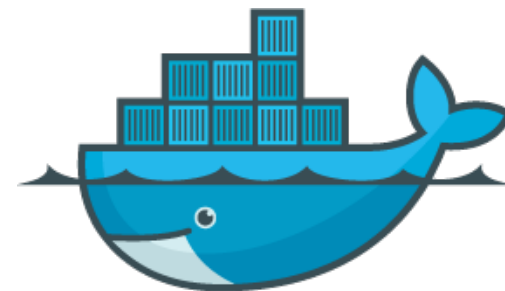# Orchestrating Docker containers at scale

Maciej Lasyk

Jesień Linuksowa 2014

Szczyrk, 2014-11-09

Join Fedora Infrastructure!

- learn Ansible
- learn Docker with Fedora Dockerfiles

http://fedoraproject.org/en/join-fedora

Agenda

- Why use Docker?
- Brief history of envs evolution
- Docker – what is it?
- To PaaS or not to PaaS
- Orchestration at scale

Quick survey

- How many of you...

Quick survey

- How many of you...
    - Knows what Docker is?

Quick survey

- How many of you...
    - Knows what Docker is?
    - Played with Docker?

Quick survey

- How many of you...
    - Knows what Docker is?
    - Played with Docker?
    - Runs it on production?

Why use Docker?

With Docker we can solve many problems

Why use Docker?

With Docker we can solve many problems

- "it works on my machine"

Why use Docker?

With Docker we can solve many problems

- "it works on my machine"
- reducing build & deploy time

Why use Docker?

With Docker we can solve many problems

- "it works on my machine"
- reducing build & deploy time
- Infrastructure configuration spaghetti – automation!

Why use Docker?

With Docker we can solve many problems

- "it works on my machine"
- reducing build & deploy time
- Infrastructure configuration spaghetti – automation!
- Libs dependency hell

Why use Docker?

With Docker we can solve many problems

- "it works on my machine"
- reducing build & deploy time
- Infrastructure configuration spaghetti – automation!
- Libs dependency hell
- Cost control and granularity

A brief story of envs evolution

- classical approach (for & scp/ssh & conf.d)

A brief story of envs evolution

- classical approach (for & scp/ssh & conf.d)
- configuration management

A brief story of envs evolution

- classical approach (for & scp/ssh & conf.d)
- configuration management
- Bare vs IaaS vs PaaS

A brief story of envs evolution

- classical approach (for & scp/ssh & conf.d)
- configuration management
- Bare vs IaaS vs PaaS
- Continuous Integration

A brief story of envs evolution

- classical approach (for & scp/ssh & conf.d)
- configuration management
- Bare vs IaaS vs PaaS
- Continuous Integration
- So when to use Docker?

Docker – what is it?

Docker – what is it?

"automates the deployment of any application as a lightweight, portable, self-sufficient container
that will run virtually anywhere"

Docker – what is it?

Java's promise: Write Once. Run Anywhere.

Docker – what is it?

Java's promise: Write Once. Run Anywhere.

Even on Windows now!

https://blog.docker.com/2014/10/docker-microsoft-partner-distributed-applications/
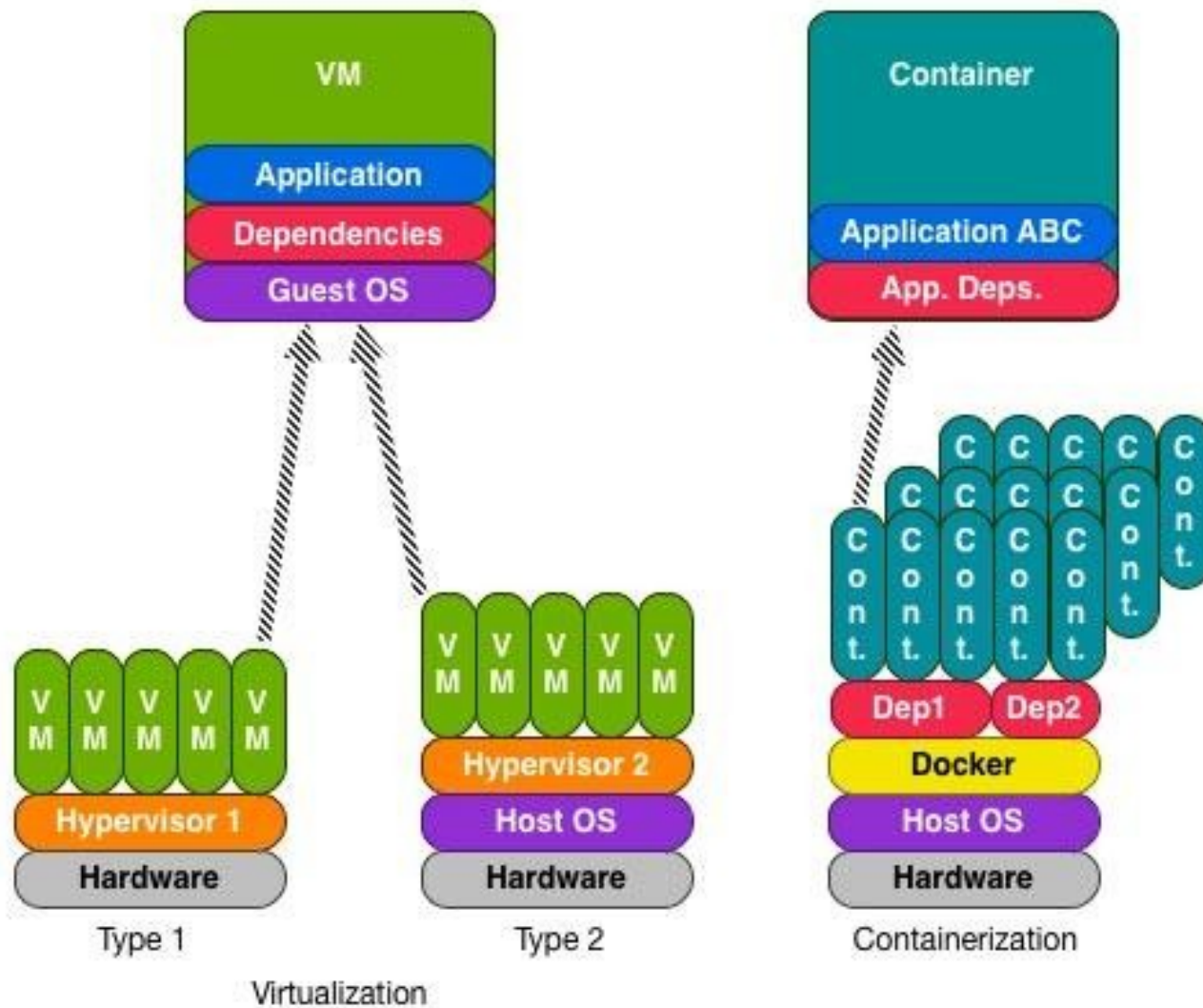
Docker – what is it?

