PostgreSQL and variations of linux containers

Honza Horak <hhorak@redhat.com>
P2D2, Praha, February 2017

The goal today

- Show how we think about containers
- Convince you that you shouldn’t ignore them
- Introduce technologies that we care about in Red Hat
- Get feedback

Honza Horak

Brno, Czech Republic

- Red Hat, Platform Engineering
- Databases, Python, Ruby
- RHEL, Fedora, CentOS
- Software Collections

Honza Horak

Brno, Czech Republic

- Red Hat, Platform Engineering
- Databases, Python, Ruby
- RHEL, Fedora, CentOS
- Software Collections
- **AND CONTAINERS**

What this talk includes

1. Containers basics
2. PostgreSQL Docker container
3. Standalone containers
4. Tools containers
5. GUI apps in containers
6. OS containers
7. Ansible Containers
8. OCI

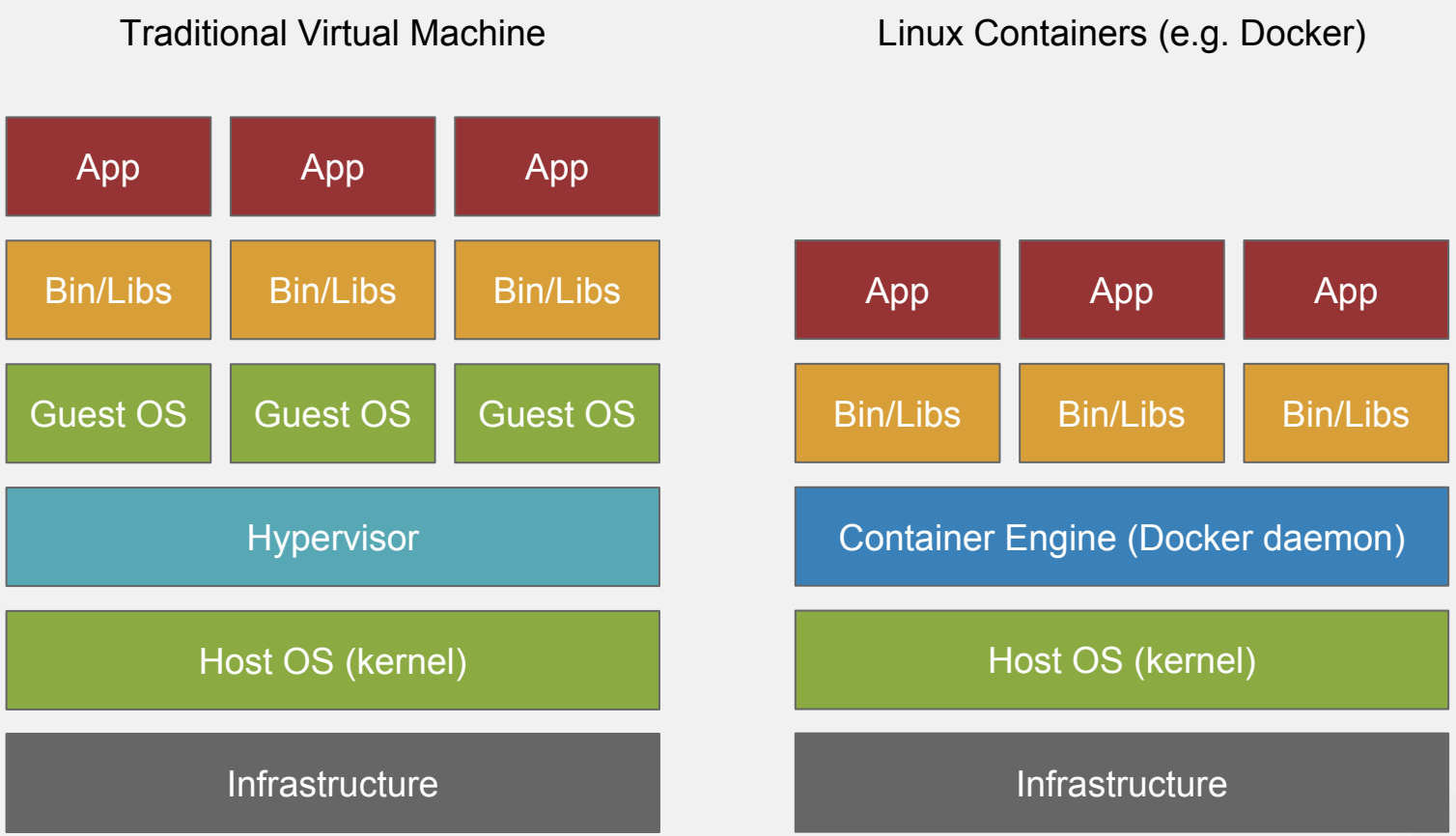
1. CONTAINERS BASICS


ВАТКИНСЦЕНТРАНС
ТЕЛ. 400-10-11

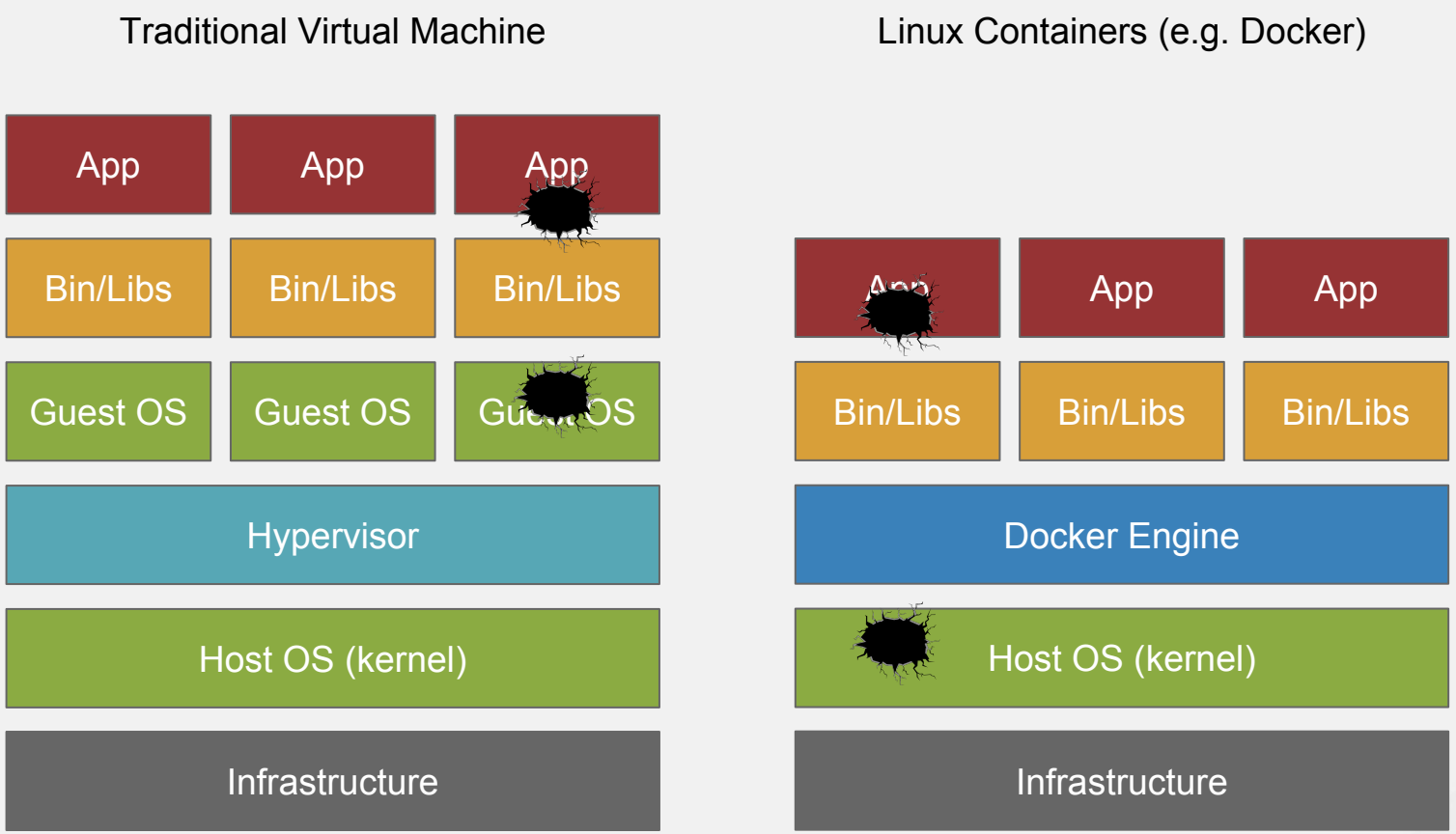
Containers and images

- Container aka Docker
  - Process isolated by Linux kernel features
    - cgroups, namespaces, SELinux, ...
  - Virtualization technology
- Image
  - Static container used to run containers
- Container is an instance of a (container) image

