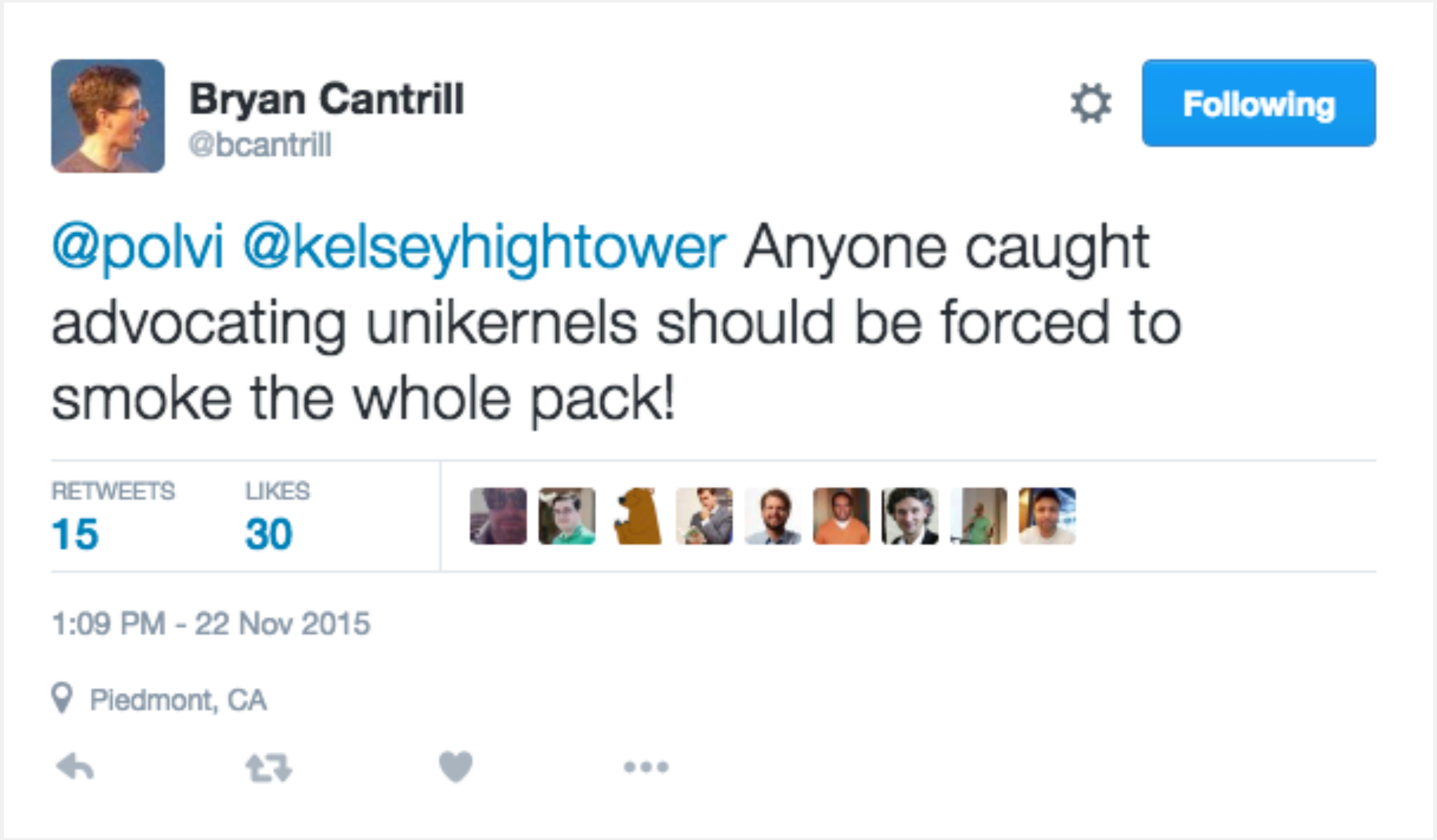
Roman Shaposhnik @rhatr, Pivotal Inc.

