

Immutable Infrastructure

The New App Deployment



AXEL FONTAINE



@axelfontaine

axel@boxfuse.com



About Axel Fontaine



- Founder and CEO of Boxfuse
- Over 15 years industry experience
- Continuous Delivery expert
- Regular speaker at tech conferences
- JavaOne RockStar in 2014

 @axelfontaine



Flyway

flywaydb.org



boxfuse

boxfuse.com

Let's start with a small **story**



Incandescent Bulb
60 W



LED Bulb
10 W



Heater that gives off
a little bit of light



Light that gives off
a little bit of heat





Edison Screw

My
responsibility



The electricity company's
responsibility



Simple, stable,
standards-compliant
interface
with a clear contract

Room
For
Innovation



Undifferentiated
Heavy Lifting



Simple, stable,
standards-compliant
interface
with a clear contract

back to IT infrastructure ...

POLL:

what type of **infrastructure** are you running on?

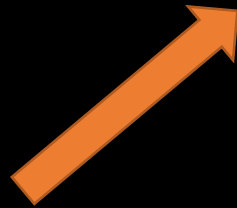
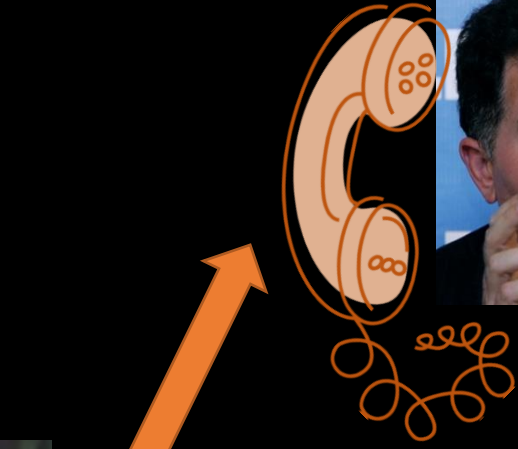
- On Premise
- Colocation
- Root Server
- Cloud

How did this **evolve** ?

sometime in the 20th century ...







ON PREM

=



+



+



Challenges

- Power, Network, Cooling
- Physical Security
- Physical Space
- Procurement, Vendor Management
- Capacity Planning
- Financing
- OS + Patches
- App + Updates

ON
PREM

=



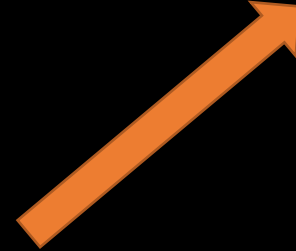
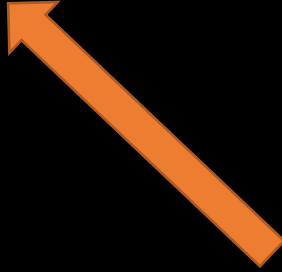
+



+



Our
responsibility



COLO =



+



+



Our
responsibility

Their
responsibility

COLO =



Simple, stable,
standards-compliant
interface:

(19" Rack, AC Power,
Ethernet, ...)

