Kubernetes on ARM64
Raspberry PI 4 Kubernetes cloud for a few Euros!

FOSDEM

Jean-Frederic Clere

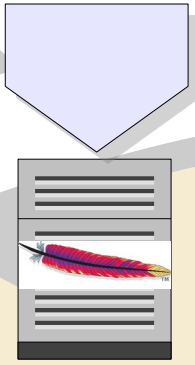
@jfclere

Who am I?

Jean-Frederic Clere

- Red Hat
- @JFCLERE email / twitter / slideshare ...

Structure of the RPI kubernetes cloud

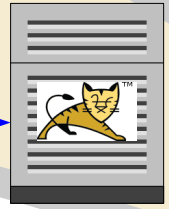


Infra: jfcportal

Infra stuff:
HTTPD mod_proxy
dhcp server
dns and registry
Hardware clock server

Master: master

Node: green



Node: blue

Kubernetes stuff:
1 master 2 nodes
Weave plugin network
Pods running tomcat images

RPI4 kernel

Use git repo:

https://github.com/raspberrypi/linux

CrossCompile on a laptop

make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu-

Build for RPI4 (and fedora/kubernetes)

https://github.com/jfclere/RPI4-Fedora30/blob/master/bcm2711_defconfig.patch

Build

make bcm2711_defconfig

make Image modules dtbs

Prepare SDcard:

Arm-image-installer: Image / ssh key / resizefs / selinux / relabel / (use fedora30 image)

arm-image-installer --image=/home/jfclere/Downloads/Fedora-Workstation-30-1.2.aarch64.raw.xz --target=rpi3 --media=/dev/mmcblk0 --norootpass --addkey=/home/jfclere/.ssh/id_rsa.pub --resizefs --relabel --selinux=ON

RPI4 kernel install

SDcard 3 partitions

Boot: vfat

Initrd: ext4

Root: ext4 (lvm2 / xfs since fedora31, mount more complex)

Copy the RPI4 files in the boot

copy the fixup4 and start4 files in boot. (from raspbian image and other in boot)

https://github.com/jfclere/RPI4-Fedora30/tree/master/boot

Copy the image

Copy arch/arm64/boot/Image to boot/kernel8.img

Install the modules and symbols in RPI4 root

make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- INSTALL_MOD_PATH=/run/media/jfclere/1a17da7a-d604-4e51-983c-e86d06d995e11 INSTALL_PATH=/run/media/jfclere/1a17da7a-d604-4e51-983c-e86d06d995e11 install modules_install

RPI4 WIFI board

Boot RPI4 and ssh to it. (nmap and ethernet cable)

Install firmware

cd /usr/lib/firmware/brcm; wget https://raw.githubusercontent.com/RPi-Distro/firmware-nonfree/master/brcm/brcmfmac43455-sdio.txt

Install tools:

NetworkManager-tui etc

Configure wifi and use it.

(note: MAC_ADDRESS_RANDOMIZATION=never)

INFRA for the demo

See http://jfclere.blogspot.com/ (if no internet connection you need hardclock/ dhcp / dns)

Docker image for ARM64

Easy centos7 or **openjdk:8-jre-alpine** based

Just need docker/podman on RPI

Maven and Java etc but all is there on fedora30

Need multi platform images:

- Manifest-tool: manifest-tool push from-spec tomcat-demo.yaml
- Build on the platforms you have ARM64/ADM64

Multi platform images

```
on amd64:
docker build -t jfclere/tomcat-demo:amd64 .
docker login  docker.io -u jfclere
docker push jfclere/tomcat-demo:amd64

on aarch64: (arm64)
docker build -t jfclere/tomcat-demo:aarch64 .
docker push jfclere/tomcat-demo:aarch64

See tomcat-demo.yaml:
image: jfclere/tomcat-demo:latest
manifests:
 -
   image: jfclere/tomcat-demo:amd64
   platform:
     architecture: amd64
     os: linux
...
 -
   image: jfclere/tomcat-demo:aarch64
   platform:
     architecture: arm64
     os: linux
```

on RPI:

docker build -t jfclere/tomcat-demo:aarch64 .

docker push jfclere/tomcat-demo:aarch64

See tomcat-demo.yaml:

```
   image: jfclere/tomcat-demo:aarch64
   platform:
    architecture: arm64
    os: linux
```

And use manifest-pool:

manifest-tool push from-spec tomcat-demo.yaml

Master

kubeadm reset -f

iptables -A INPUT -p tcp --dport 6443 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

iptables -A OUTPUT -p tcp --sport 6443 -m conntrack --ctstate ESTABLISHED -j ACCEPT

swapoff –all

kubeadm init

Start weave network using kubectl:

rm -rf $HOME/.kube

mkdir -p $HOME/.kube

scp root@master:/etc/kubernetes/admin.conf $HOME/.kube/config

kubectl apply -f weave-kube.yaml

On each node

kubeadm reset -f

Reset iptables

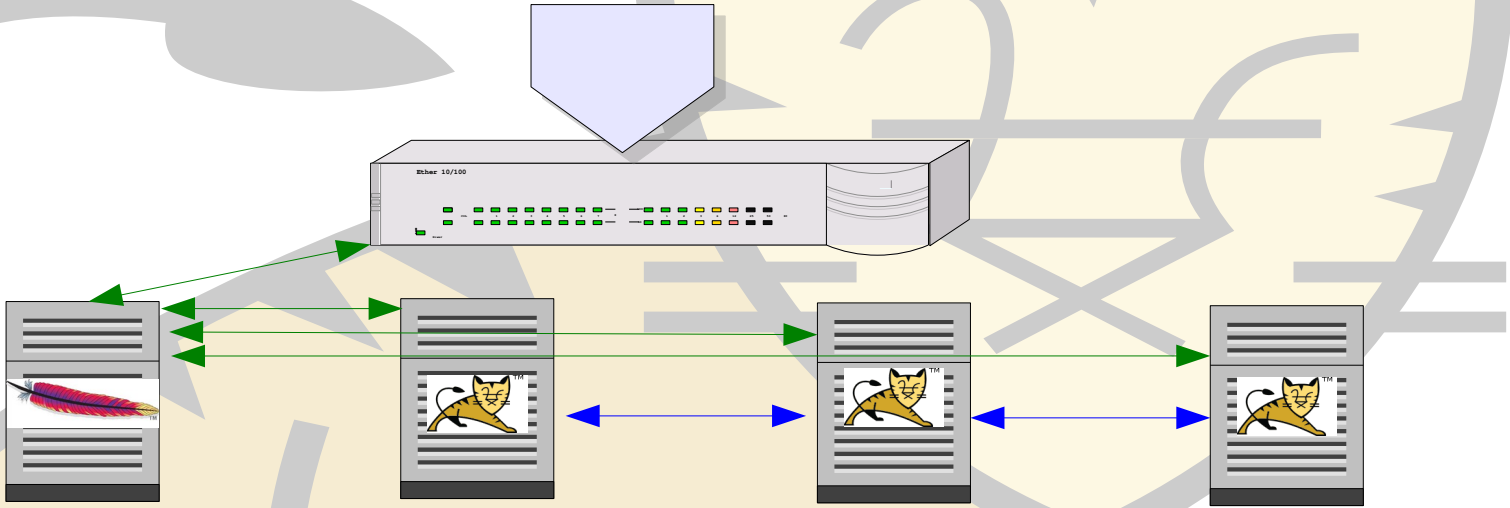
swapoff –all

kubeadm join –token=blabla (get it on master via: kubeadm token create --print-join-command)

Done when get nodes says ready.