OSV: PROBABLY THE BEST OS FOR CLOUD WORKLOADS YOU'VE NEVER HEARD OF



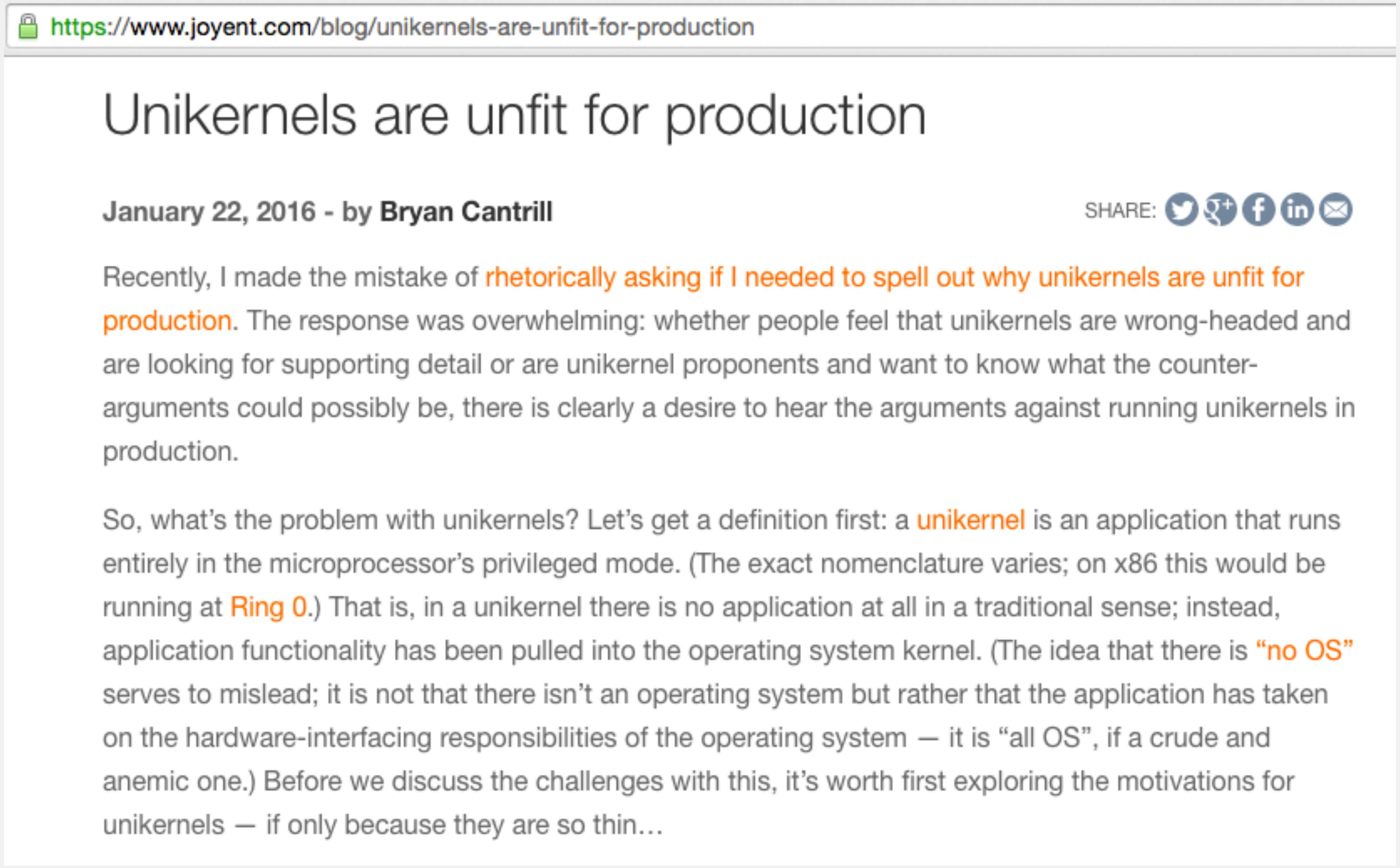
Bryan Cantrill
@bcantrill

@polvi @kelseyhightower Anyone caught advocating unikernels should be forced to smoke the whole pack!

RETWEETS 15 LIKES 30

1:09 PM - 22 Nov 2015

Piedmont, CA



https://www.joyent.com/blog/unikernels-are-unfit-for-production

Unikernels are unfit for production

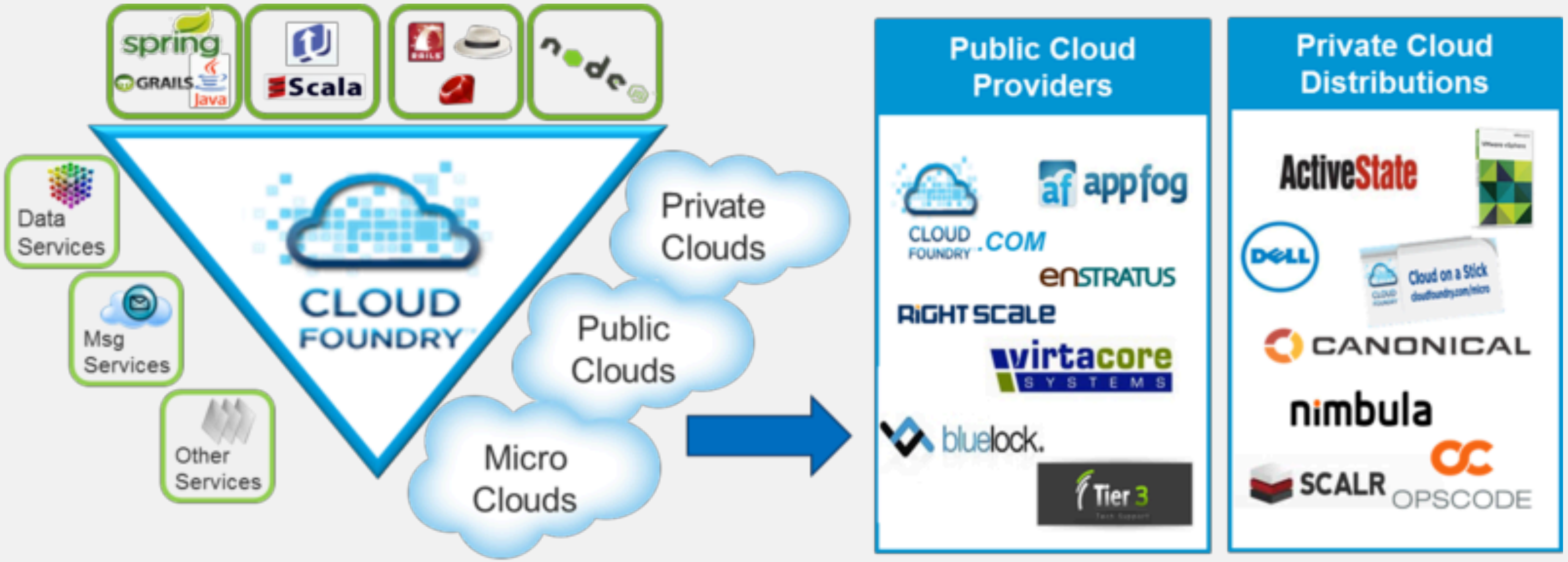
January 22, 2016 - by **Bryan Cantrill**

SHARE:

Recently, I made the mistake of rhetorically asking if I needed to spell out why unikernels are unfit for production. The response was overwhelming: whether people feel that unikernels are wrong-headed and are looking for supporting detail or are unikernel proponents and want to know what the counter-arguments could possibly be, there is clearly a desire to hear the arguments against running unikernels in production.

So, what's the problem with unikernels? Let's get a definition first: a unikernel is an application that runs entirely in the microprocessor's privileged mode. (The exact nomenclature varies; on x86 this would be running at Ring 0.) That is, in a unikernel there is no application at all in a traditional sense; instead, application functionality has been pulled into the operating system kernel. (The idea that there is "no OS" serves to mislead; it is not that there isn't an operating system but rather that the application has taken on the hardware-interfacing responsibilities of the operating system — it is "all OS", if a crude and anemic one.) Before we discuss the challenges with this, it's worth first exploring the motivations for unikernels — if only because they are so thin…

The raise of the PaaS: Cloud Foundry



The world's largest companies power their clouds with Cloud Foundry

Platinum

Pivotal
hp
IBM
EMC²
SAP
intel
vmware

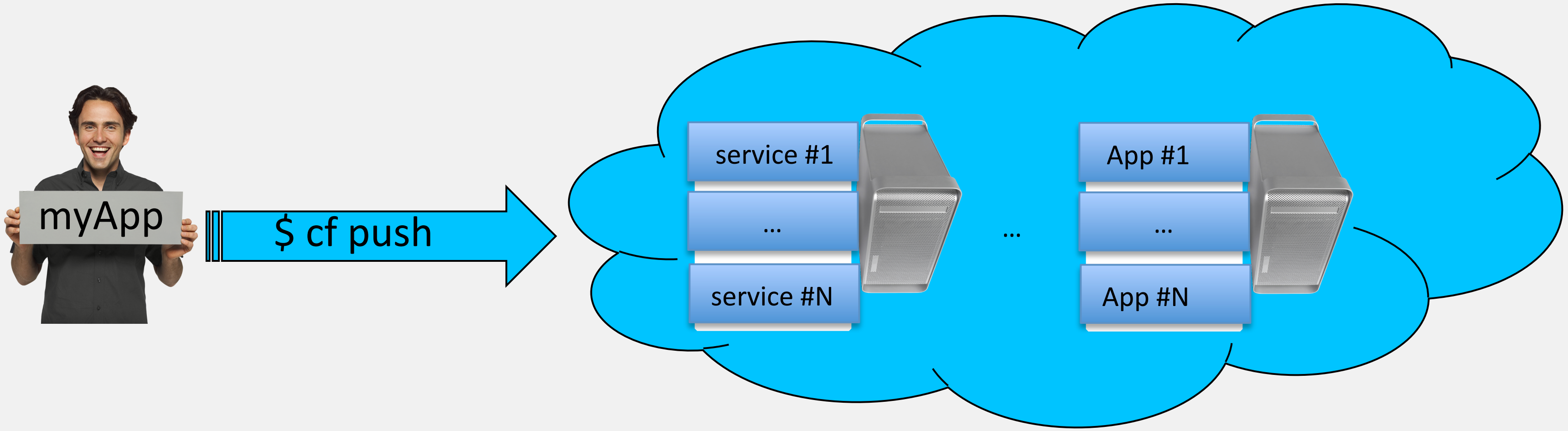
Gold

accenture High performance. Delivered.
Hortonworks
BNY MELLON
Capgemini
CenturyLink
swisscom
JPMorgan Chase & Co.
TELSTRA
HUAWEI
ERICSSON
ActiveState
NTT
GE
sas
verizon