Docker is lightweight

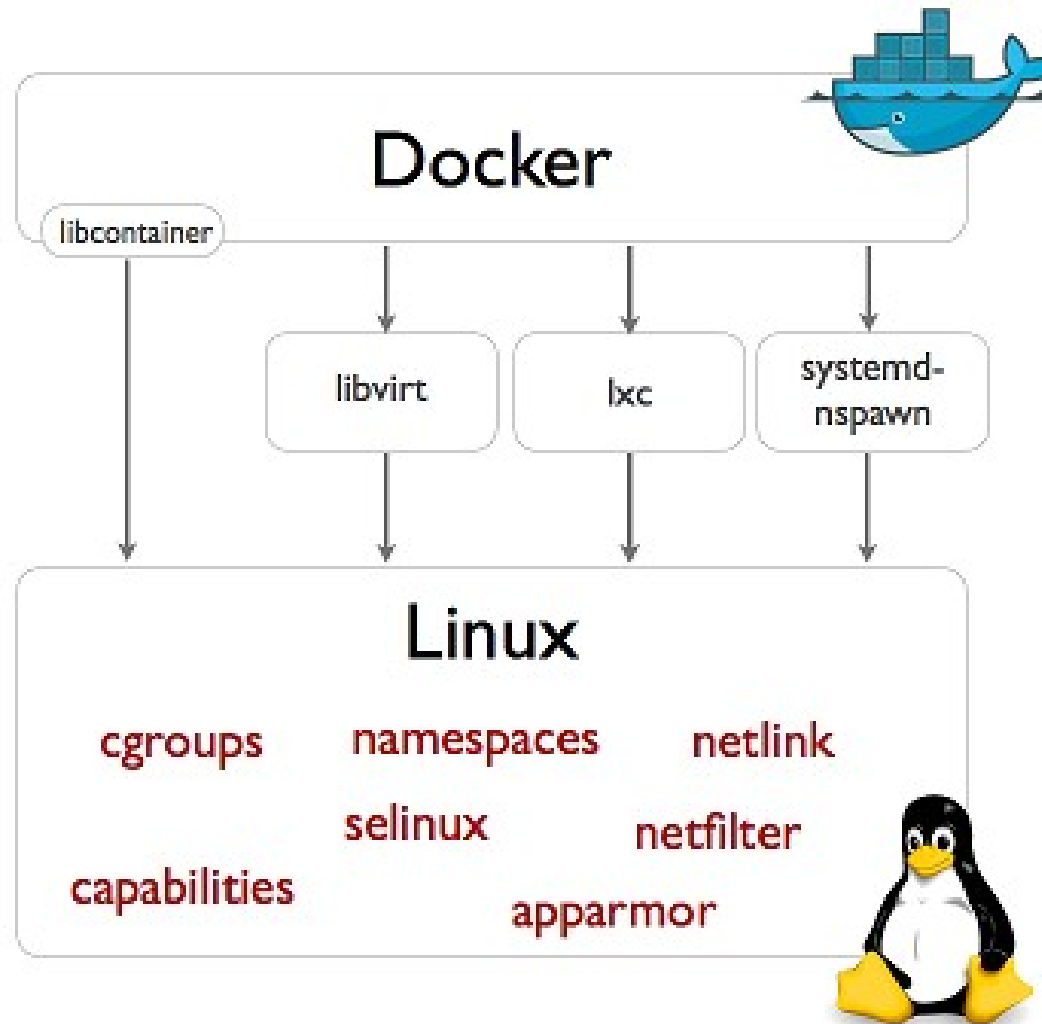# Docker – what is it?

Docker is lightweight

```
==================================================================
 Package            Arch            Version              Repository        Size
==================================================================
Installing:
 docker-io          x86_64          1.3.0-1.fc20         updates          4.3 M
```

Docker – what is it?

VMs vs. Containers

Docker – how it works?

Docker – how it works?

- LXC & libcontainer

Docker – how it works?

- LXC & libcontainer
- control groups

Docker – how it works?

- LXC & libcontainer
- control groups
- kernel namespaces

Docker – how it works?

- LXC & libcontainer

- control groups

- kernel namespaces

- layered filesystem

    - btrfs

    - devmapper thin provisioning & loopback mounts

    - no more AUFS

    - http://developerblog.redhat.com/2014/09/30/overview-storage-scalability-docker/

Docker – concepts

- images
    - read only
    - act as templates

Docker – concepts

- images
    - read only
    - act as templates
- Dockerfile
    - like a makefile
    - commands order & cache'ing
    - extends the base image
    - results in a new image

Docker – concepts

- images
    - read only
    - act as templates
- Dockerfile
    - like a makefile
    - commands order & cache'ing
    - extends the base image
    - results in a new image
- Containers: instances running apps

Docker – concepts

- images
    - read only
    - act as templates
- Dockerfile
    - like a makefile
    - commands order & cache'ing
    - extends the base image
    - results in a new image
- Containers: instances running apps

       dockerfile + base image = docker container
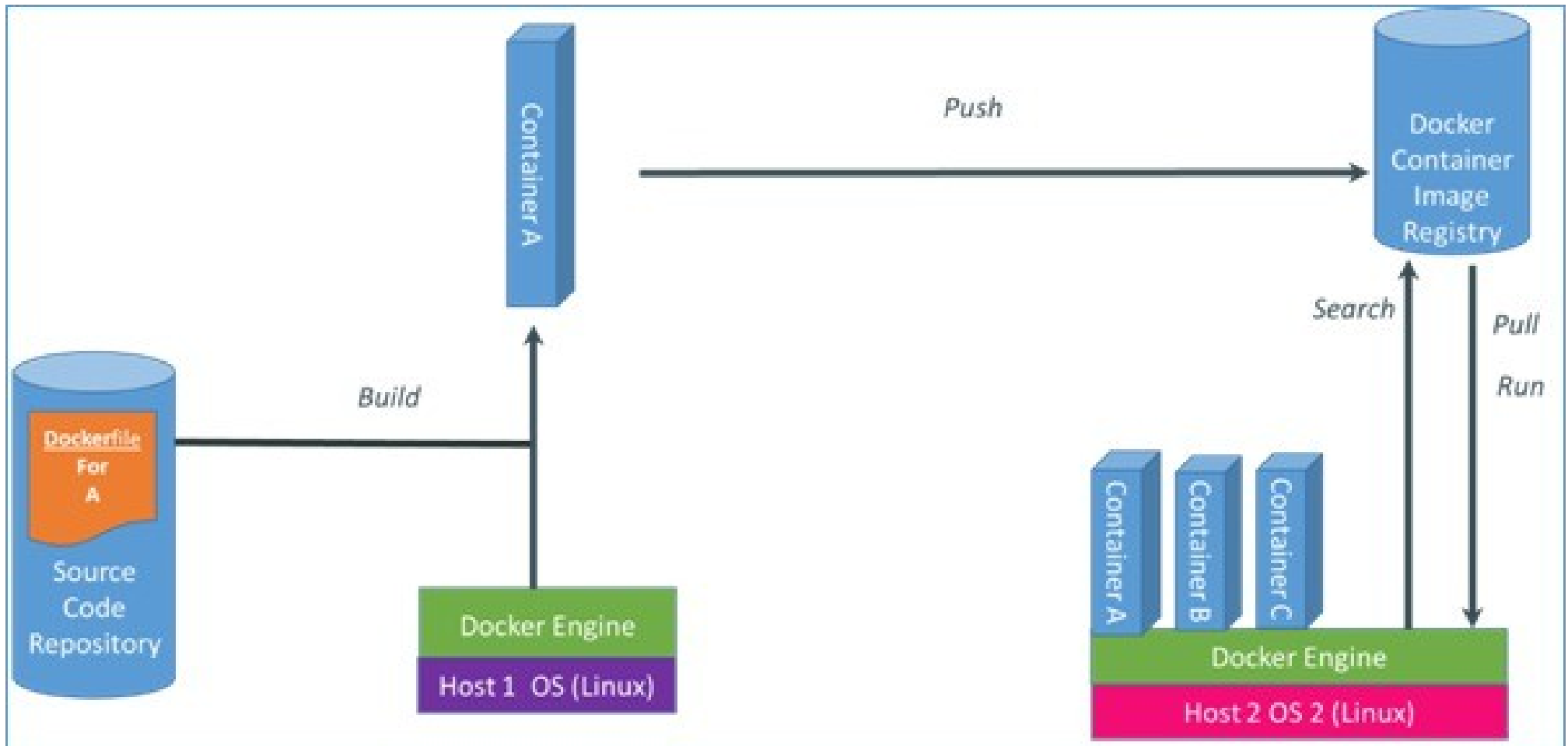
Dockerfile

```
FROM fedora
MAINTAINER scollier <scollier@redhat.com>

RUN yum -y update && yum clean all
RUN yum -y install nginx && yum clean all
RUN echo "daemon off;" >> /etc/nginx/nginx.conf
RUN echo "nginx on Fedora" > /srv/www/index.html

EXPOSE 80

CMD [ "/usr/sbin/nginx" ]
```