kubectl get nodes

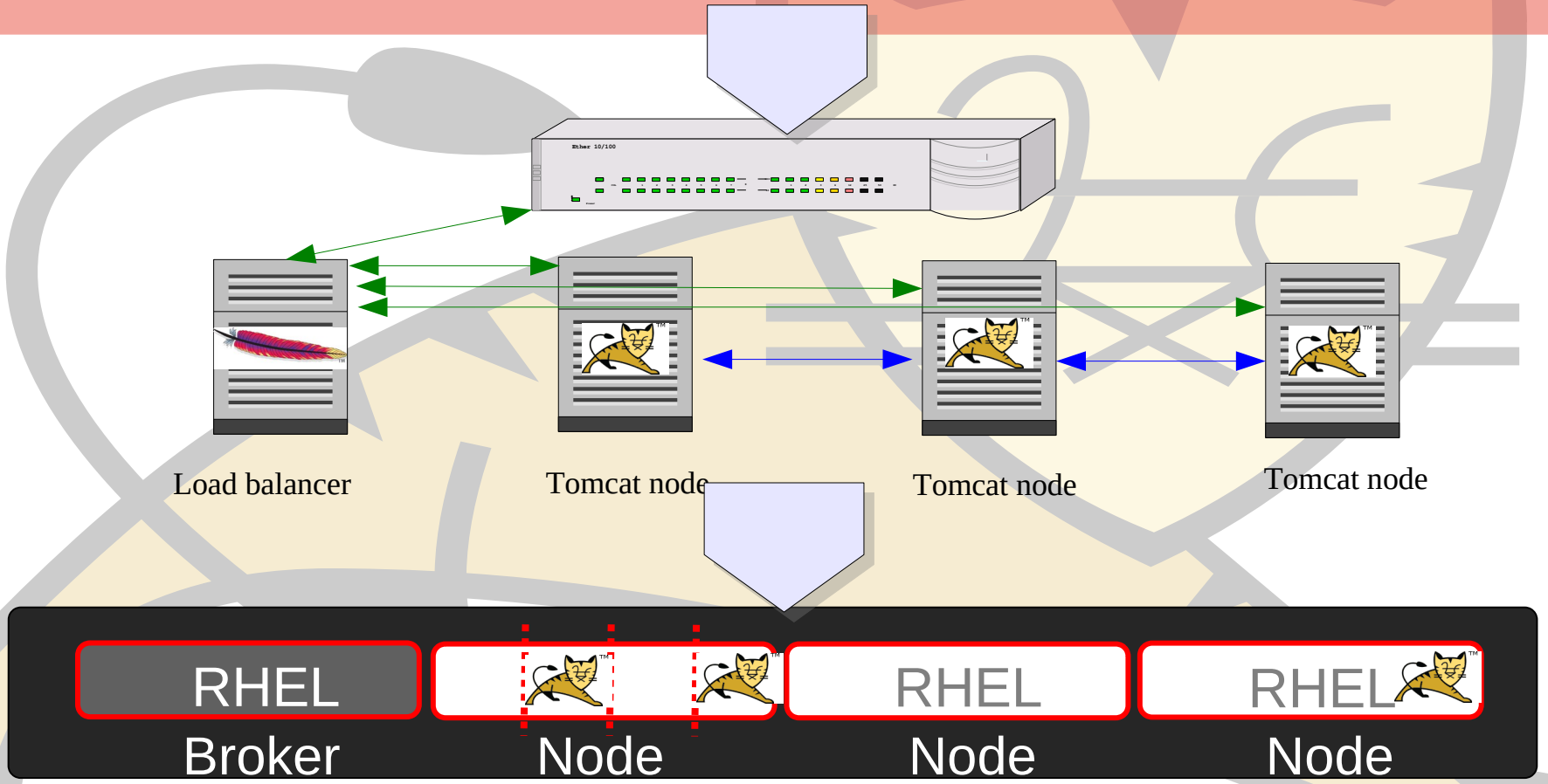
Tomcat Cluster



Session replication in a cluster

- HTTP/1.1
  - No transaction
  - No persistent connection
- Web App:
  - Using cookies to carry session ID
  - Store information in the session:
    - Shopping cart etc.
- Multi nodes and dynamic
  - Route request to right node
  - Replicate information

The move from cluster to cloud

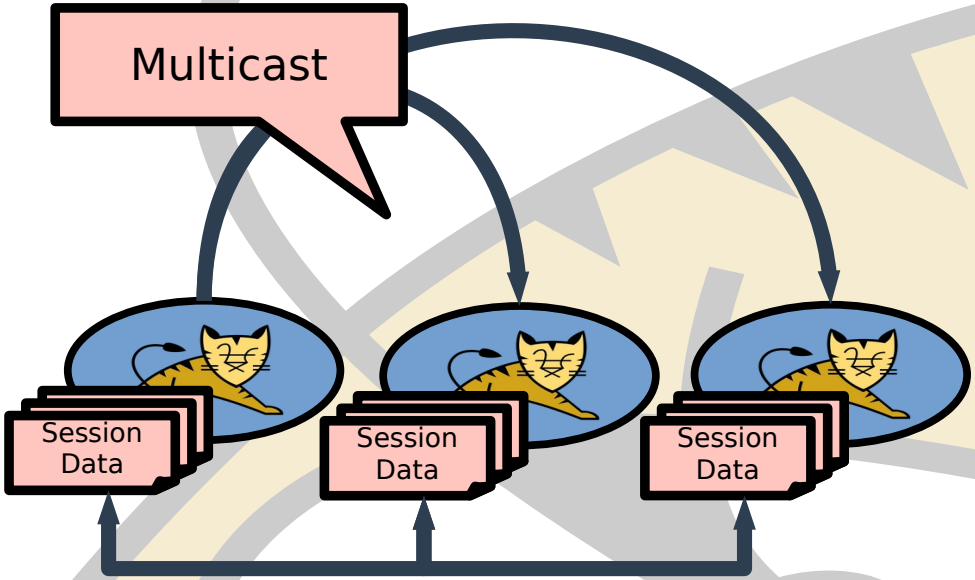


Problems for a cluster to cloud...