Silver

MIMACOM
cloudsoft
citi
apigee
bluebox
Piston
canopy
CloudCredo
New Relic.
docker
redislabs
APPDYNAMICS
AZUL SYSTEMS
anynines
Stark&Wayne.
MoPaaS
ALTOROS
TOSHIBA
Akamai
FUJITSU
MIRANTIS
BRAINFREE BRIDGING QUALITY
mendix
BIARCA
Bloomberg

No, but seriously?



Cloud-native apps AKA 12factor.net

- Codebase
- Dependencies
- Config
- Backing services
- Build, deploy, run
- Stateless processes
- Port binding
- Concurrency
- Disposability
- Dev == prod
- Logs == streams
- Admin processes

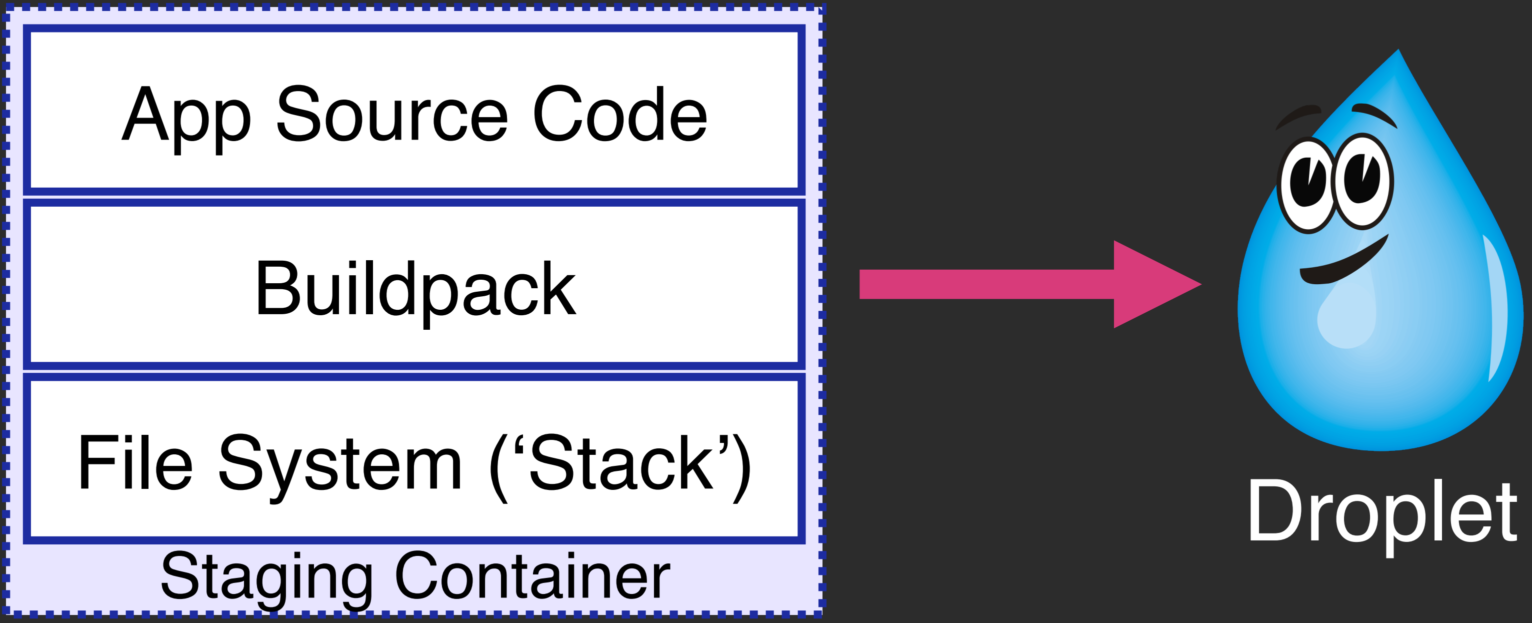
```
> cd /path/to/my/app
> tree
 .
├── README.md
├── app.groovy
├── application.properties
├── manifest.yml
```

```
> cat manifest.yml
---
applications:
- name: cf-spring
  memory: 512M
  instances: 3
  random-route: true
```

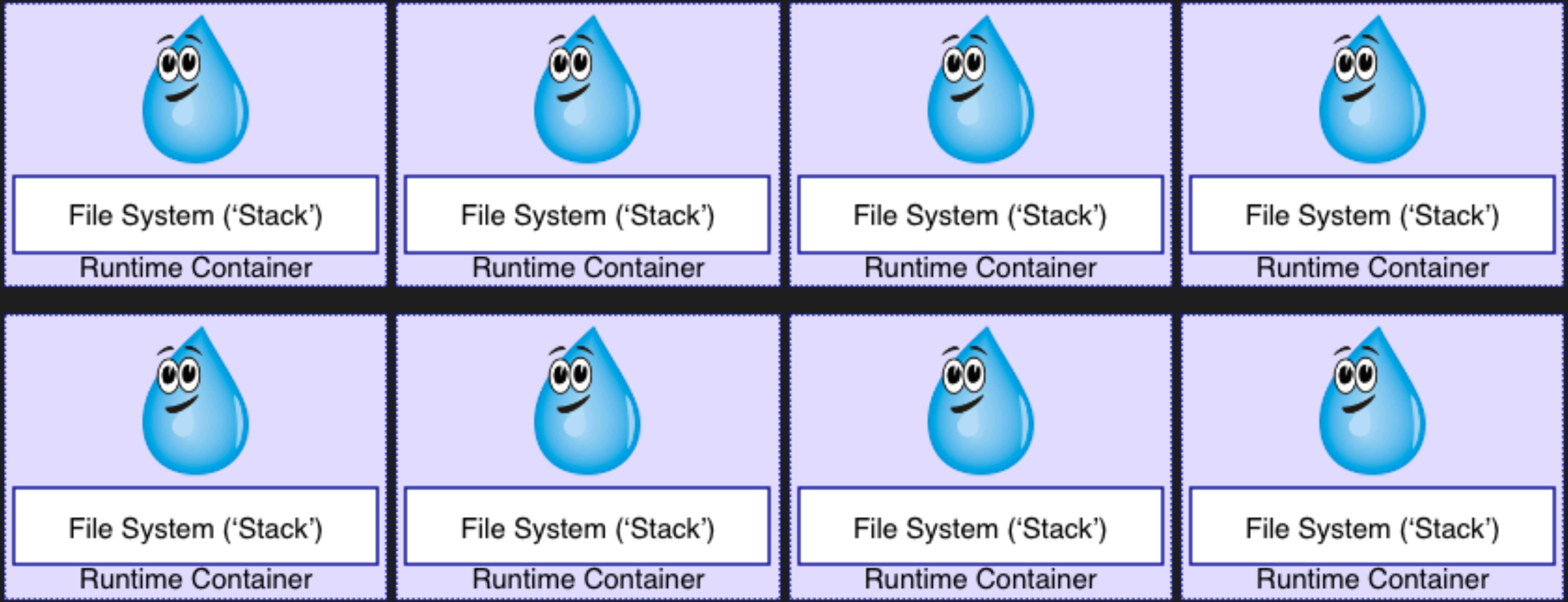
```
> cf push my-app

Using manifest file /Users/verney/workspace/cf-sample-app-spring/
manifest.yml



Creating app cf-spring in org pivot-jules / space test as
jules@verne.io...

OK
```

```
Uploading cf-spring...

Uploading app files from: /Users/vereny/workspace/cf-
sample-app-spring
Uploading 1M, 44 files
Done uploading
OK
```