Docker – registry

Docker – registry

- git like semantics
- pull, push, commit
- private and public registry
- https://github.com/dotcloud/docker-registry
- yum install docker-registry

  $ docker pull
  $ docker push
  $ docker commit

Docker – images hierarchy

```
base image
    -> child image
        -> grandchild image
```

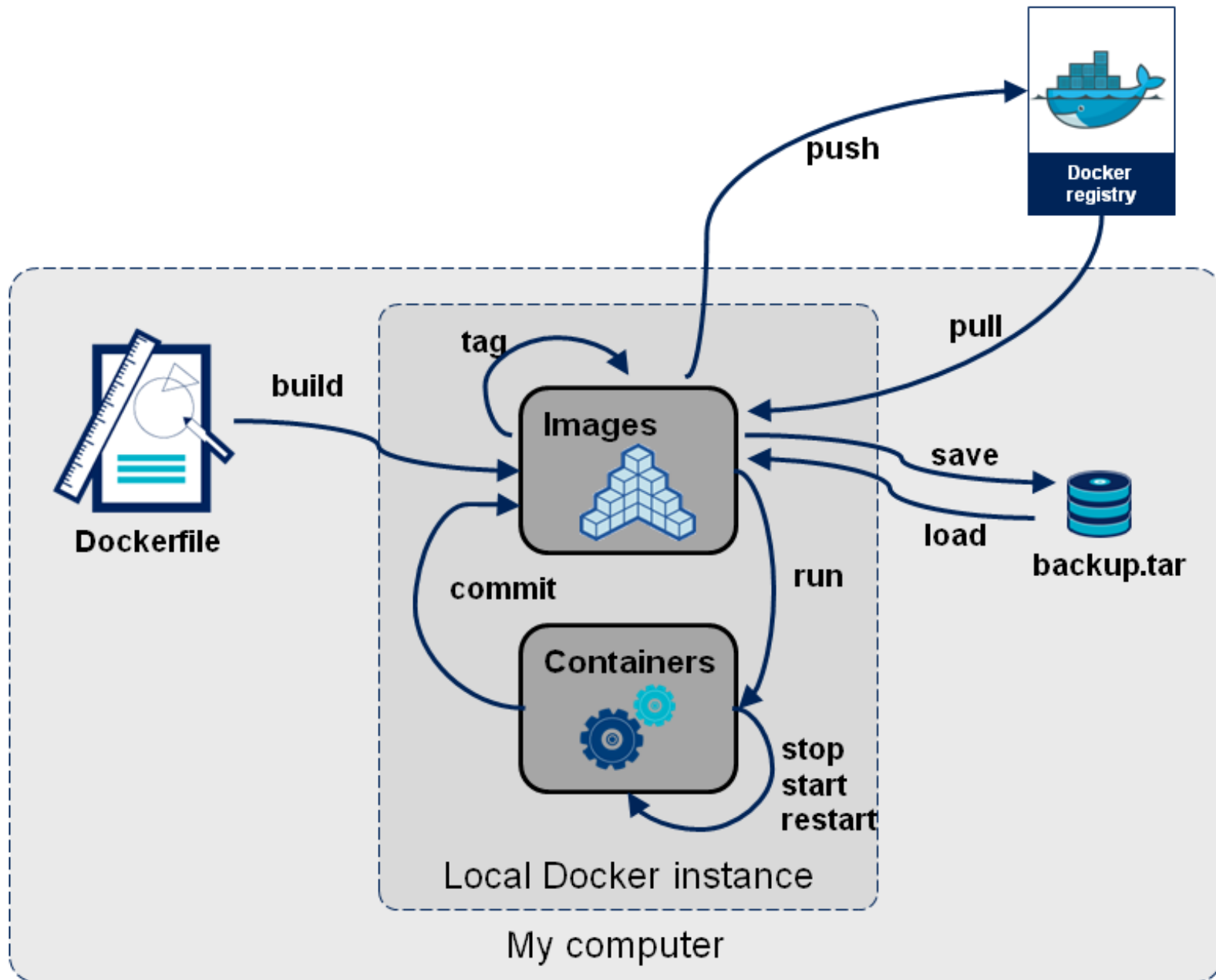# Docker – images hierarchy

base image
    -> child image
        -> grandchild image

Git's promise: Tiny footprint with
lightning fast performance

Docker – images hierarchy

Docker – security

- Isolation via kernel namespaces
- Additional layer of security: SELinux / AppArmor / GRSEC
- Each container gets own network stack
- control groups for resources limiting

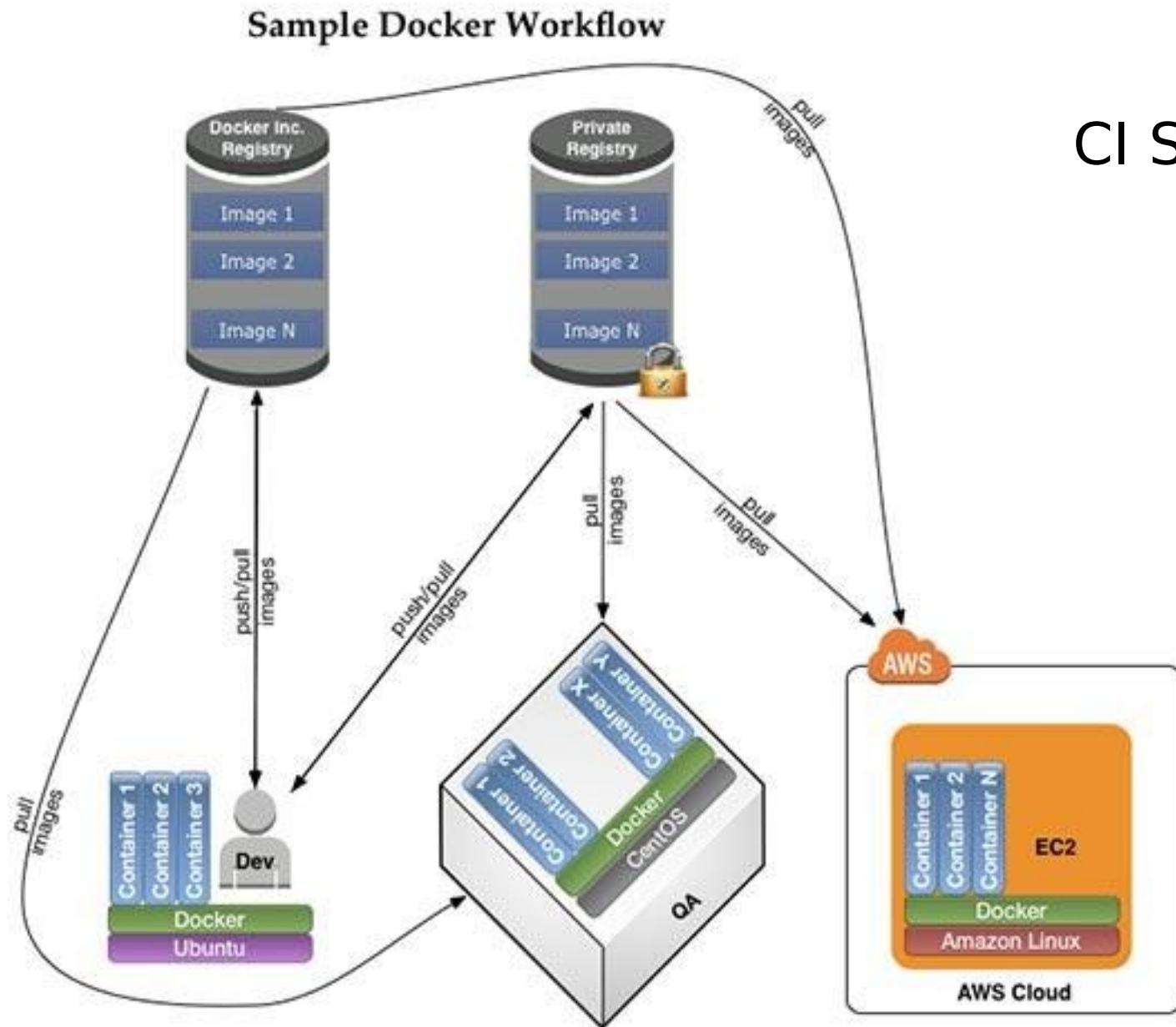f20 policy: https://git.fedorahosted.org/cgit/selinux-policy.git/tree/docker.te?h=f20-contrib