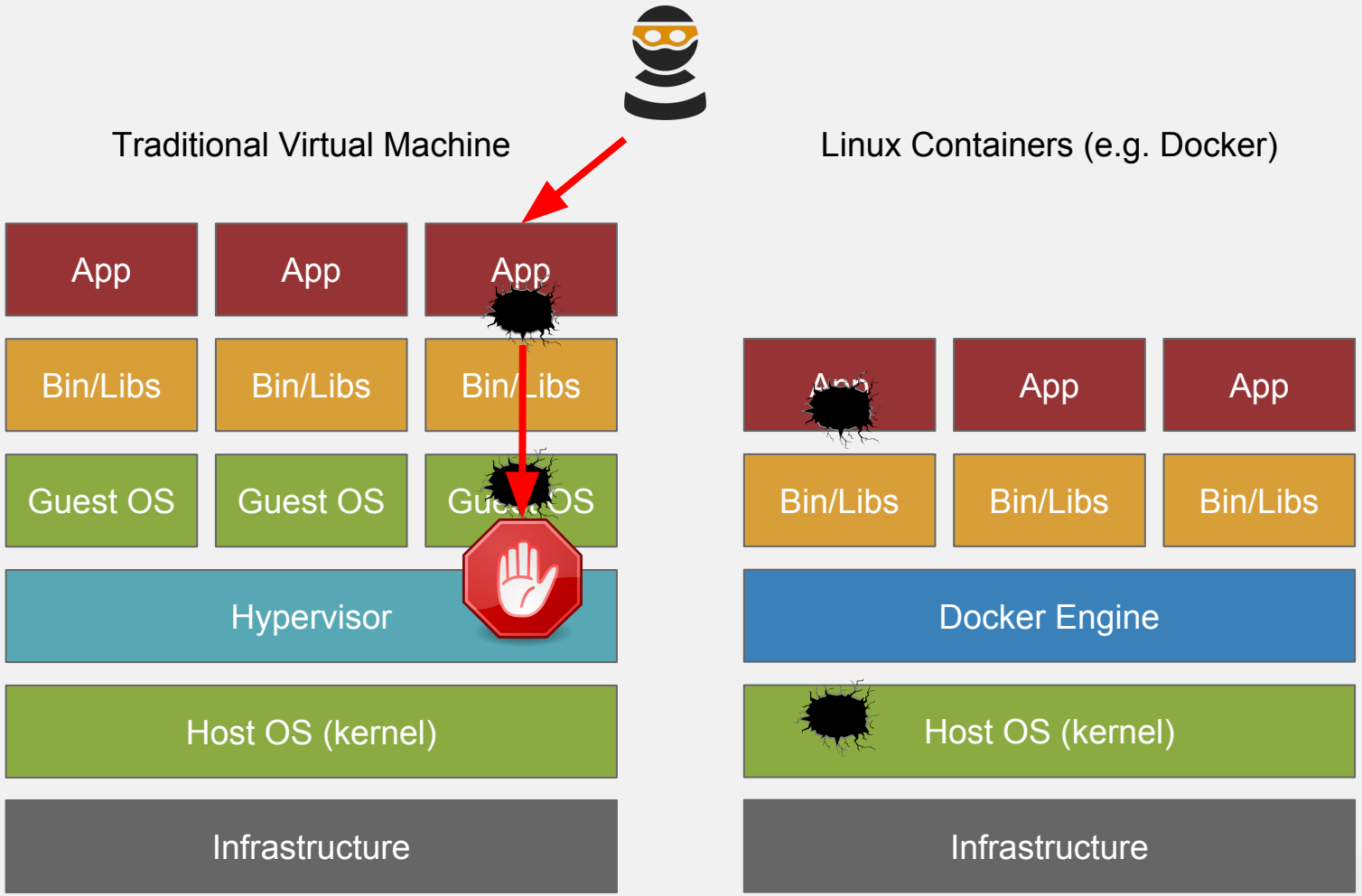
Container is not a virtual machine



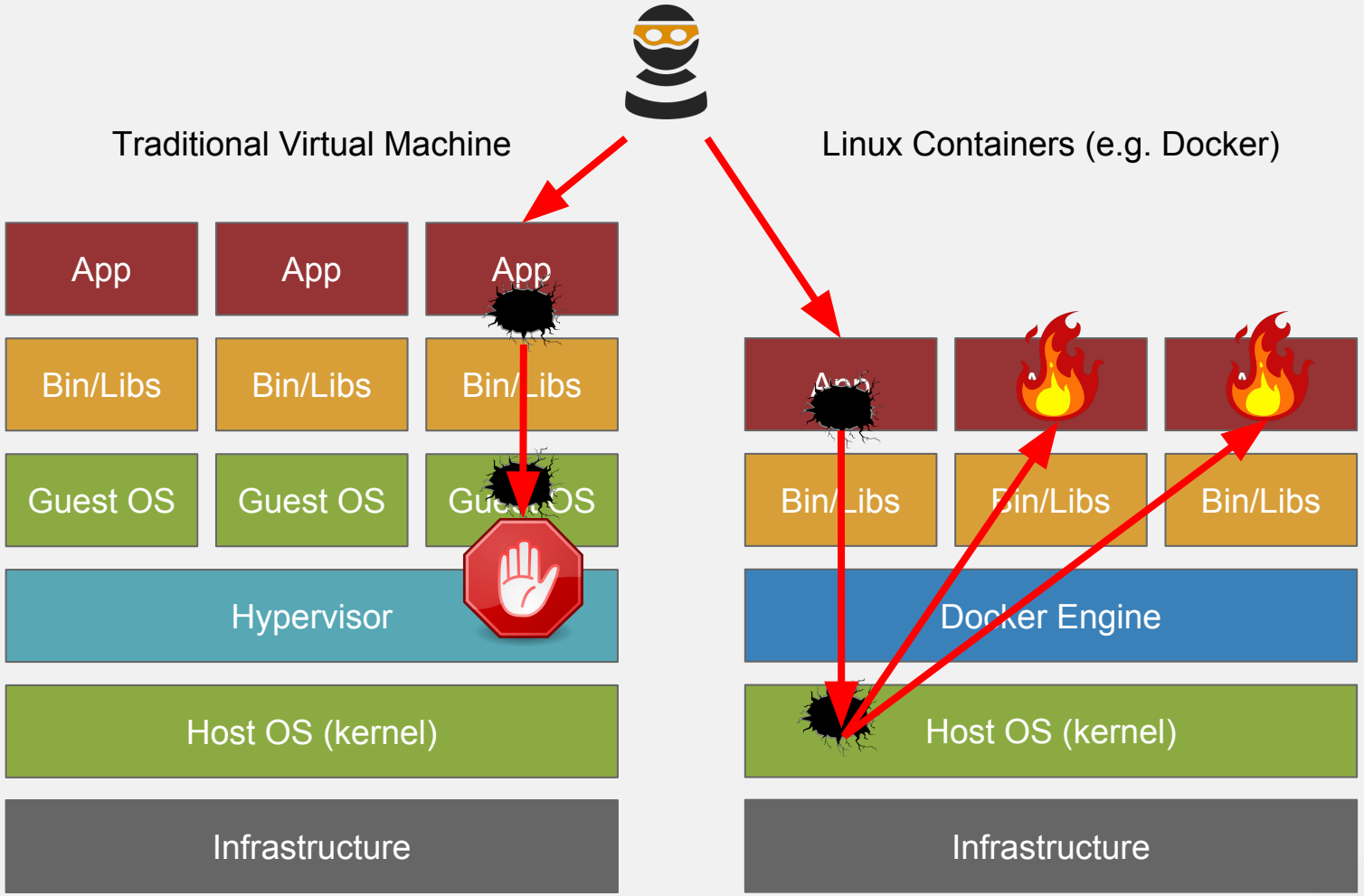
Container is not a virtual machine



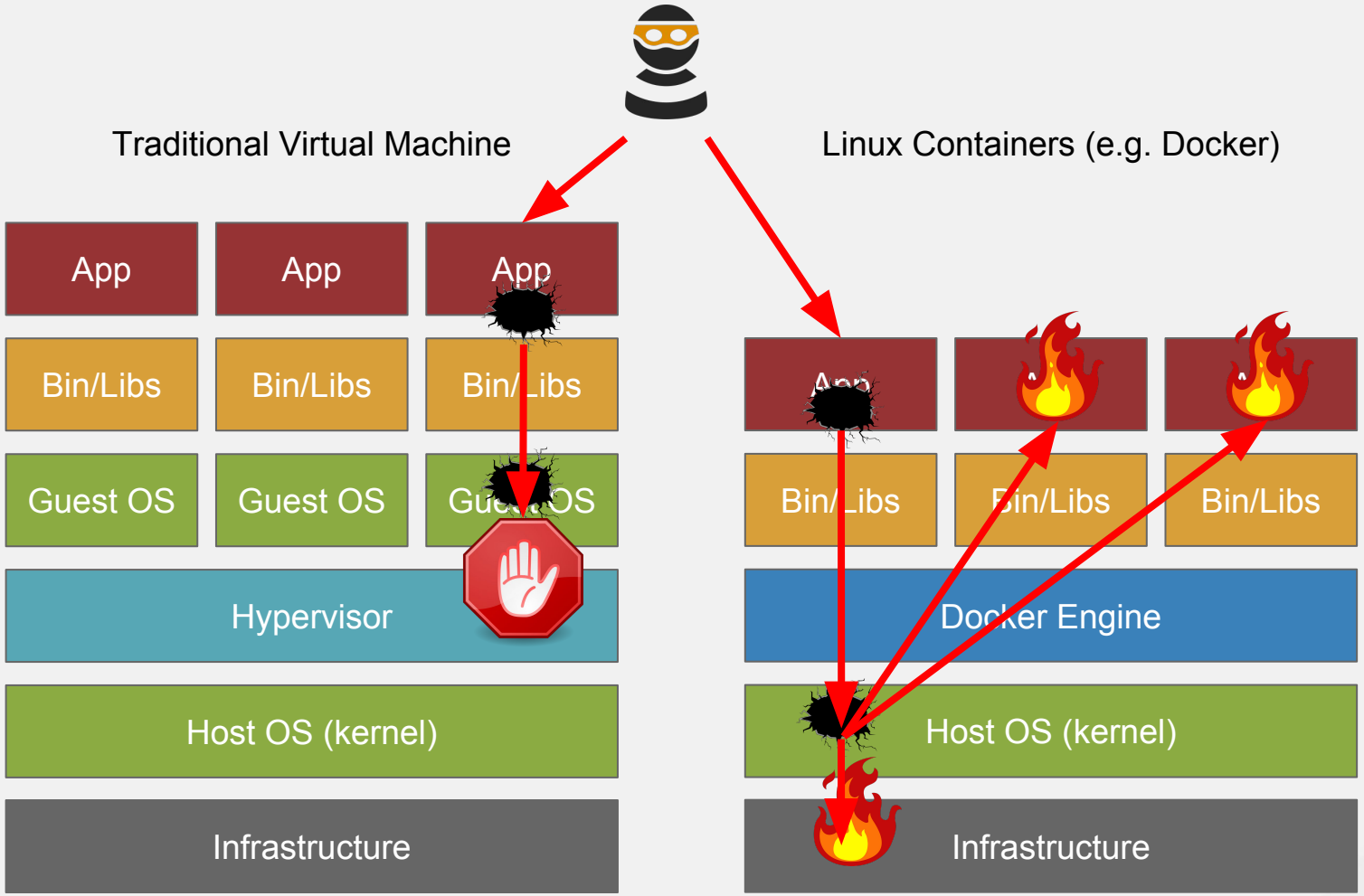