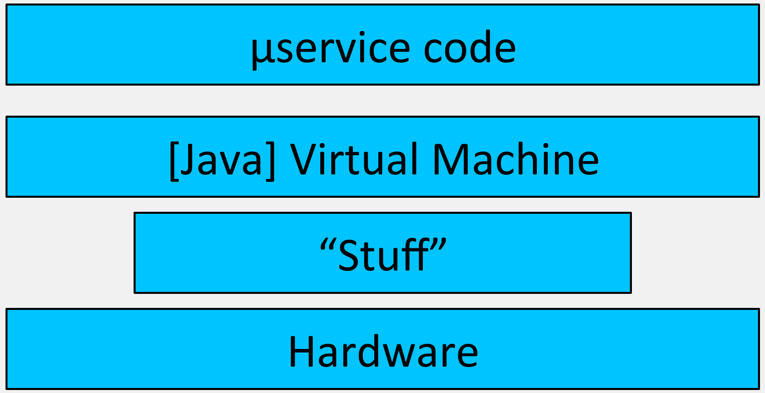
Droplets


App Source Code
Buildpack
File System (‘Stack’)
Staging Container
Droplet

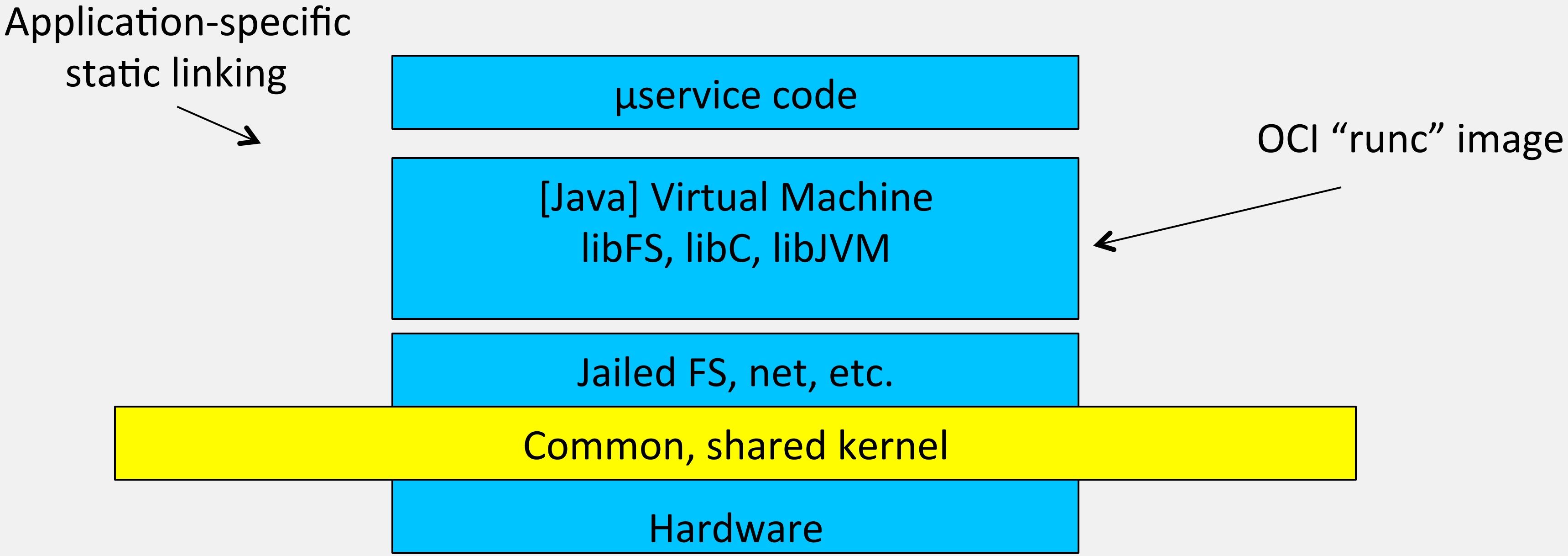
```
> cf scale my-app -i 8
```


File System ('Stack')
Runtime Container
File System ('Stack')
Runtime Container
File System ('Stack')
Runtime Container
File System ('Stack')
Runtime Container
File System ('Stack')
Runtime Container
File System ('Stack')
Runtime Container
File System ('Stack')
Runtime Container
File System ('Stack')
Runtime Container