What's there?

seinfo -t -x | grep docker

sesearch -A -s docker_t (and the rest)

or just unpack docker.pp with semodule_unpackage

Docker – use cases

Sample Docker Workflow

CI Stack

Docker – use cases

Continuous Integration

- local dev
    - with Docker it's easy to standardize envs
- deployment
    - rolling updates (e.g. w/Ansible)
- testing
    - unit testing of any commit on dedicated env
    - don't worry about cleaning up after testing
    - parrarelized tests across any machines

Docker – use cases

- version control system for apps
- microservices
    - Docker embraces granularity
    - Services can be deployed independently and faster
    - parallelized tests across any machines
- continuous delivery
- PaaS

Docker – history

- 2013-01: dotCloud worked on own PaaS (Python based)
- 2013-03: Docker went public (AUFS, LXC)
- middle 2013: Red Hat joined, devmapper, SELinux
- late 2013: removed LXC, rewritten in Go
- 2014-02: stable 1.0

Docker – popularity
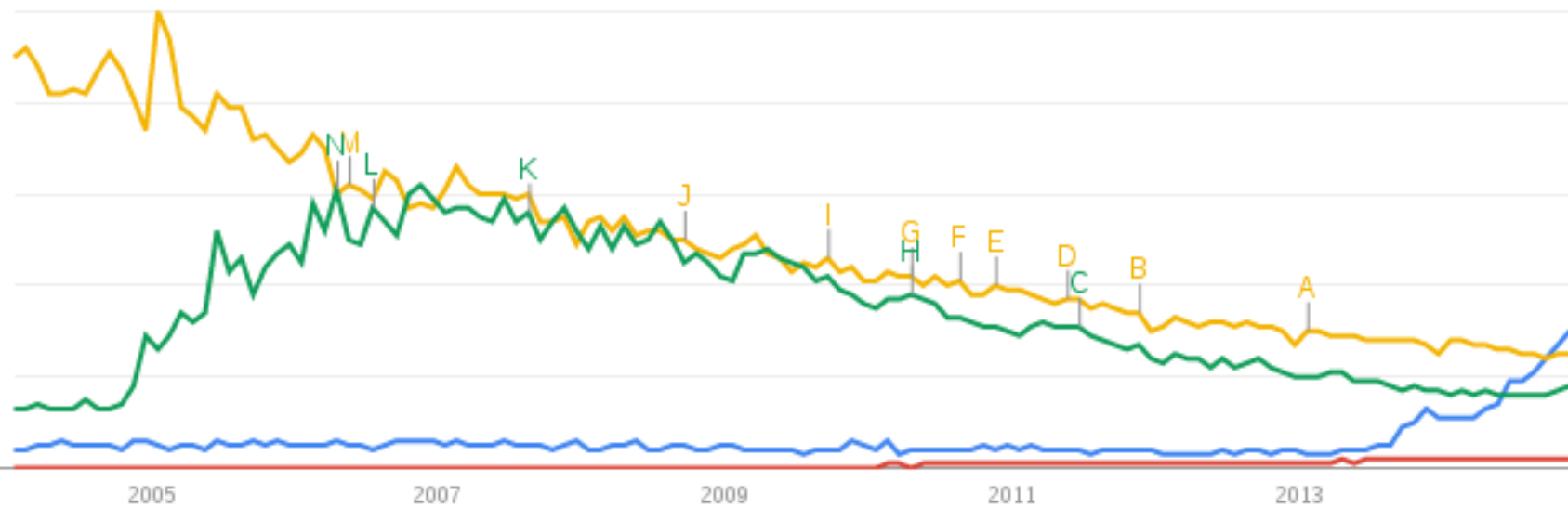
Docker – popularity
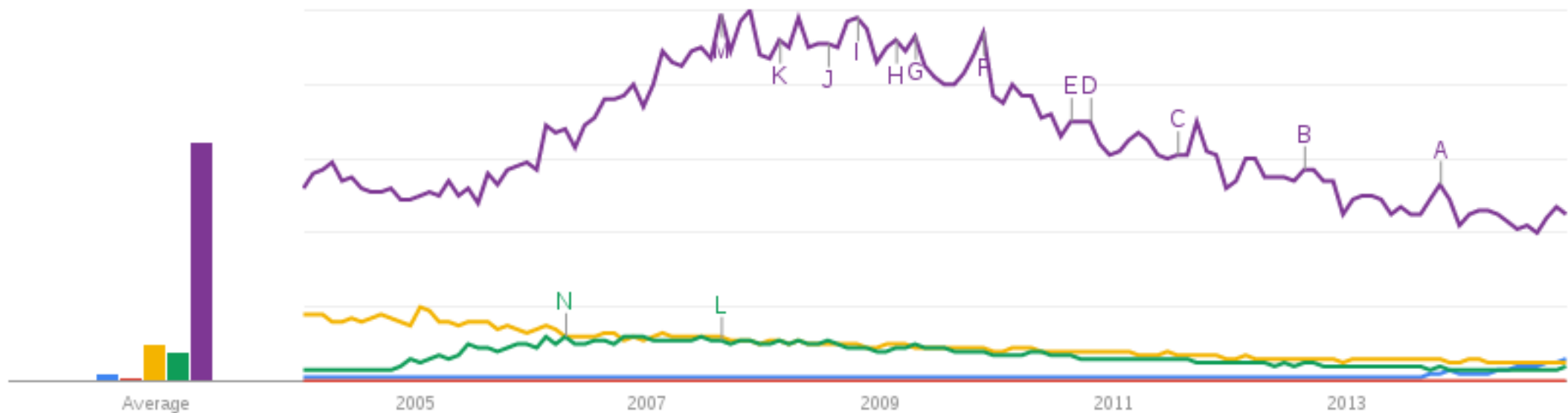
Orchestration at scale w/Docker

Orchestration at scale w/Docker

This might be a little problem

Orchestration at scale w/Docker

Orchestration at scale w/Docker

	Big Data	Cloud Platform	IaaS	Data Center OS	Docker OS	Docker Mgmt.	PaaS	Orch. Config Mgmt.
Ansible & Docker								x
Amazon EC2 & Docker		x						
Apache Brooklyn & Docker								x
Apache Hadoop & Docker	x							
Apache Storm & Docker	x							
AppScale & Docker							x	
Atomic Hosts & Docker					x			
Chef & Docker								x
Clocker & Docker								x
Cloud Foundry & Docker	x						x	
CloudStack & Docker			x					
CoreOS & Docker					x			
Deis & Docker							x	
Decker & Docker							x	
Docker & Docker			x		x	x	x	x
Dokku & Docker							x	
Eucalyptus & Docker			x					
Flynn & Docker							x	
Google Compute Platform & Docker		x						
IBM Bluemix & Docker	x						x	
Kubernetes & Docker			x			x	x	x
Mesos, Mesosphere & Docker	x			x		x	x	x
Microsoft Azure & Docker		x						
OpenCamp & Docker		x	x			x	x	
OpenShift & Docker							x	
OpenStack & Docker			x					
Panamax & Docker						x		
Puppet & Docker								x
SaltStack & Docker							x	x
Shipyard & Docker						x		
Stackato & Docker							x	
Tsuru & Docker							x	
VMware & Docker			x					

http://www.cloudssky.com/en/blog/Docker-Is-Not-Enough

To Paas or not to PaaS?

To Paas or not to PaaS?

- you think PaaS will solve your problems?

To Paas or not to PaaS?

- you think PaaS will solve your problems?
- it will rather clone them :)

To Paas or not to PaaS?