Tomcat cluster built-in solution

Peer discovery through multicast heartbeat messages

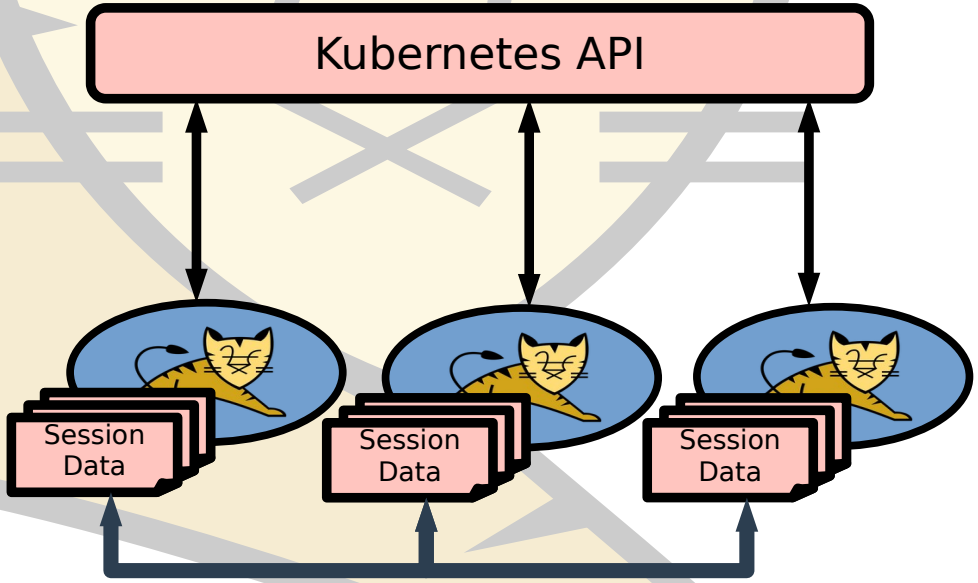
Does not work in a cloud environment



solution

Peer discovery through Kubernetes Downward API

Works in all kubernetes clouds



Solutions: KUBEPing

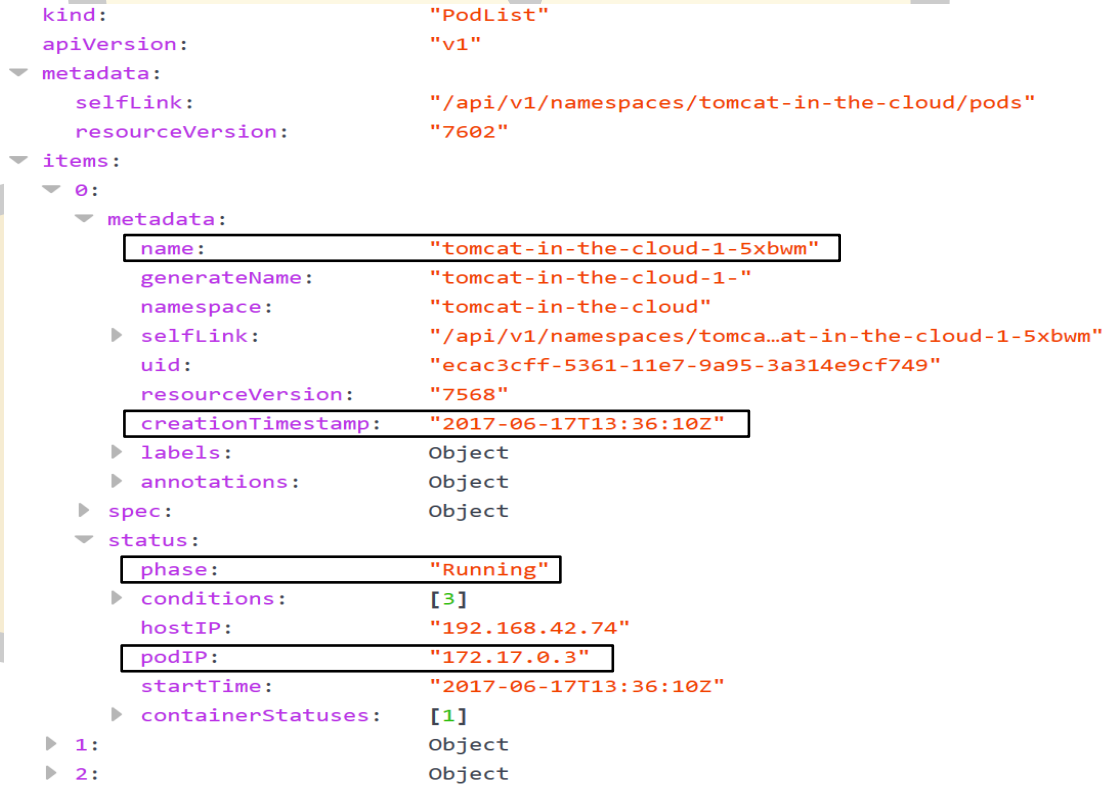
Tools for managing a Kubernetes cluster

Accessible from the pods within the cluster

GET /api/v1/namespaces/tomcat-in-the-cloud/pods

→ Return a JSON representation of all the pods in the cluster