Container is not a virtual machine



Container is not a virtual machine



Container is not a virtual machine

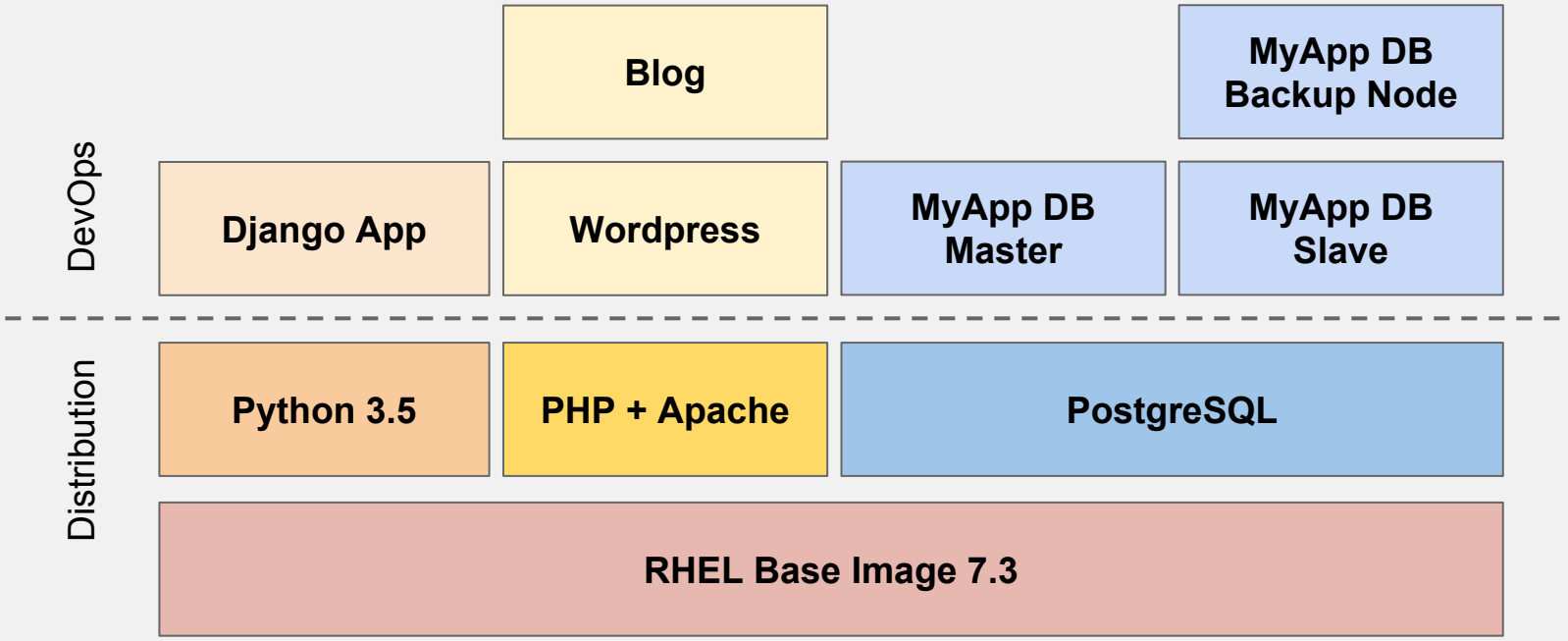


Use only content you trust.