- you think PaaS will solve your problems?
- it will rather clone them :)
- running PaaS might be easy

To Paas or not to PaaS?

- you think PaaS will solve your problems?
- it will rather clone them :)
- running PaaS might be easy
- operating PaaS will be tough

To Paas or not to PaaS?

- you think PaaS will solve your problems?
- it will rather clone them :)
- running PaaS might be easy
- operating PaaS will be tough
- so is operating your apps tough enough to move?

To Paas or not to PaaS?

- you think PaaS will solve your problems?
- it will rather clone them :)
- running PaaS might be easy
- operating PaaS will be tough
- so is operating your apps tough enough to move?
- private PaaS or not?

To Paas or not to PaaS?

- you think PaaS will solve your problems?
- it will rather clone them :)
- running PaaS might be easy
- operating PaaS will be tough
- so is operating your apps tough enough to move?
- private PaaS or not?
- Rainbow and Unicorn Piss:

    http://blog.lusis.org/blog/2014/06/14/paas-for-realists/

To Paas or not to PaaS?

- flynn.io
    - Open source PaaS (Go)
    - Uses Docker to manage containers
    - Git push deployment
    - https://flynn.io

To Paas or not to PaaS?

- flynn.io
    - Open source PaaS (Go)
    - Uses Docker to manage containers
    - Git push deployment
    - https://flynn.io
- dokku
    - The smallest PaaS implementation you've ever seen
    - Less than 100 lines of Bash
    - Git push deployment
    - https://github.com/progrium/dokku

To Paas or not to PaaS?

- flynn.io
    - Open source PaaS (Go)
    - Uses Docker to manage containers
    - Git push deployment
    - https://flynn.io
- dokku
    - The smallest PaaS implementation you've ever seen
    - Less than 100 lines of Bash
    - Git push deployment
    - https://github.com/progrium/dokku
- deis.io
    - Open source PaaS (Python)
    - Git push deployment
    - On top of CoreOS w/ Docker
    - http://deis.io/

To Paas or not to PaaS?

Remember, that PaaS might fail; plan & test for disaster !

Docker & CLI

```
$ docker run -t -i fedora /bin/bash
> yum -y update
> yum -y install nginx
$ docker commit 73fa45674fd docent/fedora
```

Docker & CLI

Or via Dockerfile:

$ docker build -t fedora –rm .
$ docker run --name=nginx fedora

FIG

- http://www.fig.sh
- for single host env

FIG

- http://www.fig.sh
- for single host env

```
FROM python:2.7
ENV PYTHONUNBUFFERED 1
RUN mkdir /code
WORKDIR /code
ADD requirements.txt /code/
RUN pip install -r requirements.txt
ADD . /code/
```

FIG

```
db:
  image: postgres
web:
  build: .
  command: python manage.py
runserver 0.0.0.0:8000
  volumes:
    - /srv/app/code:/code
  ports:
    - "8000:8000"
  links:
    - db
```

FIG

$ fig run web django-admin.py startproject figexample .
$ fig up

Management during runtime?
$ fig run web python manage.py syncdb

Docker & Ansible

Ansible + Docker

Vs

Docker + Ansible

Docker & Ansible

Ansible docker core module:
http://docs.ansible.com/docker_module.html

```
- hosts: web
  sudo: yes
  tasks:
  - name: run tomcat servers
    docker: >
        image=centos
        command="service tomcat6 start"
        ports=8080
        count=5
        memory_limit=128MB
        link=mysql
        expose=8080
        registry=...
        volume=...
```

Docker & Ansible

Building image with Ansible:

```
FROM ansible/centos7-ansible:stable
ADD ansible /srv/example/
WORKDIR /srv/example
RUN ansible-playbook web.yml -c local
EXPOSE 80
ENTRYPOINT ["/usr/sbin/nginx", "-DFOREGROUND"]
```

Docker & Ansible

Building image with Ansible:

```
FROM ansible/centos7-ansible:stable
ADD ansible /srv/example/
WORKDIR /srv/example
RUN ansible-playbook web.yml -c local
EXPOSE 80
ENTRYPOINT ["/usr/sbin/nginx", "-DFOREGROUND"]

ansible/web.yml:
- hosts: localhost
tasks:
- yum: pkg=nginx state=latest
- copy:
    src: web.conf
    dest: /etc/nginx/conf.d/web.conf
    group: "nginx"
    mode: "0644"
```