→Requires permissions

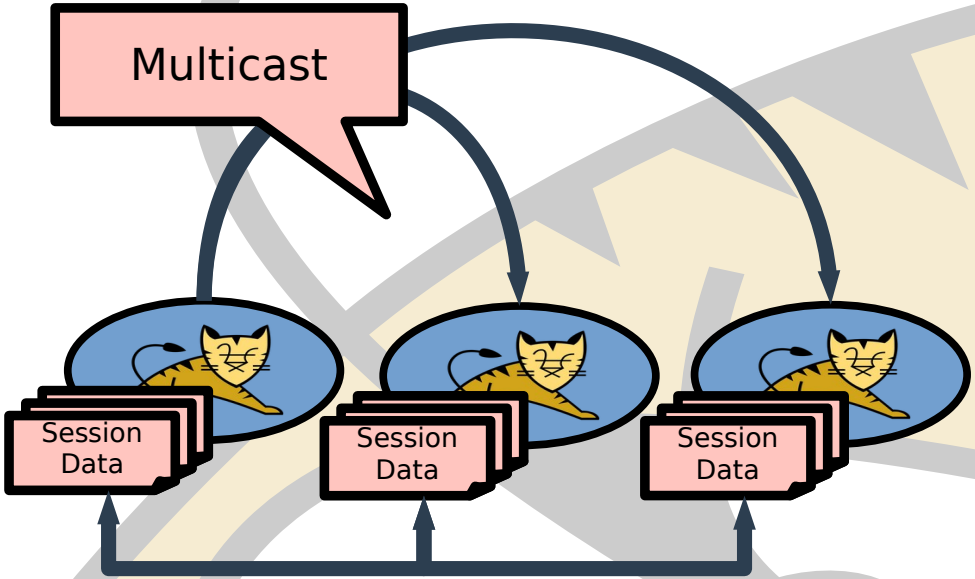


Architecture KUBEPing case

Tomcat cluster built-in solution

Peer discovery through multicast heartbeat messages

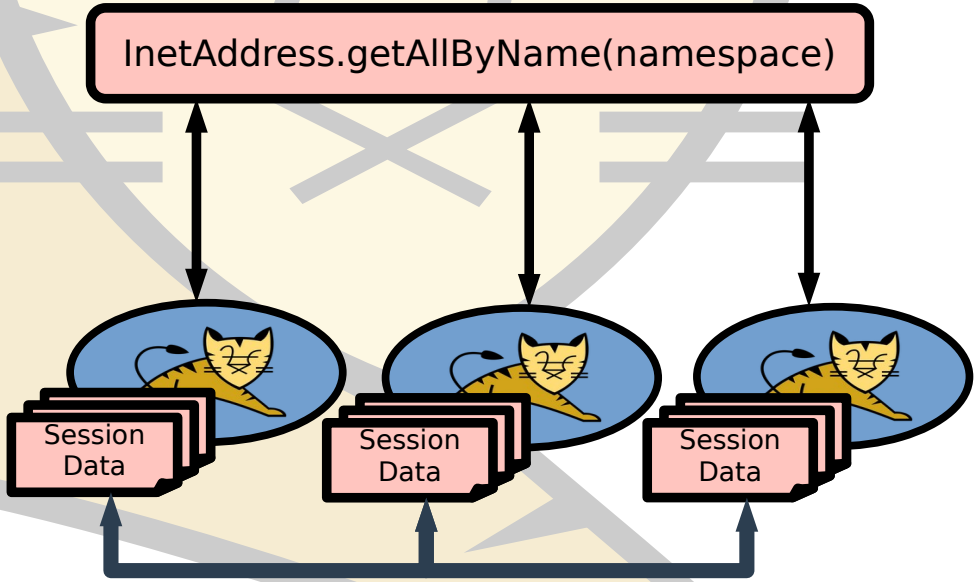
Does not work in a cloud environment



solution

Peer discovery through DNS lookup

Works in all kubernetes clouds



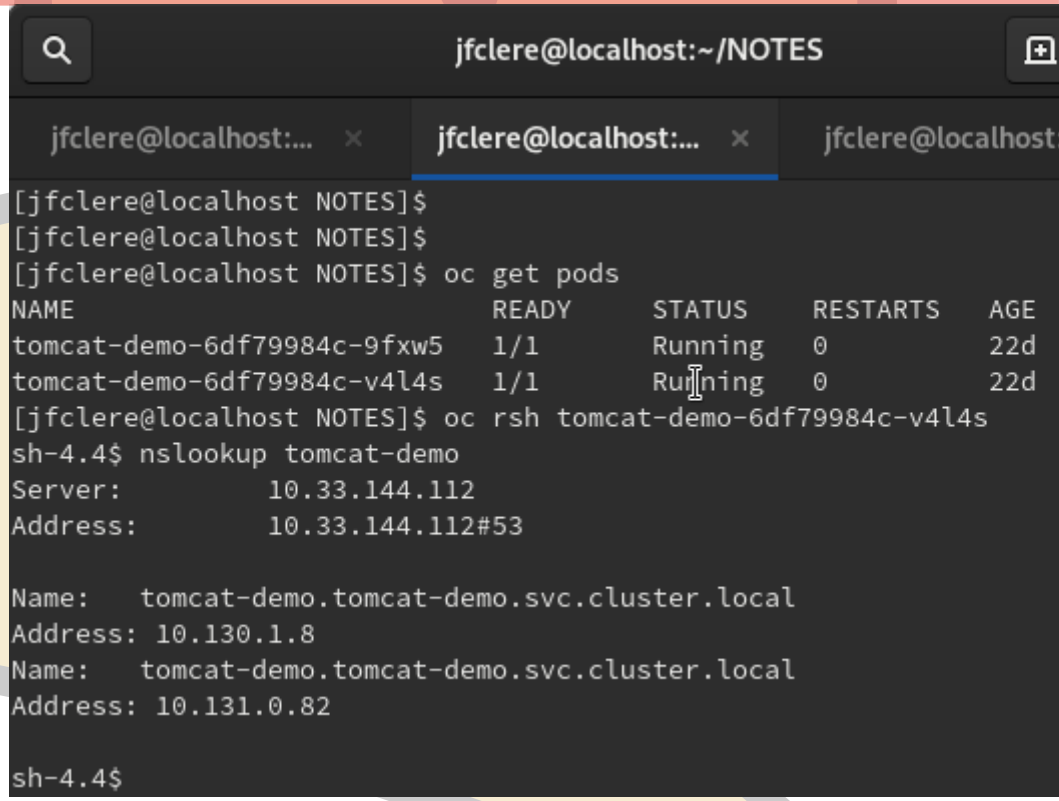
Solutions: DNSPing

nslookup name-space

Accessible from the pods within the cluster

InetAddress.getAllByName()

Needs a service.