COLO =



+



+

Undifferentiated
Heavy Lifting

Our
responsibility

Can change as
long as it
complies with
the interface
contract

ROOT SERVER

=



+

Undifferentiated
Heavy Lifting

Our
responsibility

Can change as
long as it
complies with
the interface
contract

ROOT SERVER

=



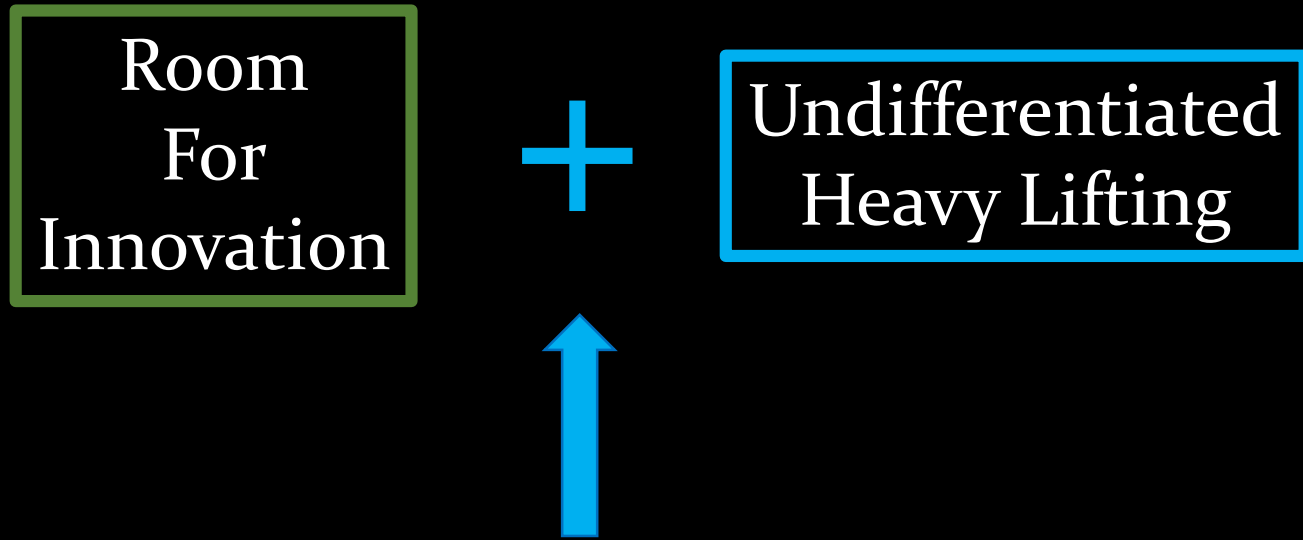
+

Undifferentiated
Heavy Lifting



Simple, stable, standards-
compliant interface

Software <-> Hardware

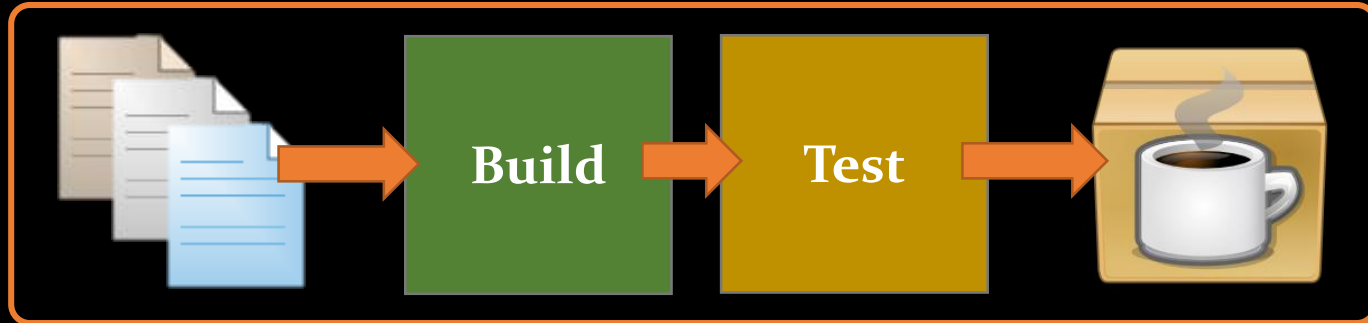


Could this be our industry's Edison Screw?

Let's talk about software

POLL:
which level of **automation** are you at?

- Build
- Unit Tests
- Continuous Integration
- Acceptance Tests
- Continuous Deployment (Code)
- Continuous Deployment (Code + DB + Configuration)
- Infrastructure





- One **immutable** unit
- **Regenerated** after every change
- **Promoted** from Environment to Environment

Classic Mistake: Build per Environment

App

App Server

Language

Libraries

OS Kernel



App

App Server

Language

Libraries

OS Kernel



why aren't we doing the same
for the **layers** this is running on ???

what could possibly go wrong
in these other **layers** ???



missing software



S.H.I.T.

Servicio de Hosteleria

Industrial de Terrassa

wrong name



bad version



incorrect permissions

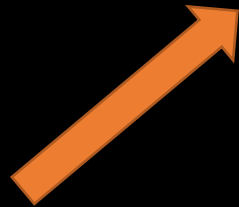
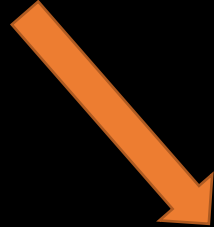
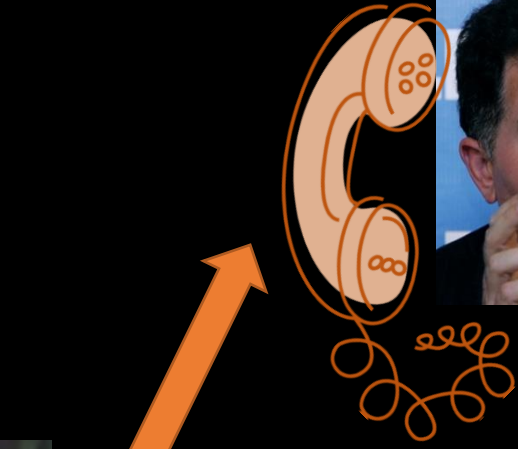


