



WHAT KEEPS CONTAINER MAINTAINERS AWAKE

Honza Horak
Petr Kubat
Red Hat Platform Engineering
January 2019

Who is this container maintainer

Who is this container maintainer

Click to add subtitle

RPM Package maintainer

- Takes source from upstream
- Maintains RPM SPEC
- Drives productization
- Fixes bugs

Who is this container maintainer

Click to add subtitle

~~RPM Package~~ Container image maintainer

- ~~Takes source from upstream~~
- Maintains ~~RPM SPEC~~ Dockerfile and other scripts
- Drives productization
- Fixes bugs
- Tips how to write Dockerfile:
 - <http://docs.projectatomic.io/container-best-practices/>

What we do

Click to add subtitle

A team that develops and maintains several container images

- PostgreSQL, MariaDB, Python, Ruby, PHP, Httpd, Nginx, ...
- OpenShift is the main target platform
- RHEL, CentOS, Fedora
- http://github.com/sclorg/*-container

Challenges we meet

Challenges we meet

Click to add subtitle

No upstream source for the container image

Duplication of the code for different platforms and streams

With more streams: dozens of container images for each platform

Container is a static bundle

- For each CVE or bug fix or behaviour change it needs to be rebuilt
- ...and tested
- Inter-project dependencies

Not every SME is a container master

How we solve the challenges

Container is a bundle vs. one size doesn't fit all

Users have specific use cases

Flexible container is when:

- Tweaking to a specific use case is simple
- Changing the configuration by k8s templates
- New layer just adds new files

Source-to-image strategy in OpenShift

- Configuration of a service becomes an app from s2i perspective

A nice write-up by Eliska:

<https://developers.redhat.com/blog/2017/11/29/flexible-images-using-s2i-image-configuration/>

How we solve the challenges


No upstream source for the container image











Upstream sources under <http://github.com/sclorg>

Single repo for more streams and platforms

- Stream examples: MariaDB 10.2, MariaDB 10.3, ...

CI testing for each PR (ci.centos.org + internal Jenkins)

 **All checks have passed** [Hide all checks](#)
6 successful checks

  centos7 — Build finished.	Details
  centos7 - openshift — Build finished.	Details
  diff — Generated diff.	Details
  fedora — Build finished.	Details
  rhel7 — Build finished.	Details

How we solve the challenges

Duplication of the code for different platforms and streams

Symlinks approach: e.g.

<https://github.com/sclorg/mariadb-container>

- Only README.md, and Dockerfile* are duplicated
- All the differences defined by ENV variables

```
├─ 10.2
|   ├── Dockerfile
|   ├── Dockerfile.rhel7
|   ├── Dockerfile.fedora
|   ├── root
|   │   └── usr
|   ├── root-common -> ../root
|   ├── s2i-common -> ../s2i
|   └── test -> ../test
```

How we solve the challenges

Deduplication: ENV in Dockerfile and a branch in scripts

```
...  
ENV MYSQL_VERSION=10.2  
...
```

```
    if [ "$MYSQL_VERSION" \> "10.0" ] ; then  
mysql $mysql_flags <<EOSQL  
    DROP USER IF EXISTS 'root'@'%';  
    FLUSH PRIVILEGES;  
EOSQL  
    else  
        ...
```

How we solve the challenges

Duplication of the code for different platforms and streams

Dist-gen: e.g.

<https://github.com/sclorg/postgresql-container>

- Requires distgen tool:
<https://github.com/devexp-db/distgen>
- Jinja templates
- No IFs in the code
- Every file only once