Avoid root inside container
or use user namespaces.

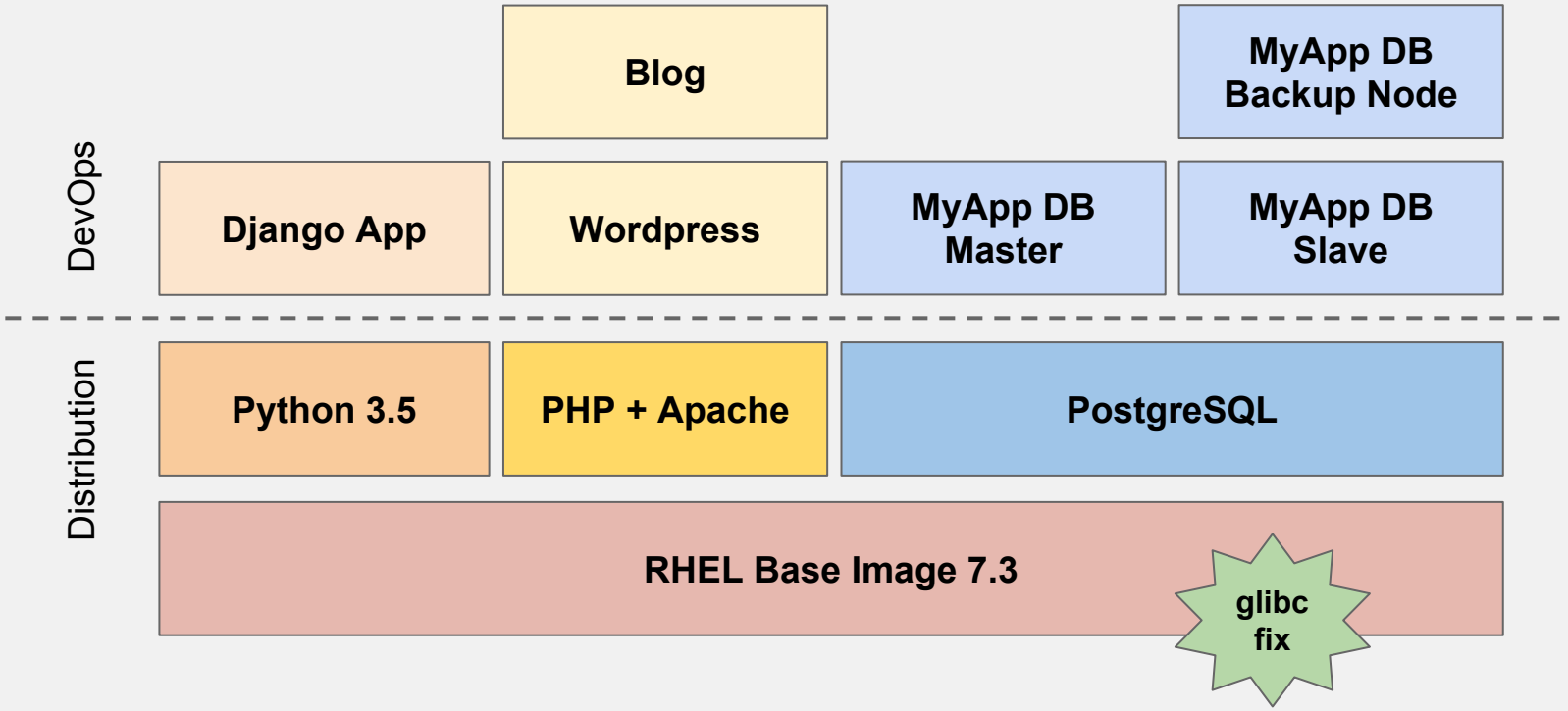
Docker and Layers

What is gonna be part of distribution and what is beyond.



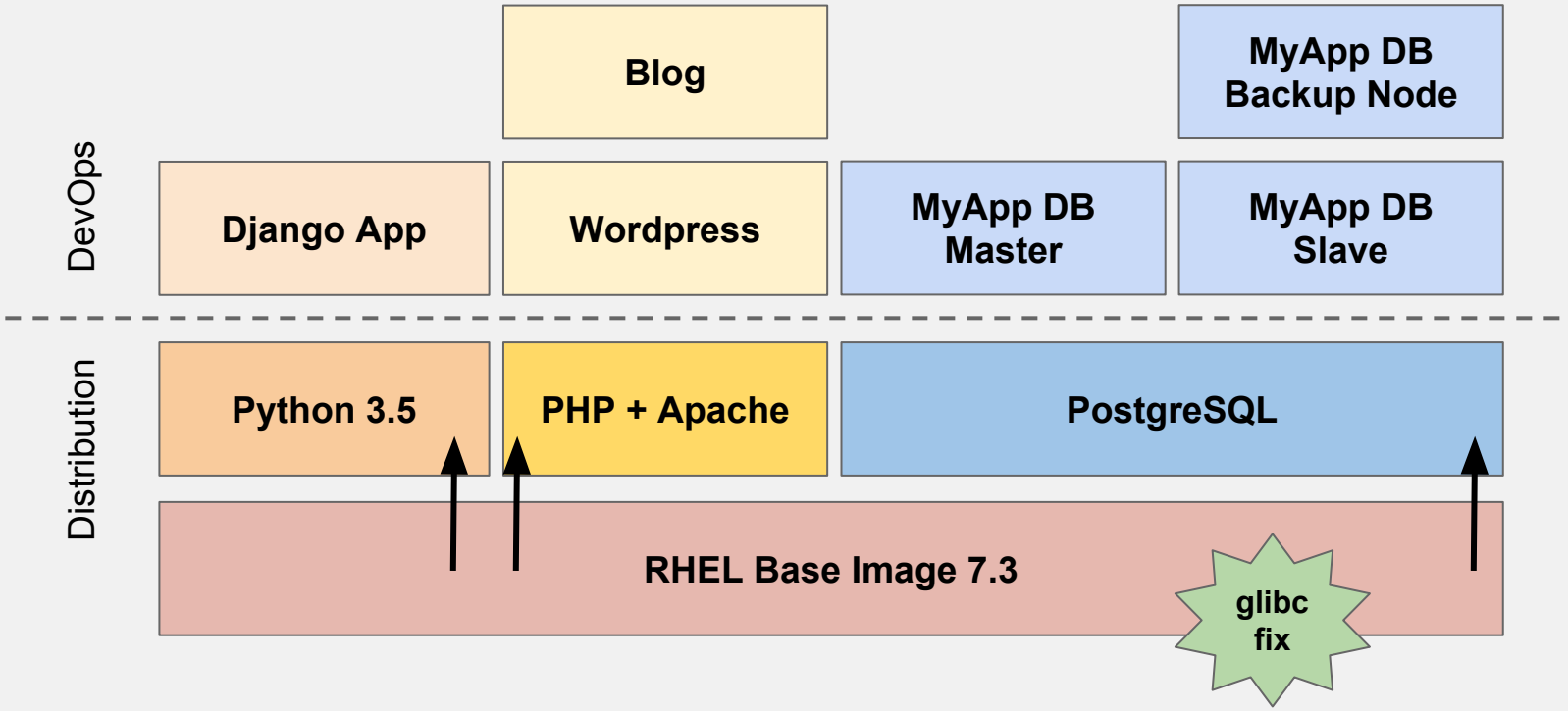
Docker and Layers

What is gonna be part of distribution and what is beyond.