CoreOS

- Designed for massive server deployments
- Support Docker container out of the box
- Available on Linux, Mac, and Windows
- It's a Chrome OS fork
- Consists of couple of components:
    - SystemD – not just a init system ;)
    - Fleet – cluster level manager & scheduler
    - etcd – light & distributed key / value store
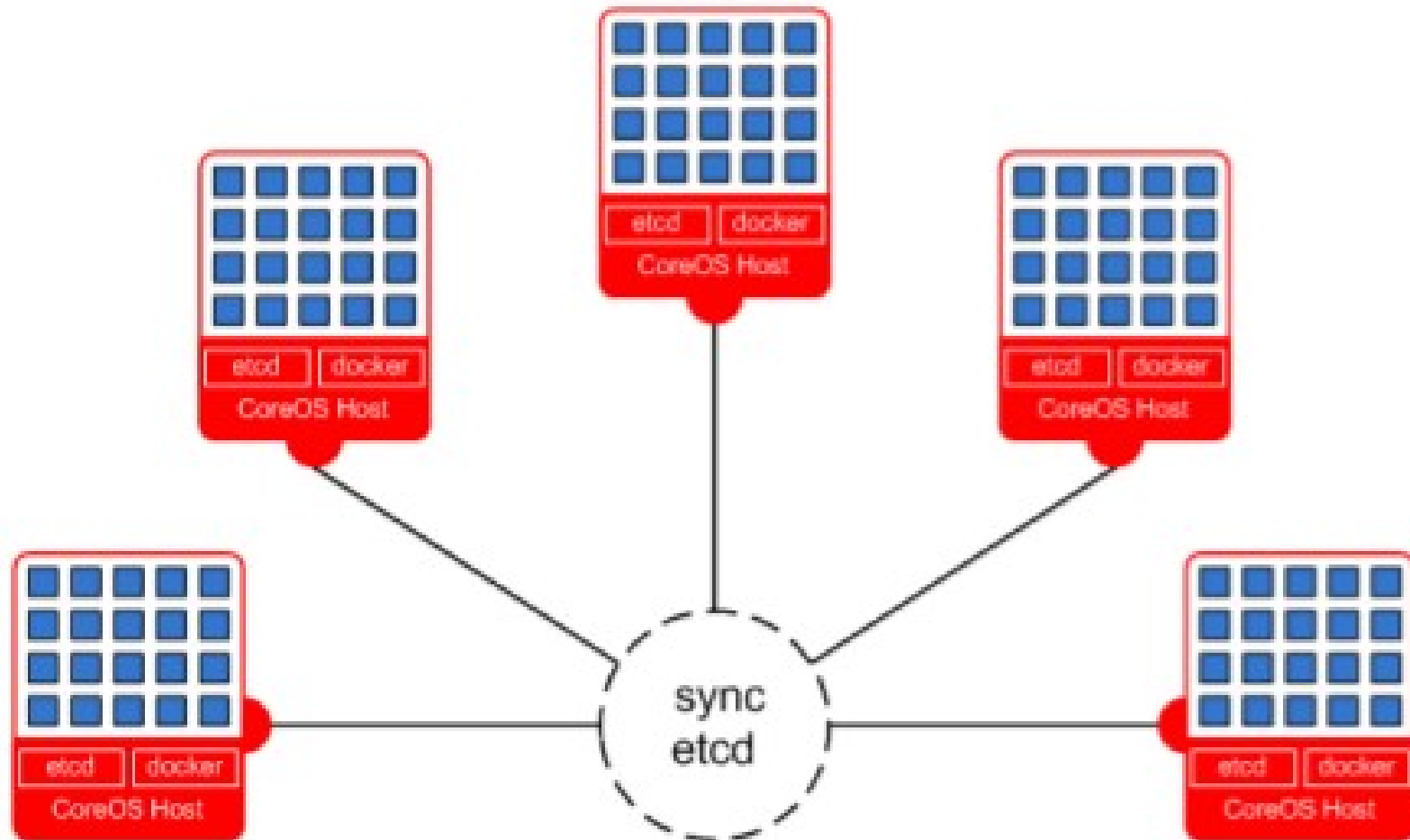   - Docker – the only packaging method in CoreOS
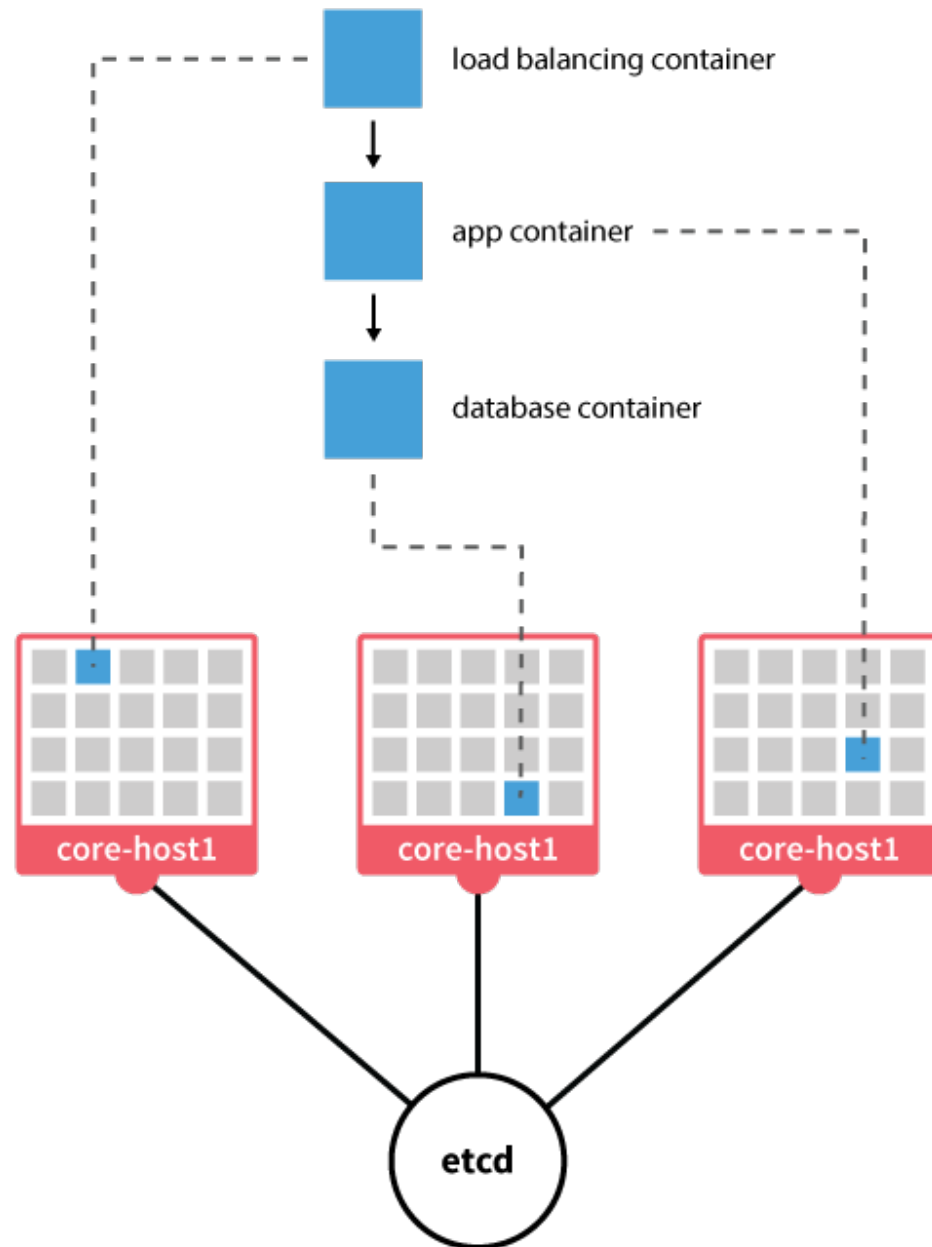
CoreOS

Fleet – cluster level manager & scheduler
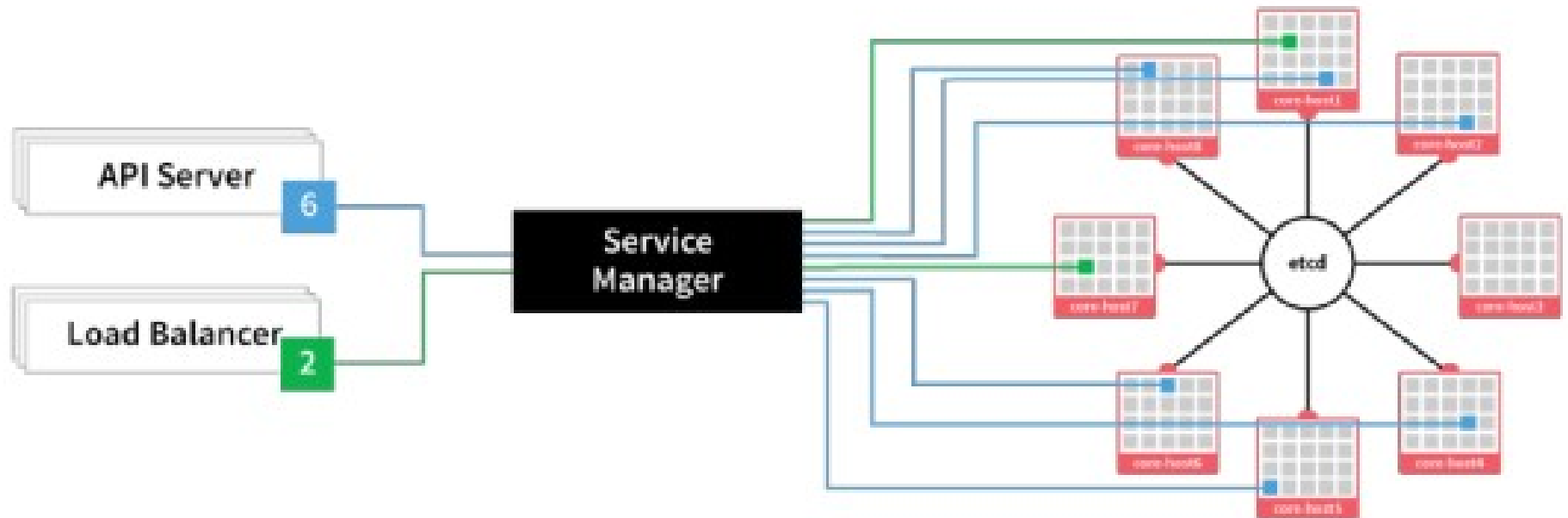
CoreOS