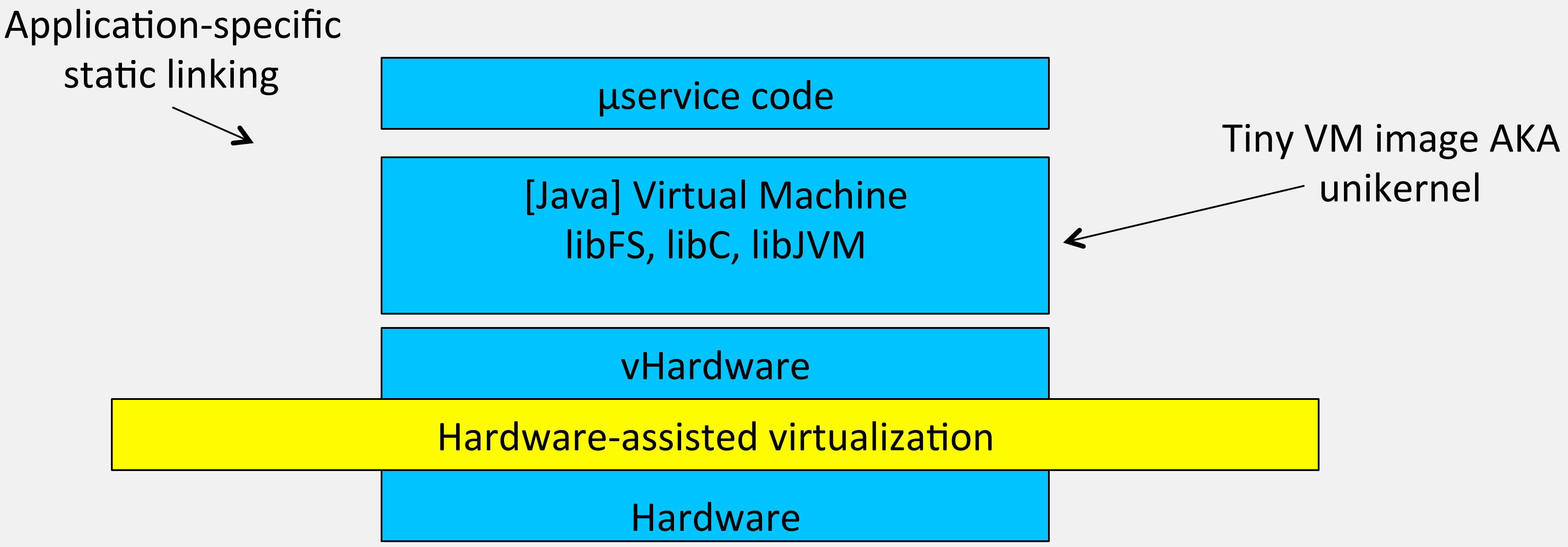
Anatomy of a droplet



How are we doing it today?



Is there a better way?


Application-specific static linking
µservice code
[Java] Virtual Machine
libFS, libC, libJVM
Tiny VM image AKA unikernel
vHardware
Hardware-assisted virtualization
Hardware

Unikernels

- “Unikernels: library operating systems for the cloud” came out in 2013
- A “library” operating system
- A kernel that can only support one process
- An ‘executable’ that needs virtualization to run
  - Qemu, VB, VMWare, Xen, Public Cloud

Anykernels

- Programming discipline for kernel code reuse
- “The Design and Implementation of the Anykernel and Rump Kernels” by Antti Kantee
- Capabilities
  – NetBSD filesystems as Linux processes
  – User-space TCP/IP stack
- Building blocks for… any kernels

What unikernels are available

- Mirage OS
  - Emerged from Xen, OCaml specific, research
- Clive
  - Go specific, Plan 9 lineage, research
- Rump Kernels (brought to you by A. Kantee)
  - Rumprun unikernel, “static linking” down to the kernel
- OSv

UniK: Unikernel Builds & Deployment

- An open source tool
  - https://github.com/emc-advanced-dev/unik
- A familiar Docker-like CLI
- Abstracts away details of virtualization backends
- Integrates with Docker & Cloud Foundry
- Pluggable support for Unikernels
  - OSv & rump

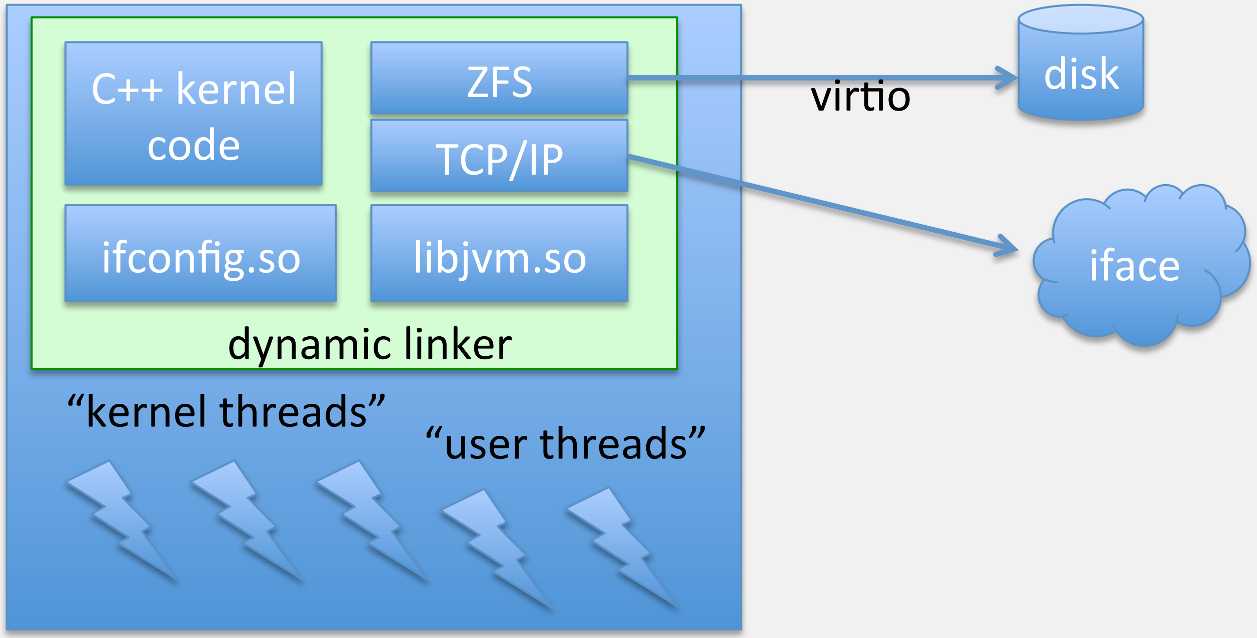
OSv from Cloudius Systems

- A unikernel for “POSIX” and memory managed platforms (JVM, Go, Lua)
- Anykernel’ish
  - E.g. ZFS
- Runs on top of KVM, Xen, VirtualBox, VMWare
- Looks like an app to the host OS
- Small, fast and easy to manage at scale

OSv manifesto

- Run existing Linux applications
- Run existing Linux applications faster
- Make boot time ~= exec time
- Explore APIs beyond POSIX
- Leverage memory managed platforms (JVM, Go)
- Stay open

What's inside?



single address space in "kernel mode"

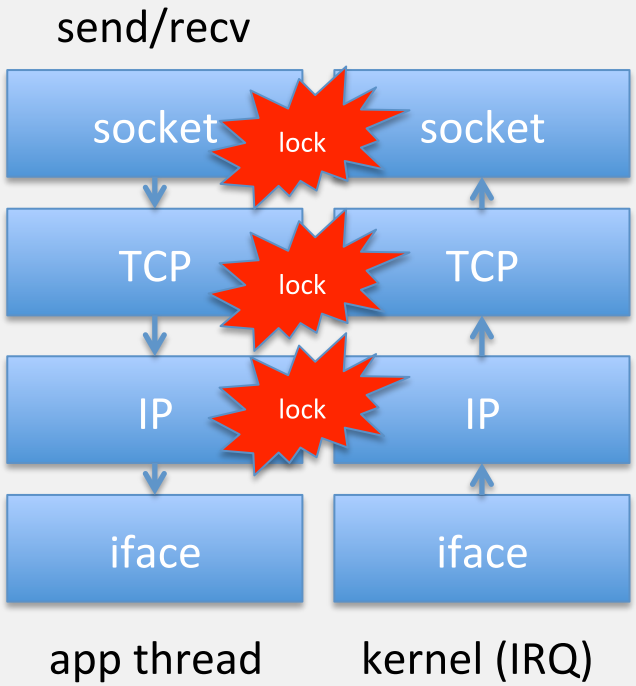
Anything it can’t do?