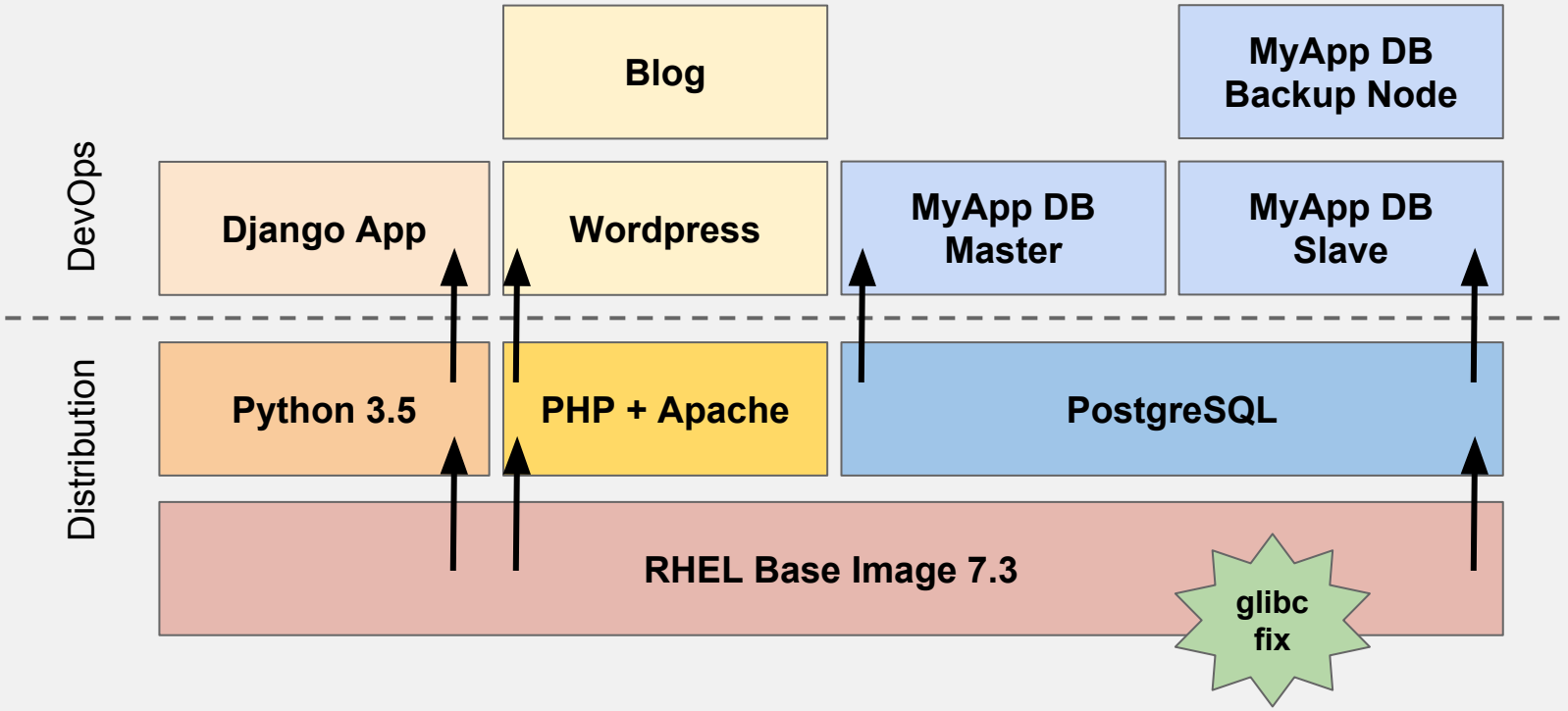
Docker and Layers

What is gonna be part of distribution and what is beyond.



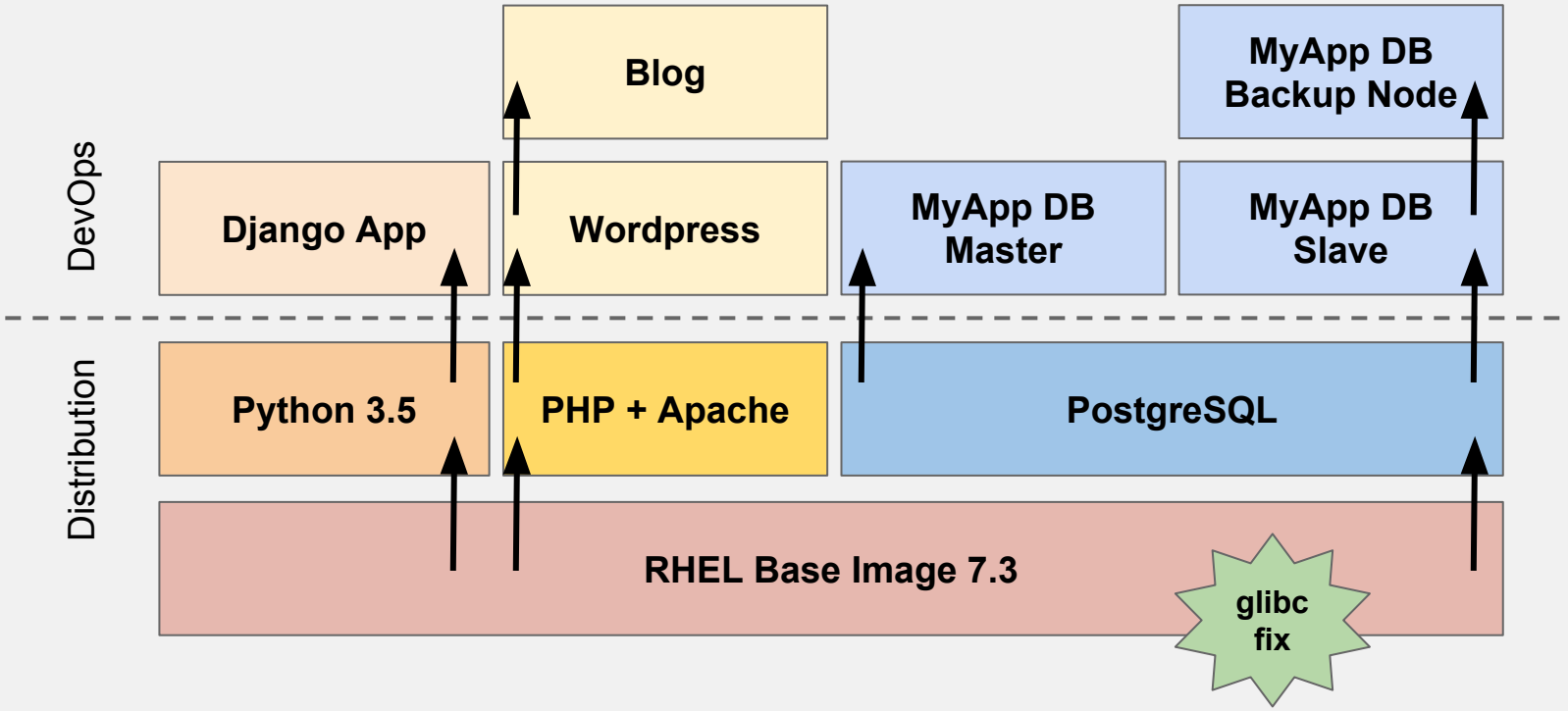
Docker and Layers

What is gonna be part of distribution and what is beyond.



Docker and Layers

What is gonna be part of distribution and what is beyond.