THE DEMO

On JFCPORTAL (INFRA)

Httpd + mod_balancer (include proxy.conf in httpd.conf)

```
ProxyPass "/" "balancer://local"
ProxyPassReverse "/" "balancer://local"

<Proxy "balancer://local">
   BalancerMember "http://master:30306"
   BalancerMember "http://green:30306"
   BalancerMember "http://blue:30306"
</Proxy>
```

Docker to do the registry:

```
docker run -d -p 5000:5000 --restart=always --name registry registry:2
```

/etc/containers/registries.conf add:

registries = ['jfcportal:5000']

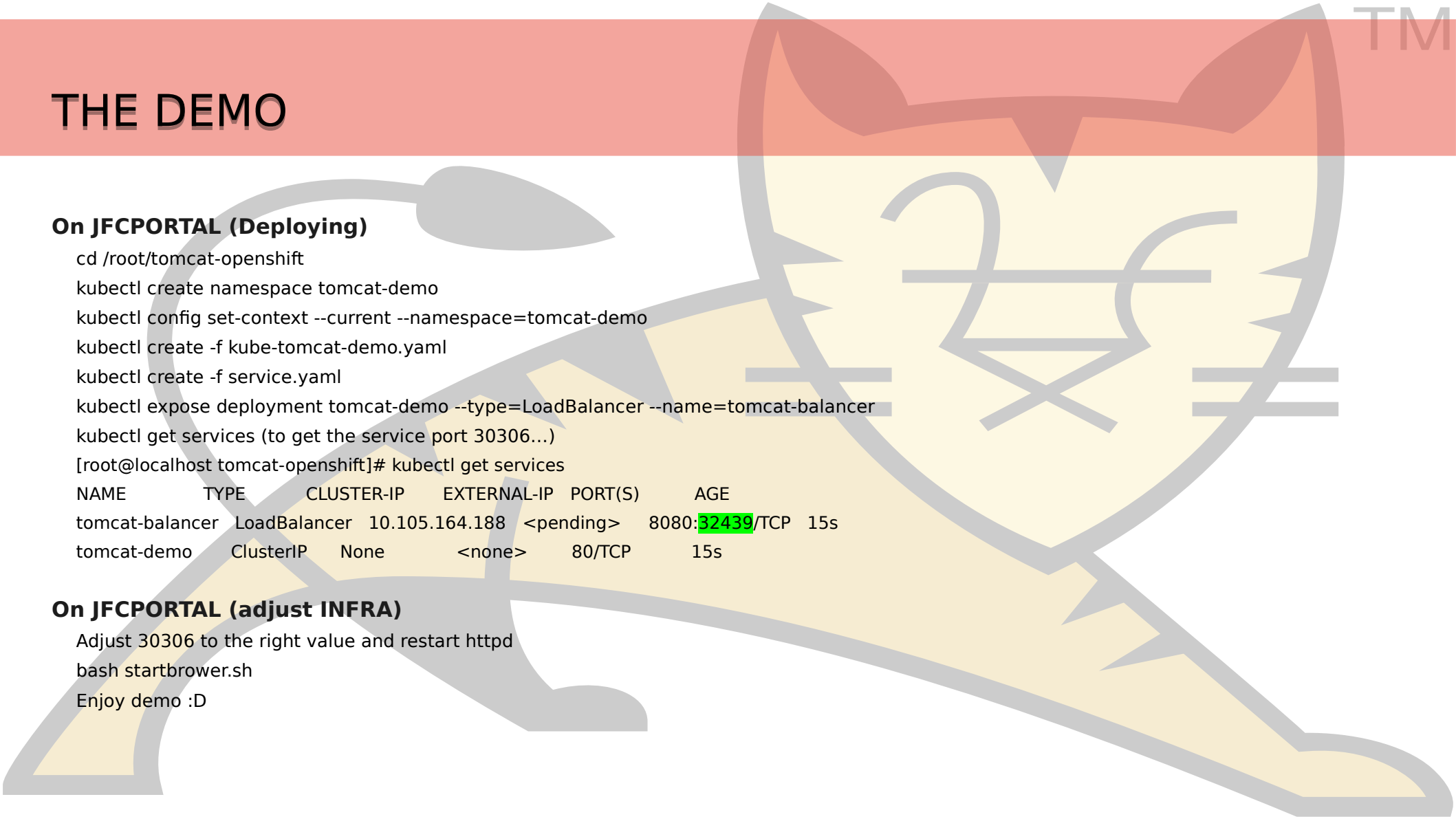
Build the image or get it and push it.

docker pull docker.io/jfclere/tomcat-demo

docker tag ID jfcportal:5000/tomcat-demo:2.2

docker push jfcportal:5000/tomcat-demo:2.2

THE DEMO



On JFCPORTAL (Deploying)

cd /root/tomcat-openshift

kubectl create namespace tomcat-demo

kubectl config set-context --current --namespace=tomcat-demo

kubectl create -f kube-tomcat-demo.yaml

kubectl create -f service.yaml

kubectl expose deployment tomcat-demo --type=LoadBalancer --name=tomcat-balancer

kubectl get services (to get the service port 30306…)