- A 100% replacement for a Linux kernel
  - No fork()ing
- No process isolation
- The least amount of device drivers ever

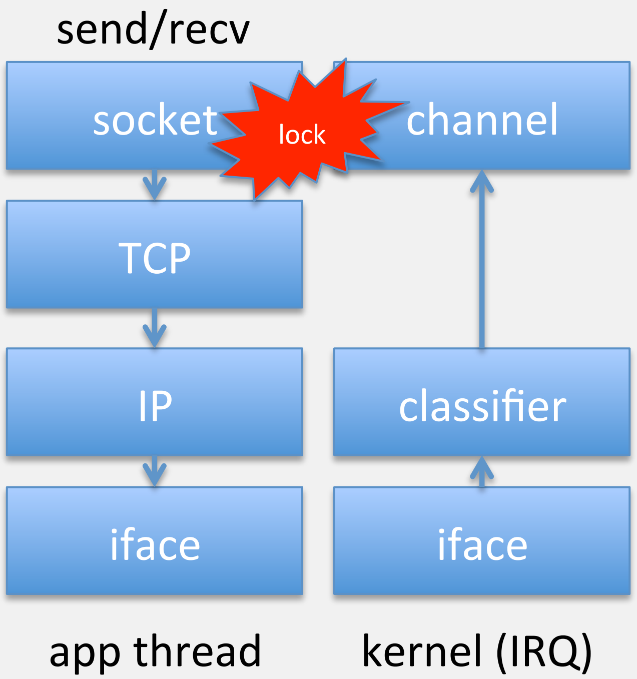
Virtualization vs. performance

- Network-intensive apps:
  - unmodified: 25% gain in throughput
    47% decrease in latency
  - non-POSIX APIs use for Memcached:
    290% increase in performance
- Compute-intensive apps:
  - YMMV