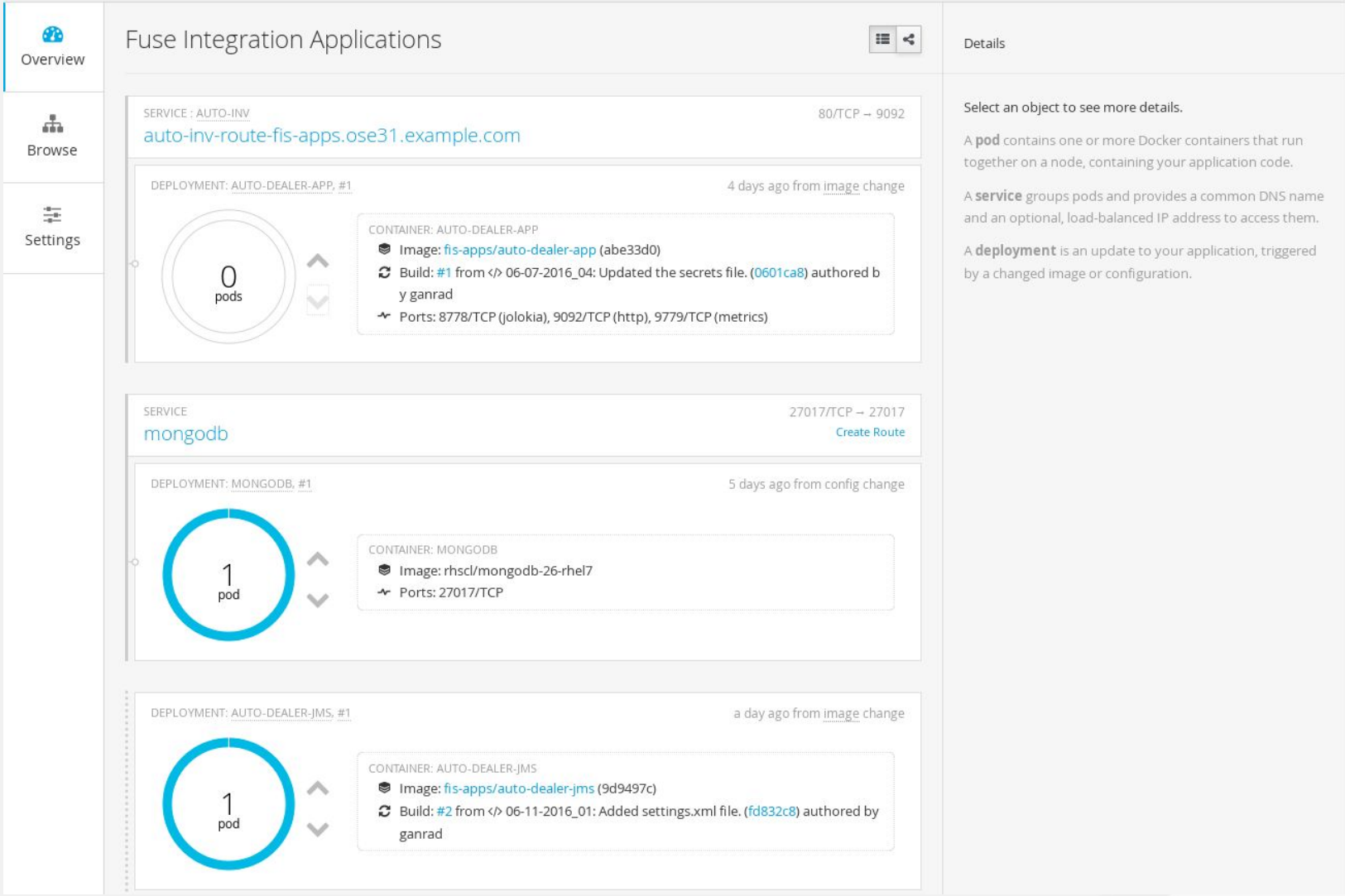


Automate.

Automate.
Everything.

Containers in Red Hat

...end in OpenShift, a PaaS based on kubernetes (k8s)



https://blog.openshift.com/red-hat-container-platform-3-3-general-availability/

2. POSTGRESQL DOCKER CONTAINER

Why actually?

It’s about efficiency.
But not about efficiency of sharing resources, it’s about efficiency of whole organization.
It’s about new thinking, a whole new paradigm.

Real Docker Adoption Is Up 30% in One Year.

https://www.datadoghq.com/docker-adoption/

Real Docker Adoption Is Up 30% in One Year.
2/3 of Companies That Try Docker Adopt It.

https://www.datadoghq.com/docker-adoption/

Real Docker Adoption Is Up 30% in One Year.
2/3 of Companies That Try Docker Adopt It.
PostgreSQL is 9th mostly used container image, using Docker to run relational databases is surprisingly common.

https://www.datadoghq.com/docker-adoption/

Building own PostgreSQL container

```
#> cat Dockerfile

FROM rhel7:7.3

RUN yum -y install postgresql-server && \
    yum clean all
```

Build the container

```
#> docker build -t hhorak/postgresql .
...
```

Correct RPMs are in a container

```
#> docker build -t hhorak/postgresql .
...

#> docker run -ti hhorak/postgresql
bash-4.2$ _
```

Correct RPMs are in a container

```
#> docker build -t hhorak/postgresql .
...

#> docker run -ti hhorak/postgresql
bash-4.2$ rpm -q postgresql-server
postgresql-server-9.2.18-1.el7.x86_64
```

So we have PostgreSQL in a Linux container.
Are we there yet?

Make container do something

```
#> cat Dockerfile

FROM rhel7:7.3

RUN yum -y install postgresql-server && yum clean all

ENV HOME=/var/lib/pgsql
ENV PGDATA=/var/lib/pgsql/data
ENV PGUSER=postgres
USER 26

COPY run-postgresql /usr/bin/
CMD [ "/usr/bin/run-postgresql" ]
```

Make container do something

Who said microservice?

```
#> cat run-postgresql

#!/bin/bash

initdb

echo "host all all 0.0.0.0/0 md5" >${PGDATA}/pg_hba.conf
echo "listen_addresses = '*'" >${PGDATA}/postgresql.conf

exec postgres "$@"
```

Connecting to PostgreSQL container

```
#> docker build -t hhorak/postgresql .

#> docker run -ti -p 5432:5432 --name p1 hhorak/postgresql

#> docker inspect --format='{{.NetworkSettings.IPAddress}}' p1
172.17.0.2

#> psql -h 172.17.0.2
Password: _
```

Connecting to PostgreSQL container

```
#> docker build -t hhorak/postgresql .

#> docker run -ti -p 5432:5432 --name p1 hhorak/postgresql

#> docker inspect --format='{{.NetworkSettings.IPAddress}}' p1
172.17.0.2

#> psql -h 172.17.0.2
Password: _
```

Do not use default passwords.

Connecting to PostgreSQL container

```
#> cat run-postgresql

...
echo "host all all 0.0.0.0/0 md5" >${PGDATA}/pg_hba.conf
echo "local all postgres peer" >>${PGDATA}/pg_hba.conf
echo "listen_addresses = '*'" >${PGDATA}/postgresql.conf

pg_ctl -w start -o "-h ''"
psql --command "ALTER USER \"postgres\" WITH ENCRYPTED PASSWORD
'${POSTGRESQL_ADMIN_PASSWORD}';"
pg_ctl stop

exec postgres "$@"
```

Connecting to PostgreSQL container

```
#> docker run -ti -p 5432:5432 --name p1 hhorak/postgresql

#> psql -h 172.17.0.2 -U postgres
Password for user postgres:
psql (9.2.18, server 9.2.18)
Type "help" for help.

postgres=# _
```

Or use k8s secrets

http://kubernetes.io/docs/user-guide/secrets/

```
if [ -e "/run/secrets/pgusers/user/username" ] ; then
  POSTGRESQL_USER="$(</run/secrets/pgusers/user/username)"
  POSTGRESQL_PASSWORD="$(</run/secrets/pgusers/user/password)"
fi
```

How to configure such a database?