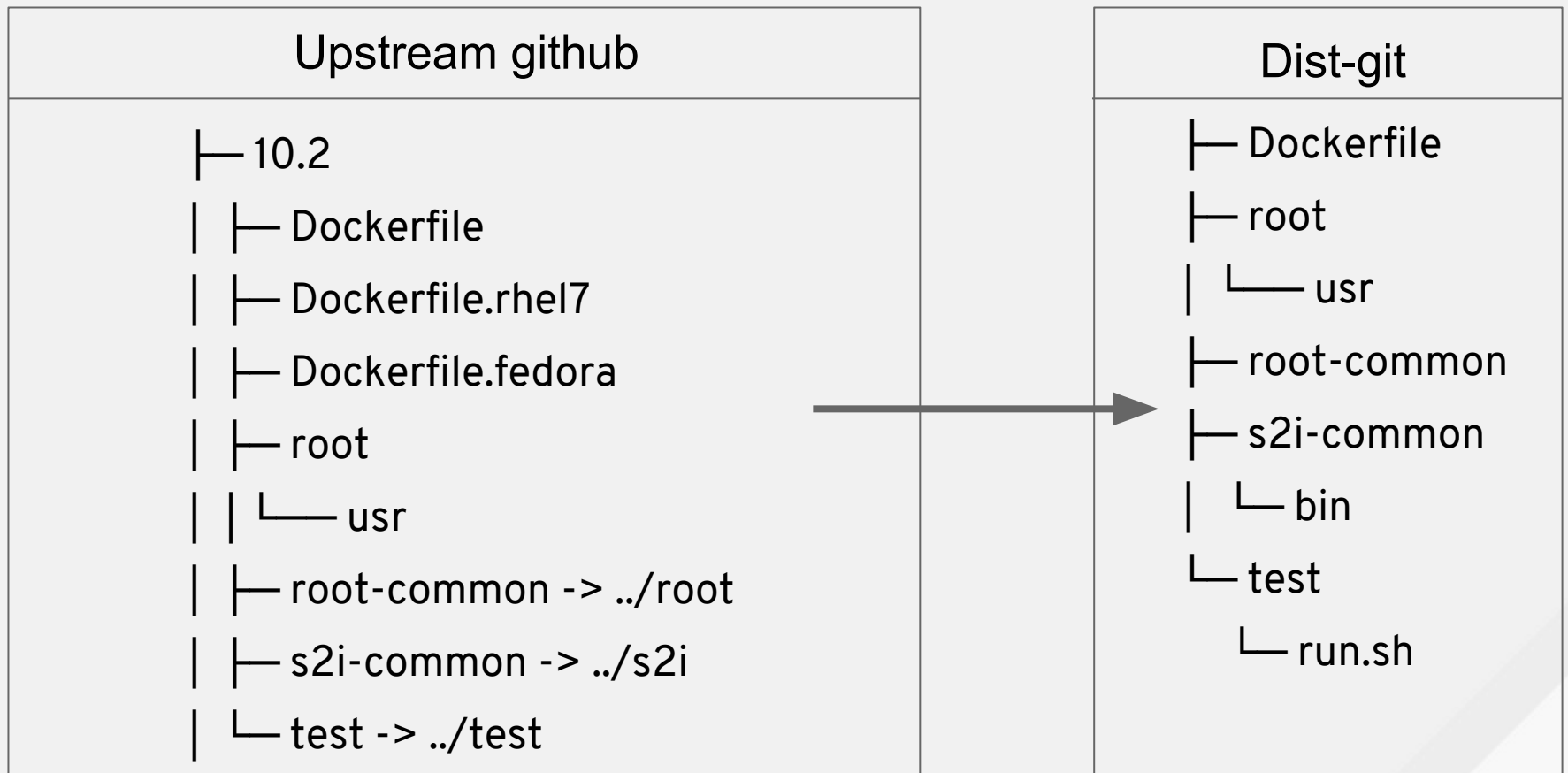
├ common
├ LICENSE
├ Makefile
├ manifest.sh
├ specs
├ src
└ test

... 10.2?