[root@localhost tomcat-openshift]# kubectl get services

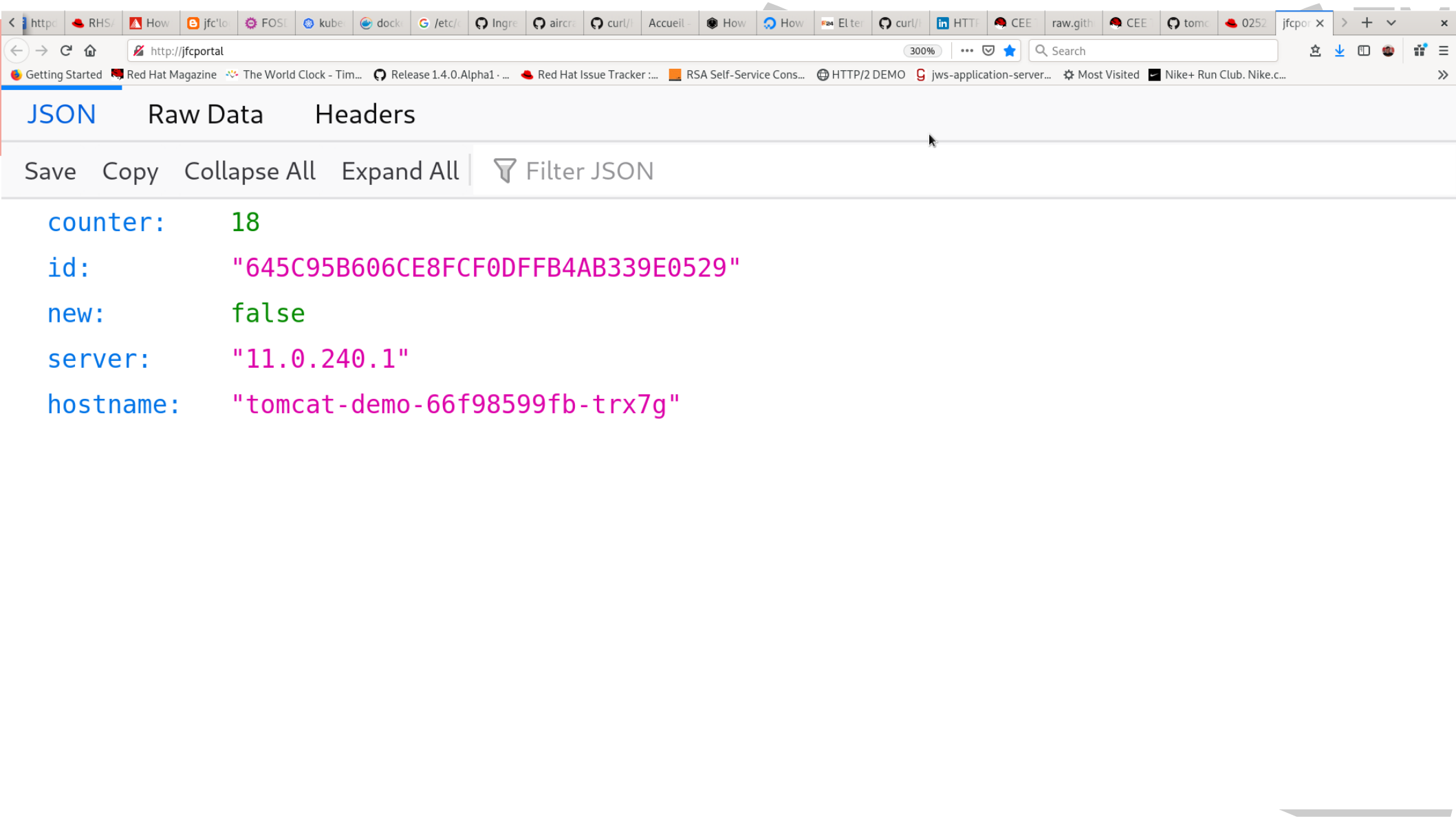
NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
tomcat-balancer	LoadBalancer	10.105.164.188	<pending>	8080:32439/TCP	15s
tomcat-demo	ClusterIP	None	<none>	80/TCP	15s