Configuring PostgreSQL container

```
#> cat run-postgresql
...
echo "max_connections = ${POSTGRESQL_MAX_CONNECTIONS}"
>>${PGDATA}/postgresql.conf
...
```

Example of PostgreSQL 9.5 container

from Red Hat Software Collections

```
#> docker run -d \
          -p 5432:5432 \
          -e POSTGRESQL_ADMIN_PASSWORD=secret \
          -e POSTGRESQL_MAX_CONNECTIONS=10 \
          -e POSTGRESQL_USER=guestbook \
          -e POSTGRESQL_PASSWORD=pass \
          -e POSTGRESQL_DATABASE=guestbook \
          -v /db:/var/lib/pgsql/data:Z \
          rhscl/postgresql-95-rhel7
```

Auto-tuning in PostgreSQL container

Containers may be slim or fat.

```
# Get available memory for container
MEMORY_IN_BYTES=$(cat /sys/fs/cgroup/memory/memory.limit_in_bytes)

# Use 1/4 of given memory for shared buffers
POSTGRESQL_SHARED_BUFFERS="$(($MEMORY_IN_BYTES/1024/1024/4))MB"

# Setting effective_cache_size to 1/2 of total memory
POSTGRESQL_EFFECTIVE_CACHE_SIZE="$(($MEMORY_IN_BYTES/1024/1024/2))MB"

# postgresql.conf is later generated as:
shared_buffers = ${POSTGRESQL_SHARED_BUFFERS}
effective_cache_size = ${POSTGRESQL_EFFECTIVE_CACHE_SIZE}
```

Support the most common configuration (let users to change them in OpenShift) and allow users to build their own specific layered container images easily.

For working code, see:

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

For working code, see:

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

And contribute!

Or play with container directly

```
#> docker pull centos/postgresql-94-centos7
#> docker pull centos/postgresql-95-centos7

#> docker pull registry.access.redhat.com/rhscl/postgresql-94-rhel7
#> docker pull registry.access.redhat.com/rhscl/postgresql-95-rhel7
```

Q: What do I want to look at closely when I want to run 100 containers?

3. STANDALONE CONTAINER

Run container as systemd service

Alias replace a classic system service with a container

Handy for transition period

- Part of the services as containers, rest as standard services

We need to:

1. Create Docker container
2. Create systemd unit file for the service
3. Work with the systemd unit as usually

Run container as systemd service

1. Create Docker container (but not run)

```
#> docker create
          --name postgresql-service
          -e ...
          -v /var/lib/pgsql:/var/lib/pgsql:Z
          fedora/postgresql
```

Run container as systemd service

2. Create systemd unit file for the service

```
# cat /etc/systemd/system/postgresql-cont.service
[Unit]
Description=PostgreSQL service as a docker container
After=docker.service

[Service]
ExecStart=/usr/bin/docker start postgresql-cont
ExecStop=/usr/bin/docker stop postgresql-cont

[Install]
WantedBy=multi-user.target
```

Run container as systemd service

3. Work with the systemd service as usually

```
#> systemctl enable postgresql-cont.service
#> systemctl start postgresql-cont.service
```

Q: Is it possible to run a container as a systemd service?

4. TOOLS CONTAINERS

Not every container is a daemon

Tools to manage daemons

(that are not part of the daemon image)

```
#> docker run -ti hhorak/postgresql-tools pgbench ...

#> docker run -ti hhorak/postgresql-tools pg_standby
...
```

Interaction is easy, we can use network socket to work with daemon.

How to interact with host

```
#> docker exec -ti postgresql1 pg_dump >/home/hhorak/dump.sql
```

```
#> docker run -ti -v /:/host hhorak/postgresql bash
bash-4.2$ pg_dump >/host/home/hhorak/dump.sql
bash-4.2$ ...
```

Q: Is it possible to use containers for non-deamon applications?

5. GUI APPS IN CONTAINERS

Why GUI in containers

- Some level of isolation
  - Filesystem, Cgroups, Namespaces
- Deps bundling ➔one app running on any Linux
- Android-like app store for all distros
- Clean system underneath

GUI in Docker

Well, it sometimes even works, but..

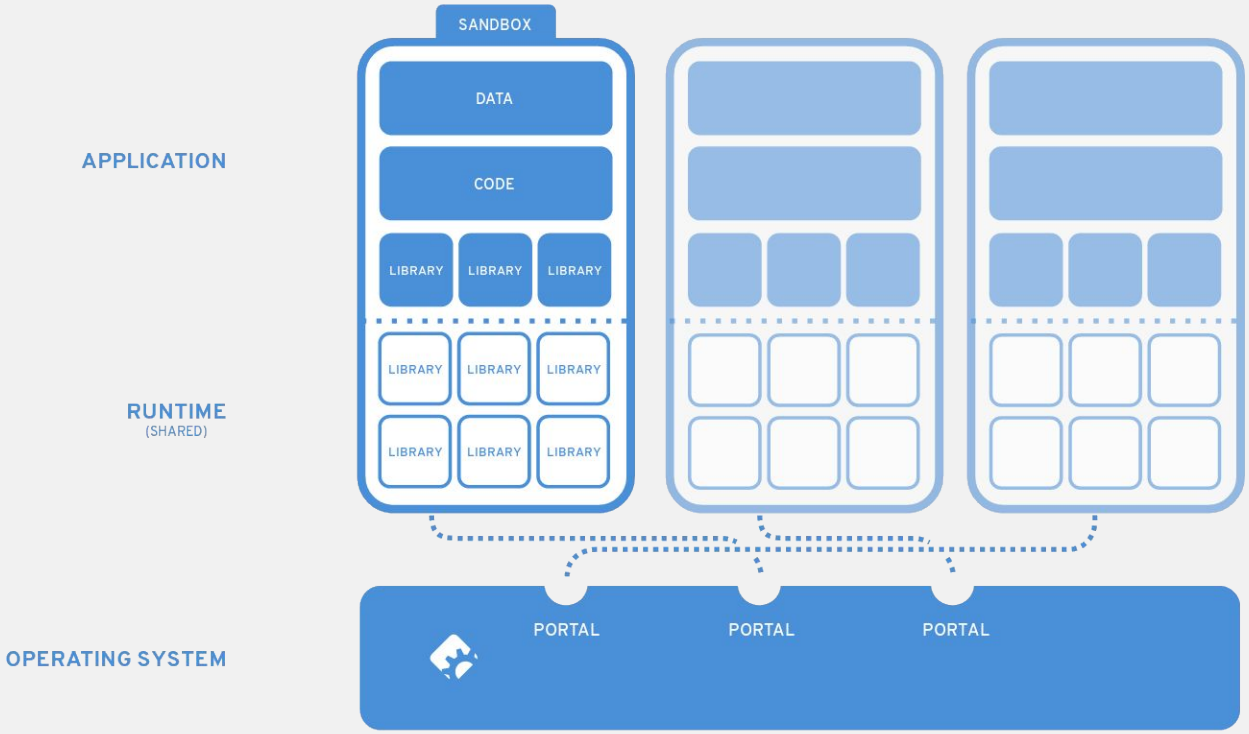
```
docker run -ti --rm \
       -e DISPLAY=$DISPLAY \
       -v /tmp/.X11-unix:/tmp/.X11-unix \
       -u 1001 \
       hhorak/pgadmin3
```

GUI in Flatpak

aka xdg-app

- Designed for GUI apps
- Sandbox rather than full container
- Runtimes define common environment to build on
- Runs in desktop session

Flatpak design



http://flatpak.org/#developers

Working with Flatpak application

http://flatpak.org/developer.html

```
$ tar xvf pgadmin3-1.22.1.tar.gz
$ cd pgadmin3-1.22.1

$ flatpak build ../dictionary ./configure --prefix=/app
$ flatpak build ../dictionary make
$ flatpak build ../dictionary make install
```

```
$ flatpak install gnome org.gnome.Platform 3.22
$ flatpak install gnome-apps org.postgresql.pgadmin3 stable

$ flatpak run org.postgresql.pgadmin3
```

Q: What type of applications Flatpack is designed for?

6. OS CONTAINERS

MultiContainer vs. OS Container

VS

Operating System Container

Running whole OS (systemd or other init process) inside container.