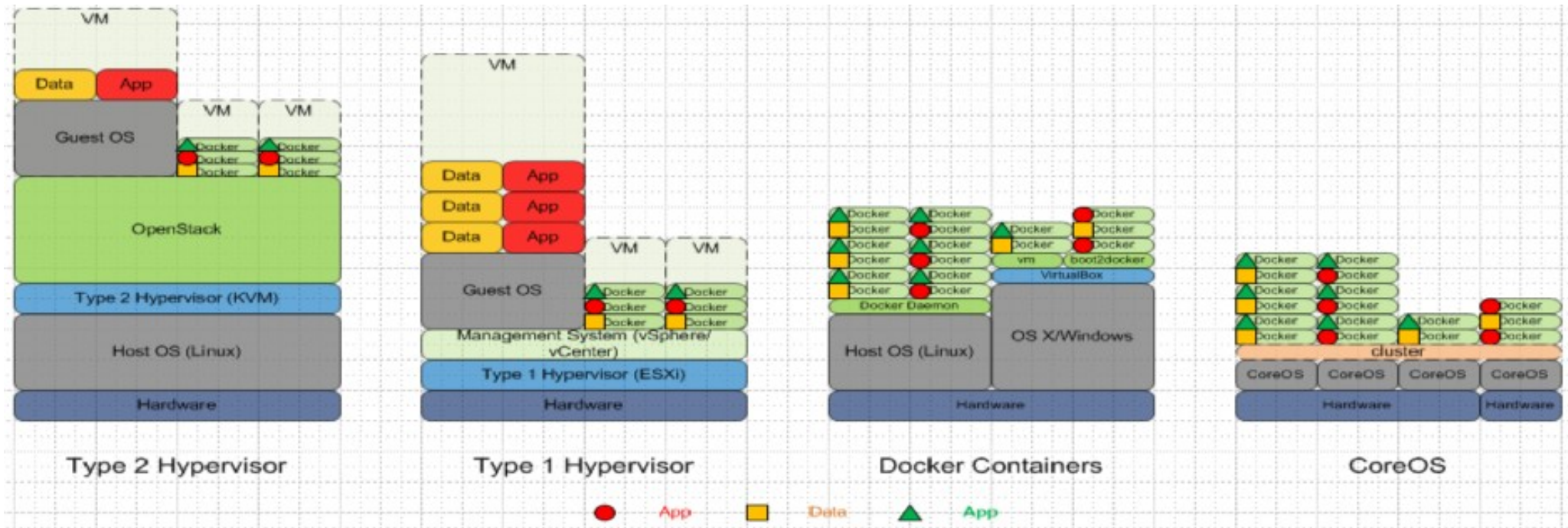etcd – light & distributed key / value store
(used for configuration store)

CoreOS

Docker – the only packaging method in CoreOS

CoreOS

CoreOS

Cluster management with Fleet & SystemD

```
[Unit]
Description=My Service
After=docker.service

[Service]
TimeoutStartSec=0
ExecStartPre=-/usr/bin/docker kill hello
ExecStartPre=-/usr/bin/docker rm hello
ExecStartPre=/usr/bin/docker pull busybox
ExecStart=/usr/bin/docker run --name hello busybox /bin/sh -c
                          "while true; do echo Hello World; sleep 1; done"
ExecStop=/usr/bin/docker stop hello
```

CoreOS

Cluster management with Fleet & SystemD

```
$ fleetctl load hello.service
Unit hello.service loaded on 8145ebb7.../10.10.1.3
$ fleetctl start hello.service
Unit hello.service launched on 8145ebb7.../10.10.1.3

$ fleetctl list-machines
MACHINE                             IP          METADATA
148a18ff-6e95-4cd8-92da-c9de9bb90d5a   10.10.1.1  -
491586a6-508f-4583-a71d-bfc4d146e996   10.10.1.2  -
c9de9451-6a6f-1d80-b7e6-46e996bfc4d1   10.10.1.3  -

$ fleetctl list-units
UNIT           MACHINE              ACTIVE    SUB
hello.service  c9de9451.../10.10.1.3  active    running
```
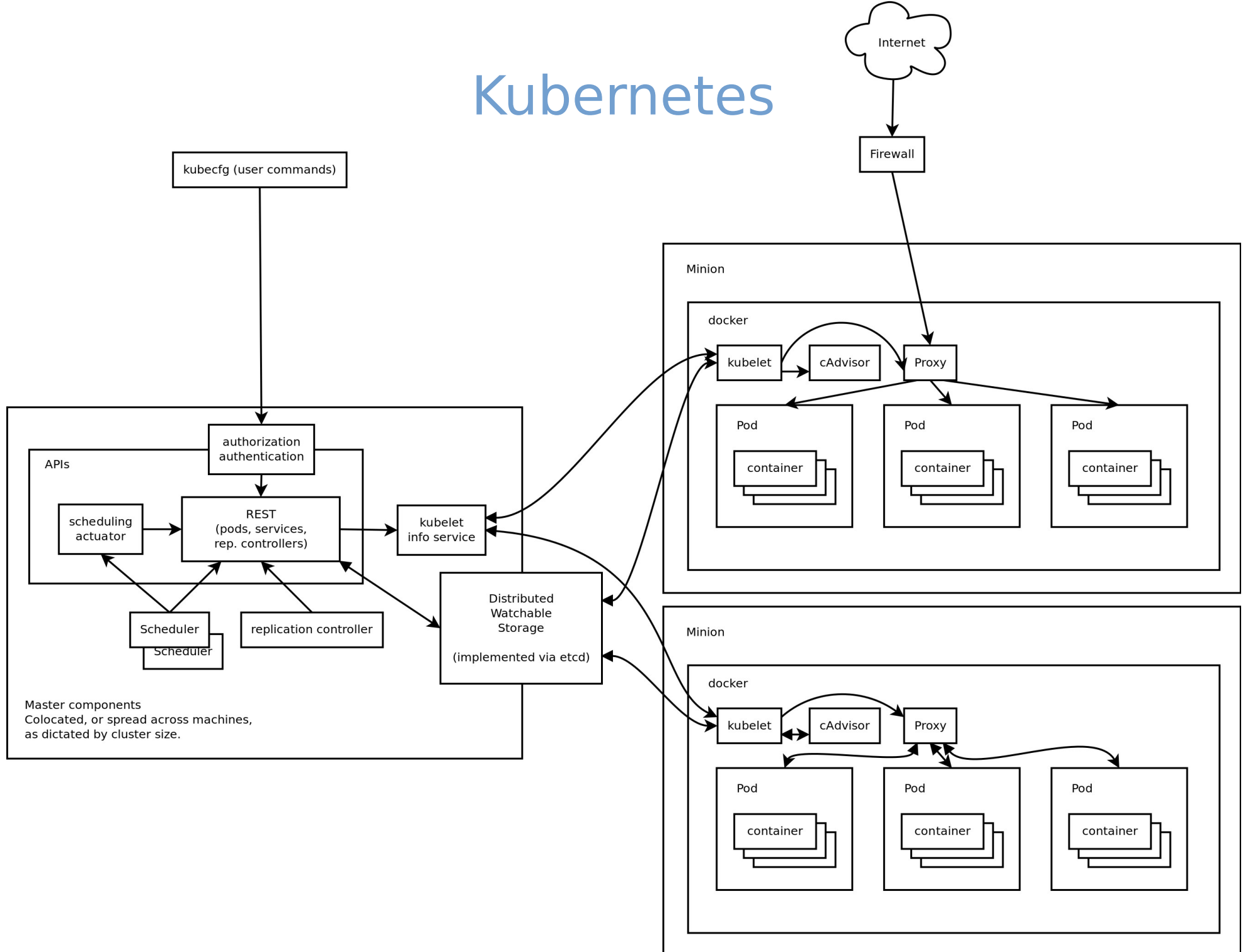
CoreOS

- scheduler
- HA
- dependencies

https://coreos.com/docs/launching-containers/launching/launching-containers-fleet/

Kubernetes

- https://github.com/GoogleCloudPlatform/kubernetes
- Advanced cluster manager (more than 1k hosts is a fit)
- Architecture:
    - master
    - minion
    - pod
    - replication controller
    - label

Kubernetes

SmartStack

- automated service discovery and registration framework
- ideal for SOA architectures
- ideal for continuous integration & delivery
- solves "works on my machine" problem