Productization in scale

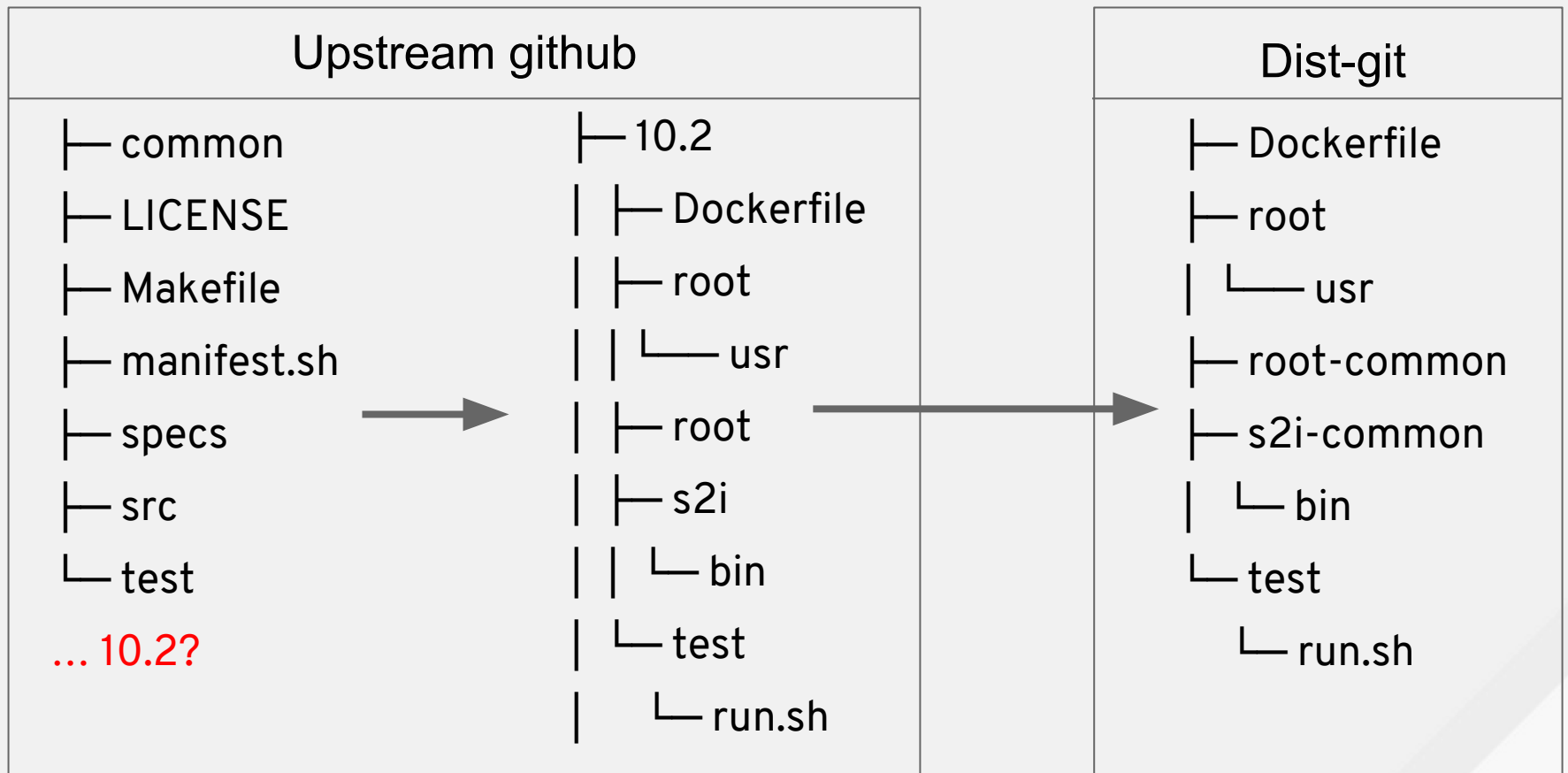
Upstream to downstream

Using symlinks to avoid duplication



Upstream to downstream

Using Distgen to avoid duplication



Productization in scale

Keeping dozens of images for each platform fresh and updated

RHEL:

- 38 images in rhel7
- No code changes for CVE rebuilds
- Source code changes when rebuilding for new base image
 - ~every 6 weeks

Fedora:

- ~15 images per Fedora release
- ~~Rebuild every 2 weeks~~ (not currently)

Automated and reliable testing is a must

Productization in scale

Automation within infrastructure

Freshmaker

- No changes to sources
- Refresh content that delivered by RPM

Chain auto-rebuilds in OSBS

- Allows to pick git HEAD, so latest changes in git may be taken

Github to dist-git

- Requires transformation
- Cyborg colleagues (Betka, ...)