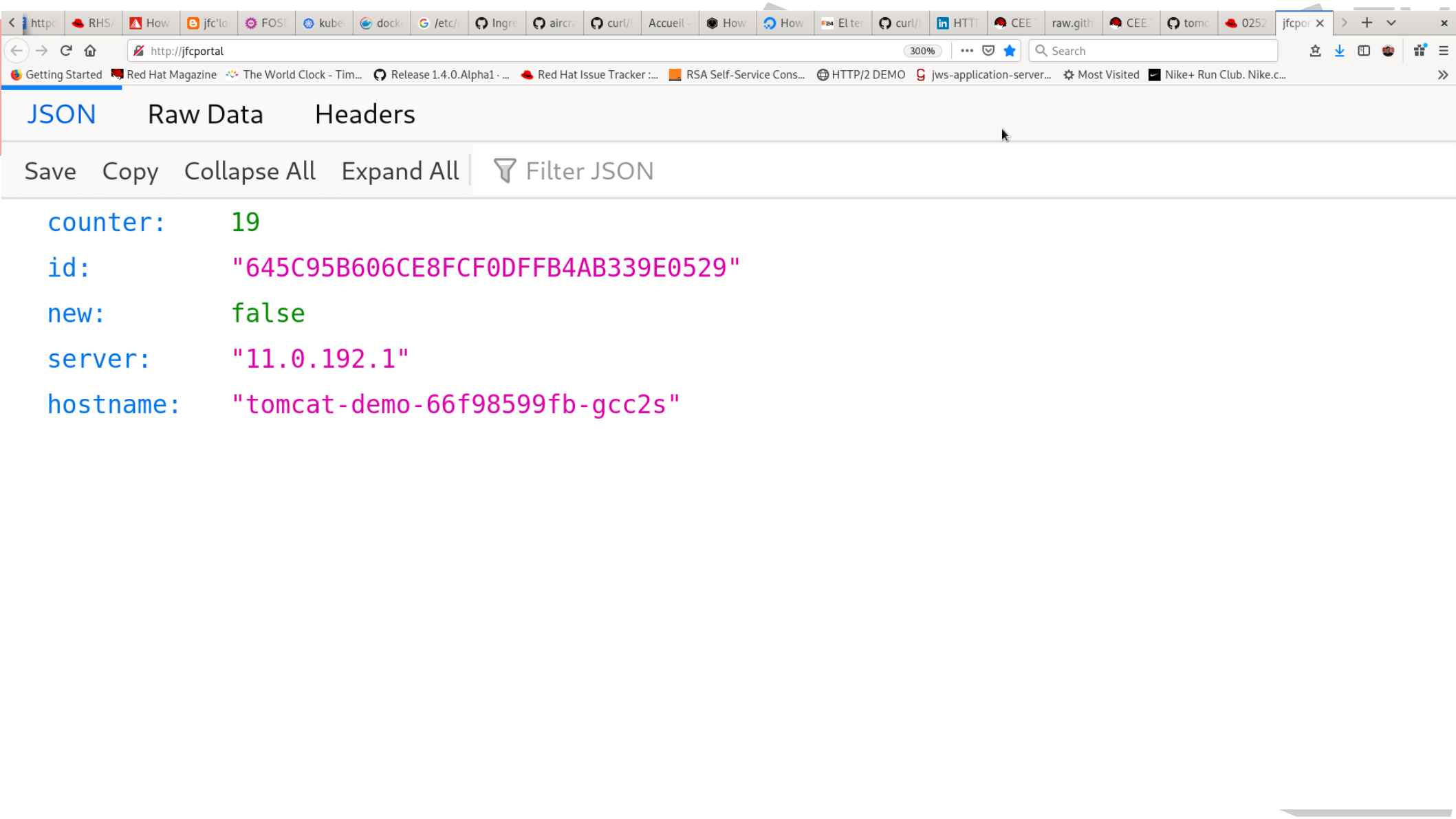
On JFCPORTAL (adjust INFRA)

Adjust 30306 to the right value and restart httpd

bash startbrower.sh

Enjoy demo :D


http://jfcportal
300%
Search
Getting Started
Red Hat Magazine
The World Clock - Tim...
Release 1.4.0.Alpha1 · ...
Red Hat Issue Tracker :...
RSA Self-Service Cons...
HTTP/2 DEMO
jws-application-server...
Most Visited
Nike+ Run Club. Nike.c...
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
counter: 18
id: "645C95B606CE8FCF0DFFB4AB339E0529"
new: false
server: "11.0.240.1"
hostname: "tomcat-demo-66f98599fb-trx7g"


http://jfcportal
300%
Search
Getting Started
Red Hat Magazine
The World Clock - Tim...
Release 1.4.0.Alpha1 · ...
Red Hat Issue Tracker :...
RSA Self-Service Cons...
HTTP/2 DEMO
jws-application-server...
Most Visited
Nike+ Run Club. Nike.c...
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
counter: 19
id: "645C95B606CE8FCF0DFFB4AB339E0529"
new: false
server: "11.0.192.1"
hostname: "tomcat-demo-66f98599fb-gcc2s"

Katacoda demo using DNSPing

https://katacoda.com/jfclere/scenarios/dnsping-tomcat

And the sources:

https://github.com/jfclere/intro-katacoda/tree/master/DNSPing-tomcat

Runs everywhere, but requires a service for DNS discovering.

Operator

What is a Kubernetes operator

kubernetes definition

Basically it automates the services, routes and build (S2I) process.

What do we have now

We have one written in GO (prototype)

S2I (source to image) just tooling :D

Katacoda demo using operator

Operator demo in Katacoda

And the sources:

https://github.com/jfclere/intro-katacoda/tree/master/war-katacoda

Every thing is created by the operator: pods, services etc.

Where we are

Main sites:

- https://github.com/jfclere/tomcat-openshift
- https://github.com/jfclere/kubernetes_f30_demo
- https://github.com/web-servers/tomcat-in-the-cloud
- https://github.com/jfclere/tomcatPI
- https://docs.openshift.com
- https://github.com/apache/tomcat
- tomcat : res/tomcat-maven
- DNSMembershipProvider / KubernetesMembershipProvider
- Tomcat operator and Source 2 Image (S2I)

THANK YOU

JEAN-FREDERIC CLERE

@jfclere

jfclere@gmail.com