oem-swapsize>
32              </oemconfig>
33          </type>
34      </preferences>
35      <users>
36          <user password="$1$wYJUgpM5$RXMMeASDc035eX.NbYWF10" home="/root" name="root" groups="root"/>
37      </users>
38      <repository type="rpm-md" alias="centos">
39          <source path="http://bo.mirror.garr.it/1/slc/centos/7.2/os/x86_64"/>
```

Examples

KIWI config file part 2

msrex on linux: /home/msrex

```
    <repository type="rpm-md" priority="1" alias="kiwi-common-boot">
        <source path="obs://Virtualization:/Appliances:/CommonBoot/CentOS_7"/>
    </repository>
    <packages type="image">
        <namedCollection name="core"/>
        <namedCollection name="console-internet"/>
        <package name="grub2"/>
        <!-- do not delete package "epel-release" or epel stops working! -->
        <package name="epel-release"/>
        <package name="bash-completion"/>
        <package name="deltarpm"/>
        <package name="kernel"/>
        <package name="grubby"/>
        <package name="yum-plugin-priorities"/>
        <package name="plymouth-theme-charge" bootinclude="true"/>
        <!-- test package pulling an epel dependency -->
<!--        <package name="datto-novnc"/> -->
    </packages>
    <packages type="image" profiles="efi">
        <package name="grub2-efi"/>
        <package name="grub2-efi-modules"/>
    </packages>
    <packages type="image" profiles="uefi">
        <package name="grub2-efi"/>
        <package name="grub2-efi-modules"/>
                                                    136,9         84%
```

Details

- Build times vary – 3-15 minutes for minimal image, depending on hardware (disk!) and tool
- All build tools spit out humongous log files (~100kb per build)
- When building cross-platform a million things can go wrong if you are not careful
- Good image tools leverage OS-vendor provided profiles and setup tools
- Not all tools allow **reliably repeatable** builds

Yes, it is dry ☺

Questions?