Van Jacabson’s net channels

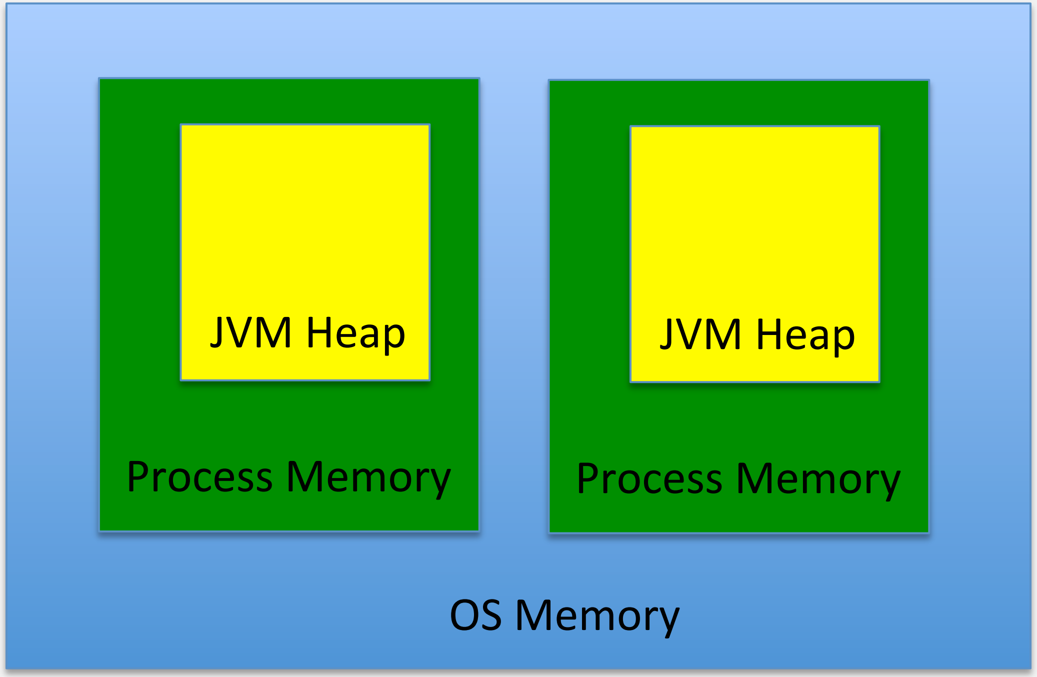


Traditional TCP/IP stack

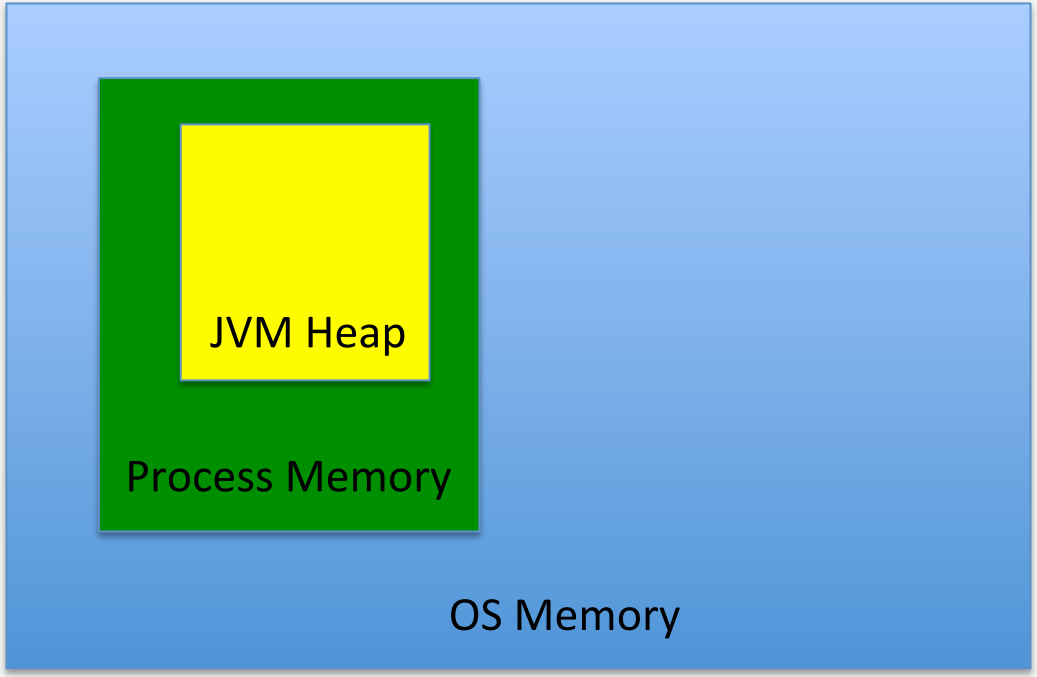


OSv TCP/IP stack

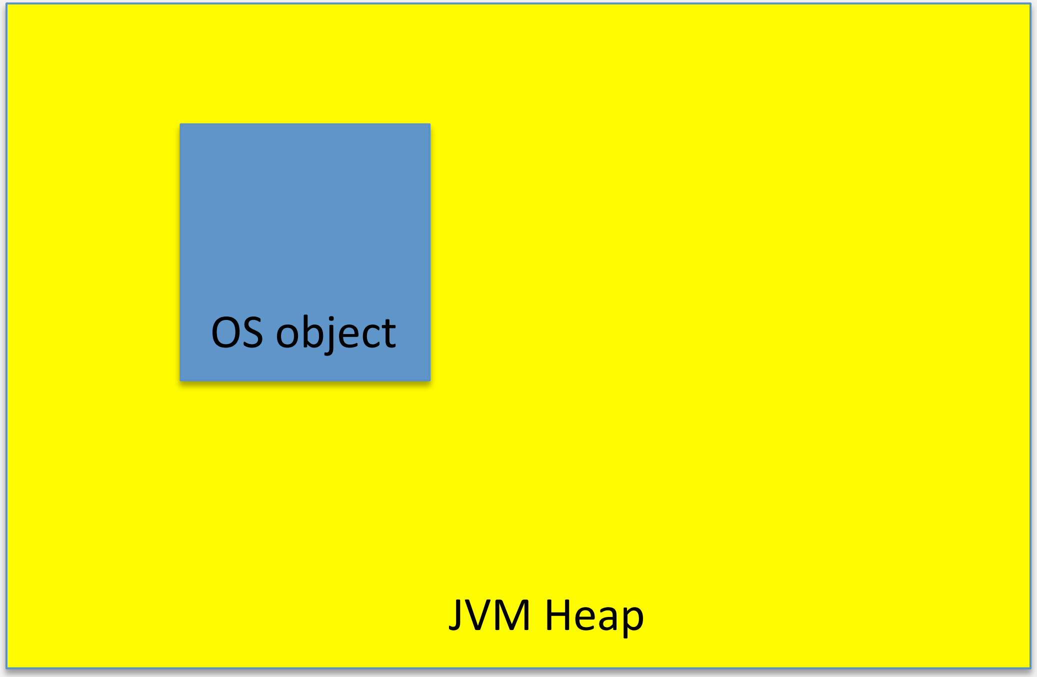
Memory management in UNIX



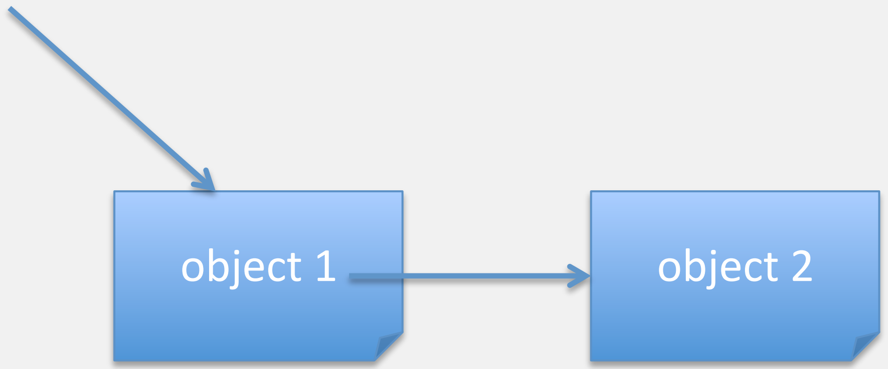
Memory management in OSv



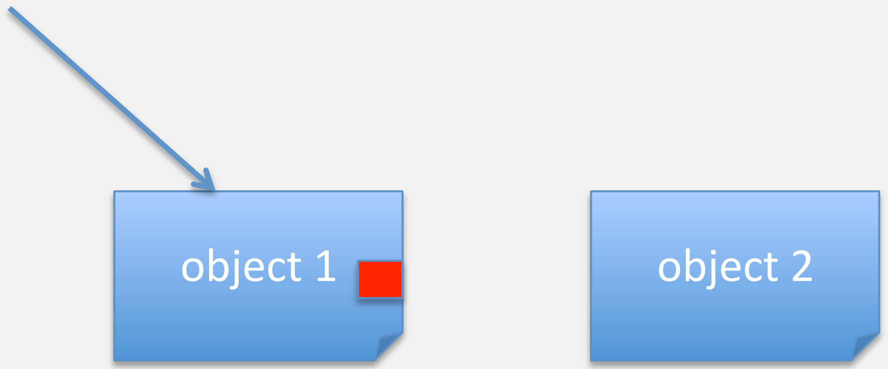
JVM balooning (no more -Xmx)