- Some applications work better when running on one machine
- Safe zombie handling
- Container’s journald integration

Running systemd inside Docker

http://developers.redhat.com/blog/2016/09/13/running-systemd-in-a-non-privileged-container

```
docker run -ti --tmpfs /run --tmpfs /tmp
           -v /sys/fs/cgroup:/sys/fs/cgroup:ro <your-image>
```

```
FROM fedora:25
RUN dnf -y install postgresql-server && dnf clean all
RUN systemctl enable postgresql
STOPSIGNAL SIGRTMIN+3
EXPOSE 80
CMD [ "/sbin/init" ]
```

OS Container using systemd-nspawn

Running whole OS (including systemd) inside container.

- better connection with host (logging, machinectl, ...)
- no image management, no containers linking
- good enough for testing or debugging something at container level

Running systemd inside nspawn

https://www.variantweb.net/blog/using-systemd-nspwan-for-lightweight-container-in-fedora-21/

```
dnf -y --releasever=25 --nogpg --installroot=/var/tmp/testnspawn
    --disablerepo='*' --enablerepo=fedora --enablerepo=updates install
systemd passwd yum fedora-release postgresql-server @standard
chcon -R -t svirt_sandbox_file_t /var/tmp/testnspawn
restorecon -R /var/tmp/testnspawn
```

```
systemd-nspawn -D /var/tmp/testnspawn
passwd
postgresql-setup --init
systemctl enable postgresql
```

```
systemd-nspawn -bD /var/tmp/testnspawn
```

Q: Is it possible to run docker image in systemd-nspawn?

Who likes Ansible?

7. Ansible containers

Ansible containers

- Building containers using popular Ansible
- Orchestrating containers like other services
- Especially useful for transition to containers

Ansible Container

https://github.com/ansible/ansible-container-examples/tree/master/wordpress

```
services:
  db:
    image: rhel:7
    ports:
      - "5432:5432"
    command: ['/usr/bin/pg_ctl', '-w', 'start']

  wordpress:
    image: rhel:7
    ports:
      - "80:80"
    links:
      - db
command: bash -c "bash /tmp/a.sh ; usr/sbin/apachectl -D FOREGROUND"
```

Ansible container

```
- hosts: db
  vars:
    - wp_pgsql_db: wordpress
    - wp_pgsql_user: wordpress
    - wp_pgsql_password: password
  tasks:
    - name: Install postgresql-server
      yum:
        name: "{{ item }}"
        state: latest
      with_items:
        - postgresql-server

    - name: Update the repository
      shell: yum -y erase vim-minimal && \
             yum -y update && \
             yum clean all
```

```
    - name: postgresql-init
      shell: postgresql-setup --upgrade

    - name: run postgres
      shell: pg_ctl -w start

    - name: Create postgres database
      pgsql_db:
        name: "{{ wp_pgsql_db }}"
        state: present

    - name: Create postgres user
      pgsql_user:
        name: "{{ wp_pgsql_user }}"
        password: "{{ wp_pgsql_password }}"
        state: present
        priv: "*.*:ALL,GRANT"
        host: "%"
```

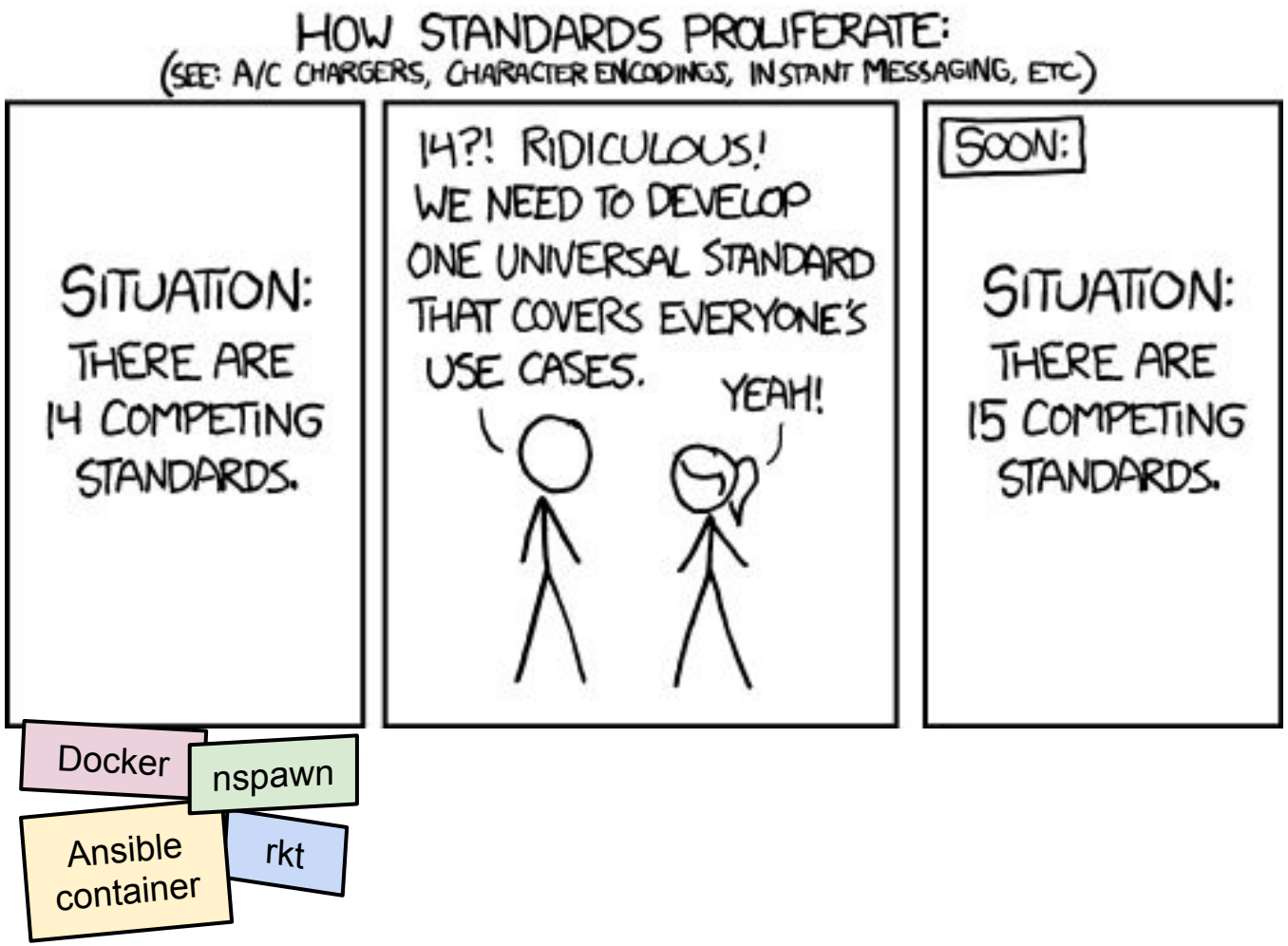
Ansible Container

https://github.com/ansible/ansible-container-examples/tree/master/wordpress

```
$> ansible-container init
 -- do changes --
$> ansible-container build
$> ansible-container run
$> docker login
$> ansible-container push
$> ansible-container shipit openshift
```

8. OCI

8. OCI
Open Container Initiative


HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)
SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.
Docker
nspawn
Ansible container
rkt

"The mission of the OCI is to promote and promulgate
a set of common, minimal, open standards
and specifications around container technology."
https://www.opencontainers.org

Open Container Initiative

Collaboration of Red Hat, Google, Docker, and others big players in containers world.

- Image tools - https://github.com/opencontainers/image-tools
- runc - https://github.com/opencontainers/runc
- Open Container Format

Q: Will I build my container today?

Thanks.

OpenShift: https://docs.openshift.com
Sources of Docker images: https://github.com/sclorg/
Project Atomic: https://www.projectatomic.io
OCI: https://www.opencontainers.org/

Honza Horak <hhorak@redhat.com>
@HonzaHorak