CWT

<https://github.com/sclorg/container-workflow-tool>

Bulk actions

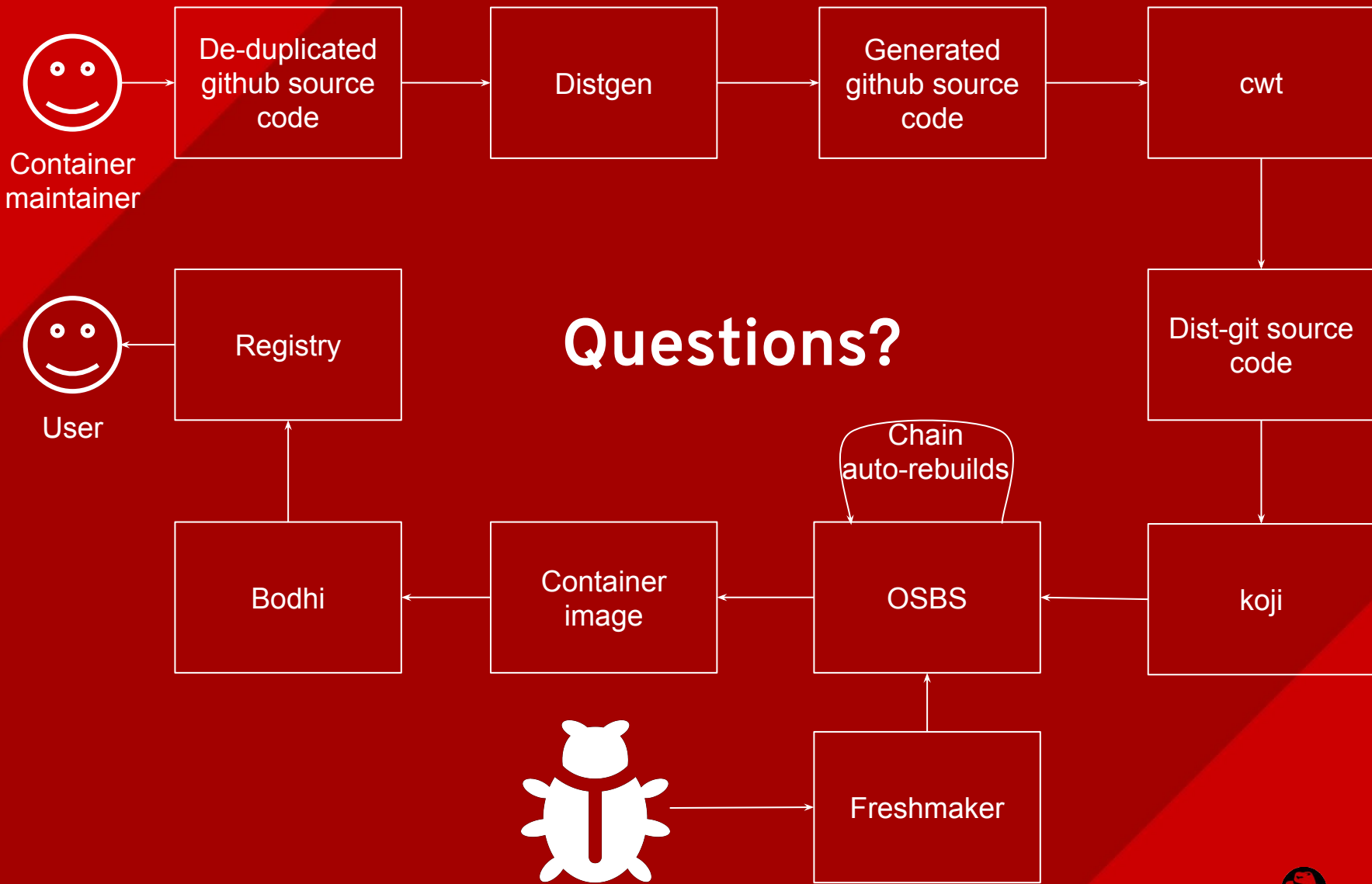
Git content synchronisation

- Incl. necessary layout transformations

Handling image builds

- Latest build status
- Triggering new builds

Future Feature: Bodhi Updates



Questions?