SmartStack

- automated service discovery and registration framework
- ideal for SOA architectures
- ideal for continuous integration & delivery
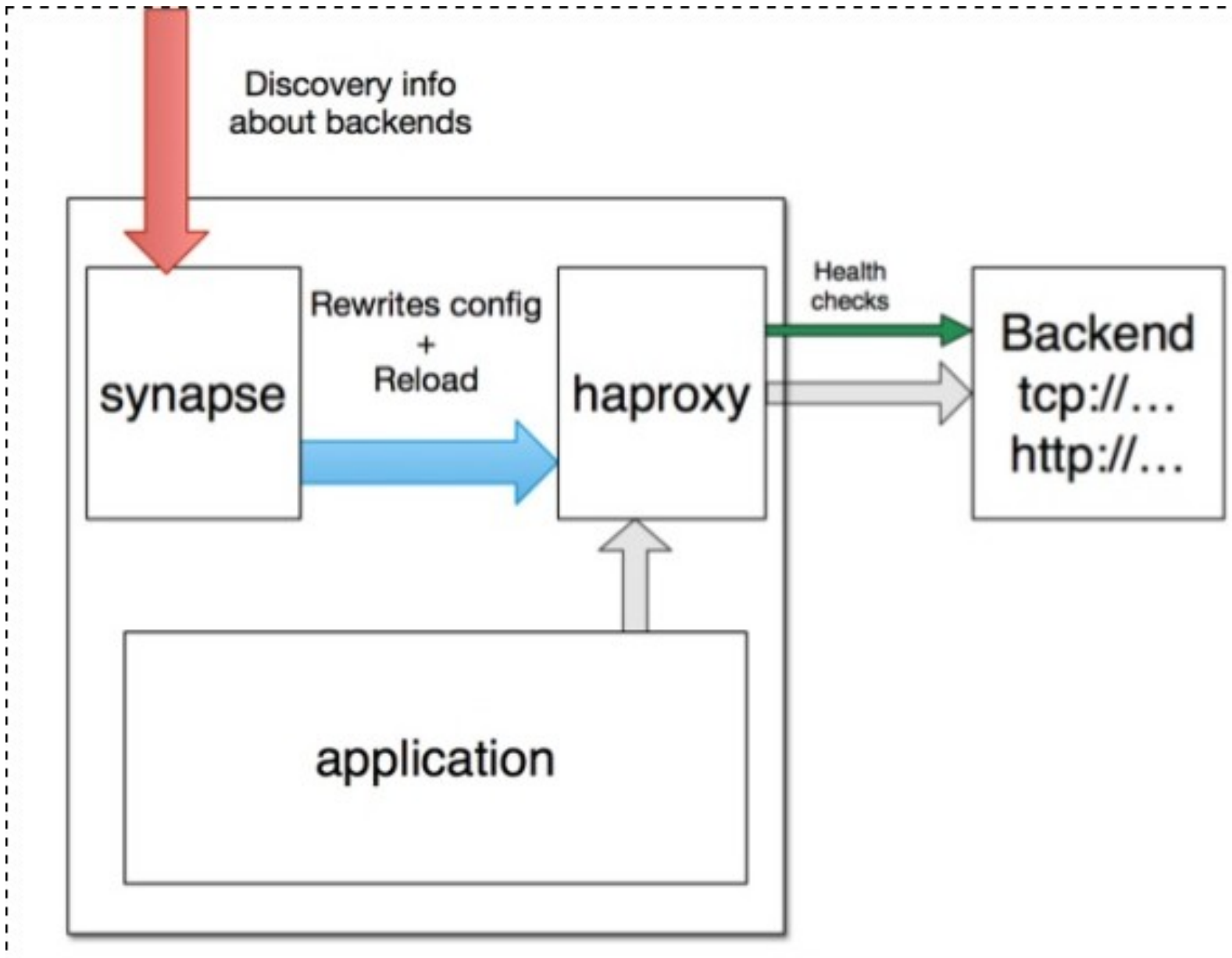- solves "works on my machine" problem

haproxy + nerve + synapse + zookeper = smartstack

SmartStack

Synapse

- discovery service (via zookeeper or etcd)
- installed on every node
- writes haproxy configuration
- application doesn't have to be aware of this
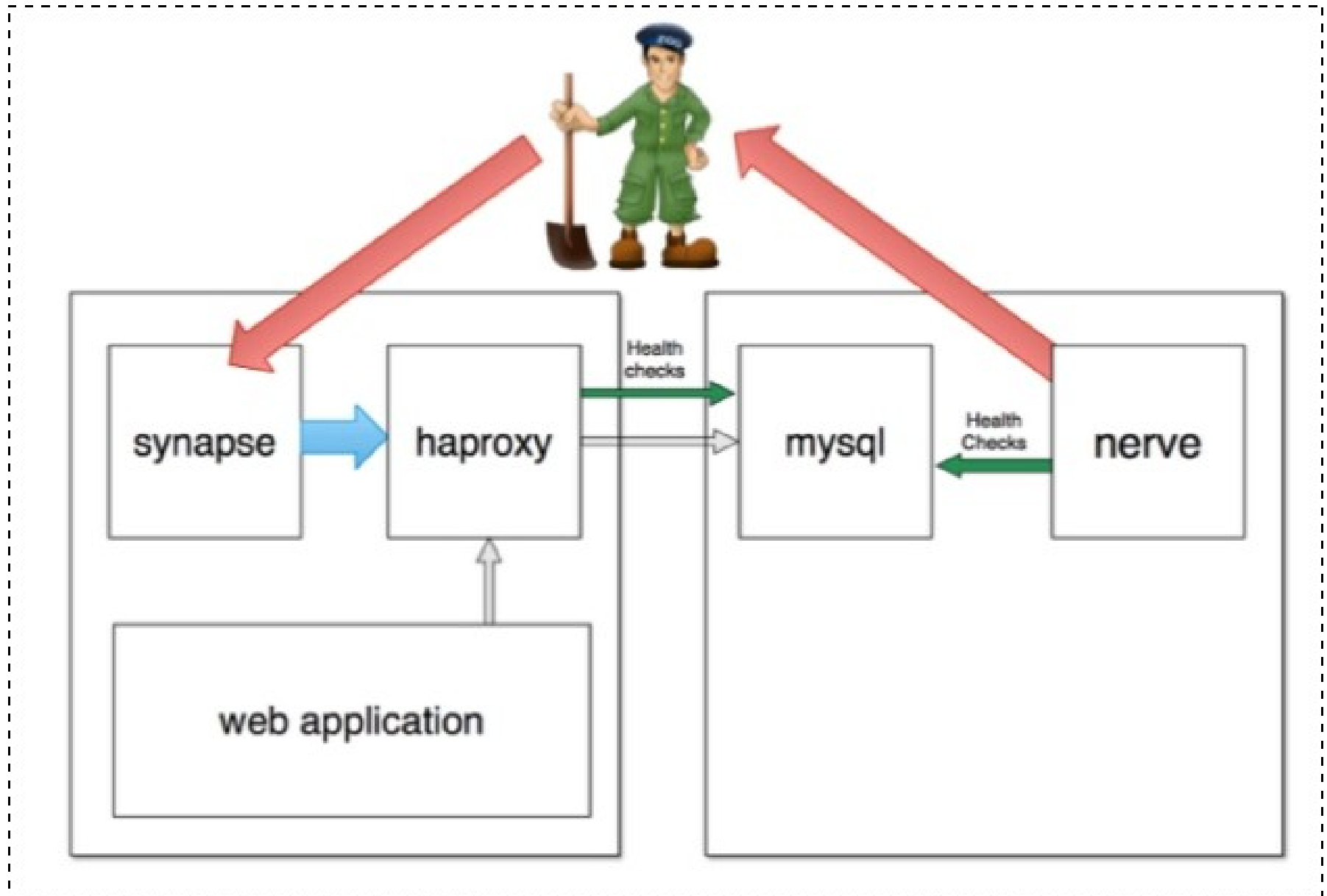- works same on bare / VM / docker

SmartStack

SmartStack

Nerve
- health checks (pluggable)
- register service info to zookeper (or etcd)

SmartStack

Summary

Summary

w/ Docker Sky is the limit!

Freenode #docker
Krk DevOPS meetups (http://www.meetup.com/Krakow-DevOps/)
https://github.com/docker/docker

sources?

- docker.io documentation
- dockerbook.com
- slideshare!
- and some experience ;)

Thank you :)

Orchestrating Docker containers at scale

Maciej Lasyk

Jesień Linuksowa 2014

2014-11-09, Szczyrk

http://maciej.lasyk.info

maciej@lasyk.info

@docent-net