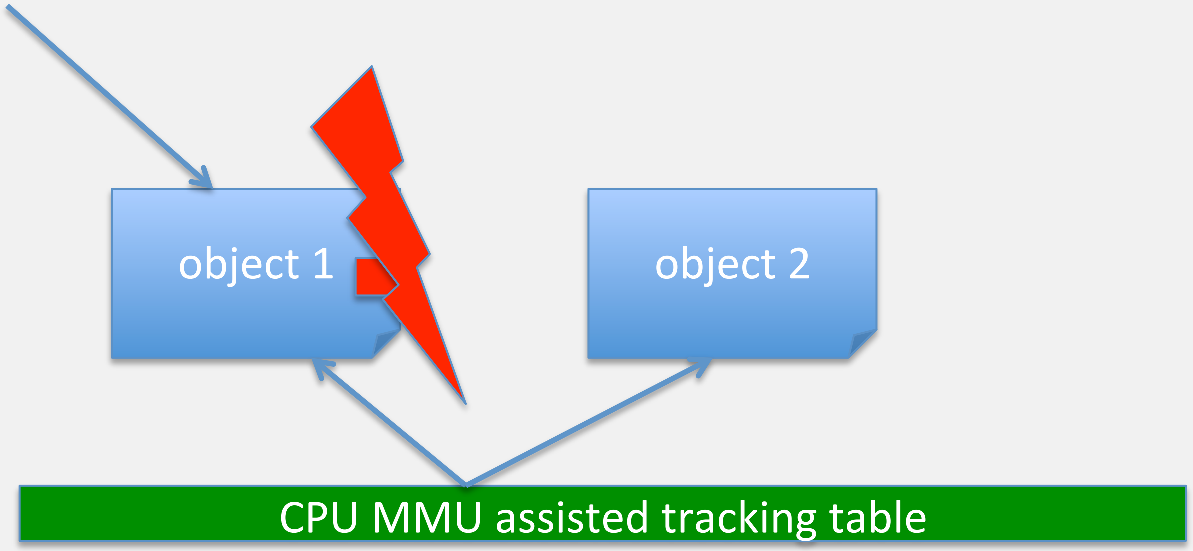
Turbo charging JVM GC



Turbo charging JVM GC



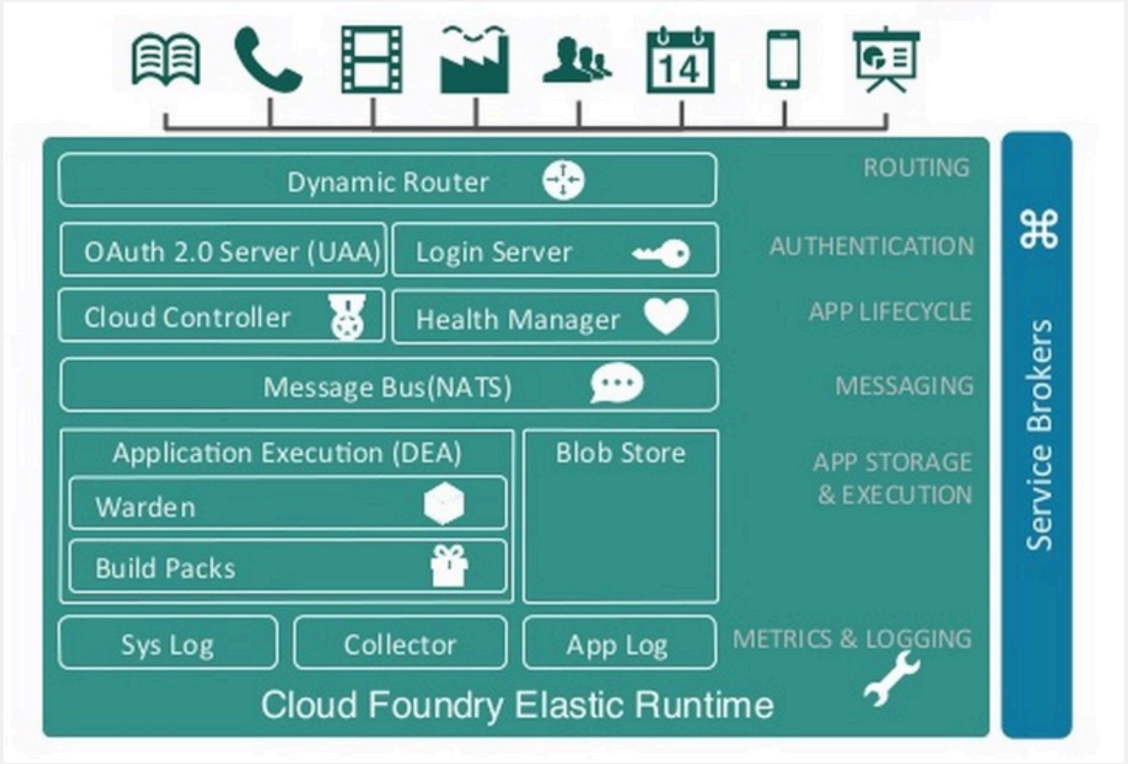
Turbo charging JVM GC



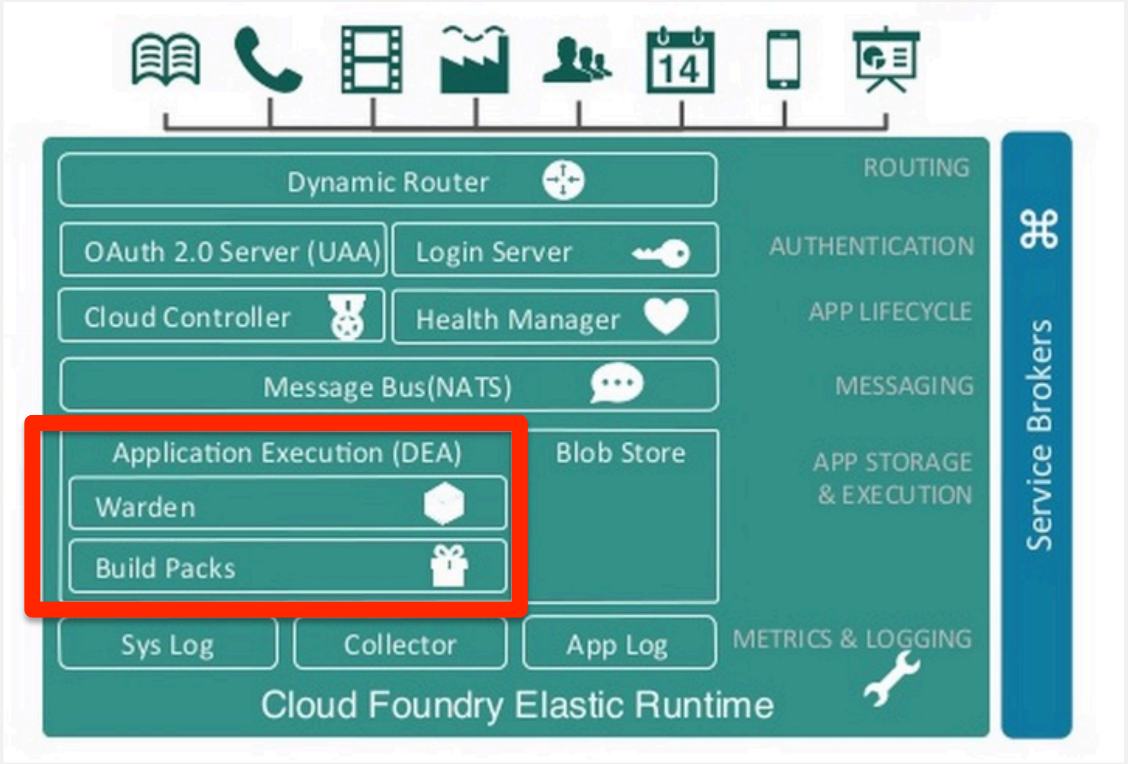
Why should it work this time?

- ~~Unikernels/exokernels back in ’90~~
- ~~JVM-on-bare-metal (Azul, BEA, etc.) back in ‘00~~
- Things they didn’t have back then
  - HW-assisted virtualization (KVM, XEN, etc.)
  - Elastic infrastructure oriented architectures
  - Cloud Foundry (PaaS)

No, really we need PaaS



No, really we need PaaS



Elastic, next generation datacenter

- Commodity, rack-provisioned Hardware
- JeOS (CoreOS, SmartOS, Xen+JeOS)
  - a glorified device driver: anything2virtio
  - optionally: a way to virtualize a “dom0” kernel
- Docker++ as the new ELF format
  - with either nokernel or unikernel inside
- Cloud Foundry to rule them all

Finally killing DevOps

- Ops (IT) maintains the bare OS
- Devs maintain the µservices
- PaaS maps µservices
  to images and orchestrates

Finally killing DevOps

- Ops (IT) maintains the bare OS
- Devs maintain the µservices
- PaaS maps µservices
  to images and orchestrates

Questions?

By @cloud_opinion

Imagine no platforms
I wonder if you can
No need for xAAS
A brotherhood of bare metal

Imagine there is no VM
It's easy if you try
No host below us
Above us only apps