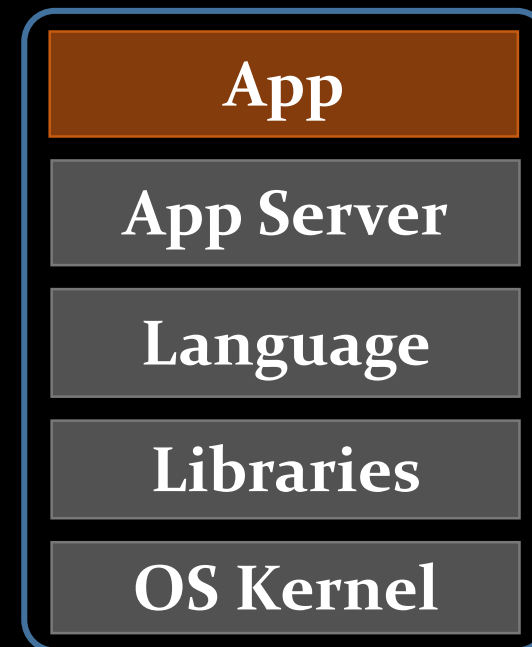
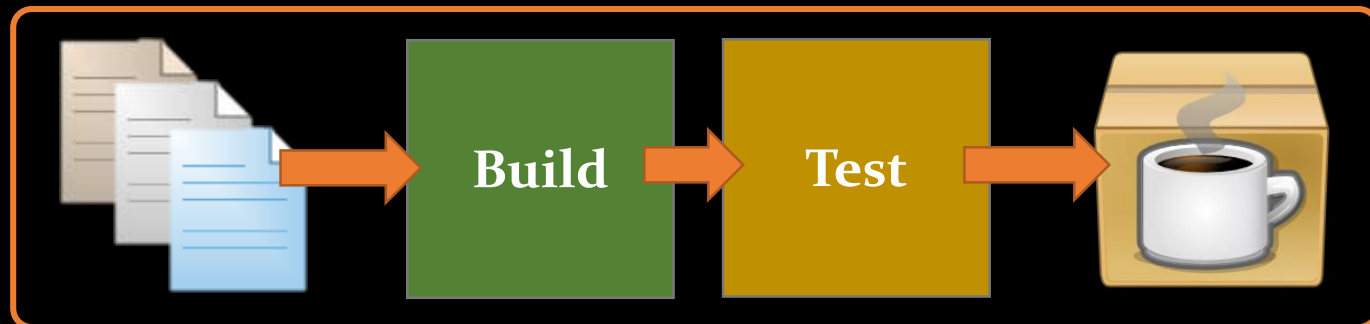
TOILET

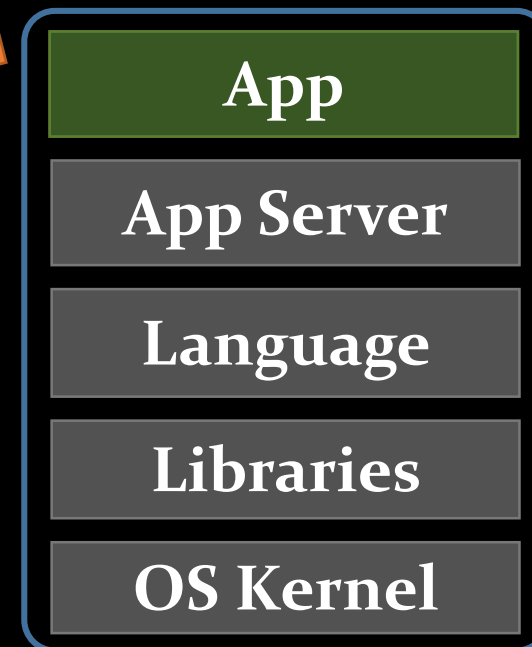
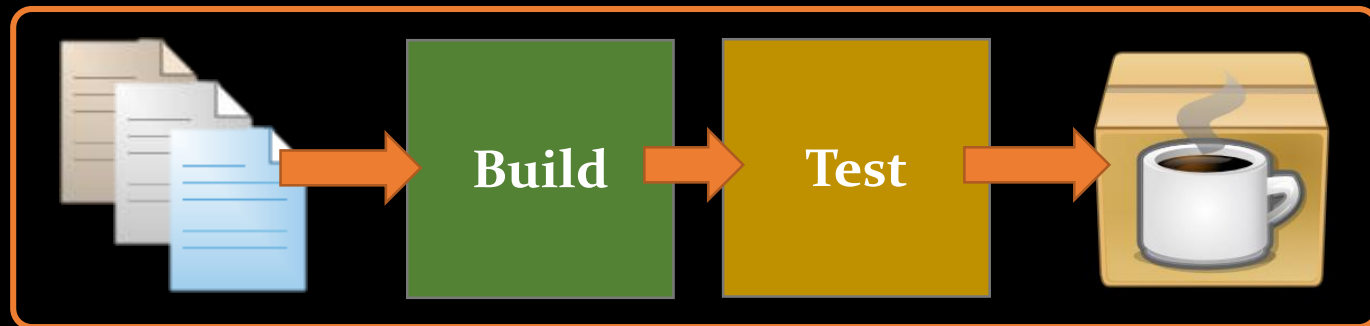
OCCUPIED

critical resource in use

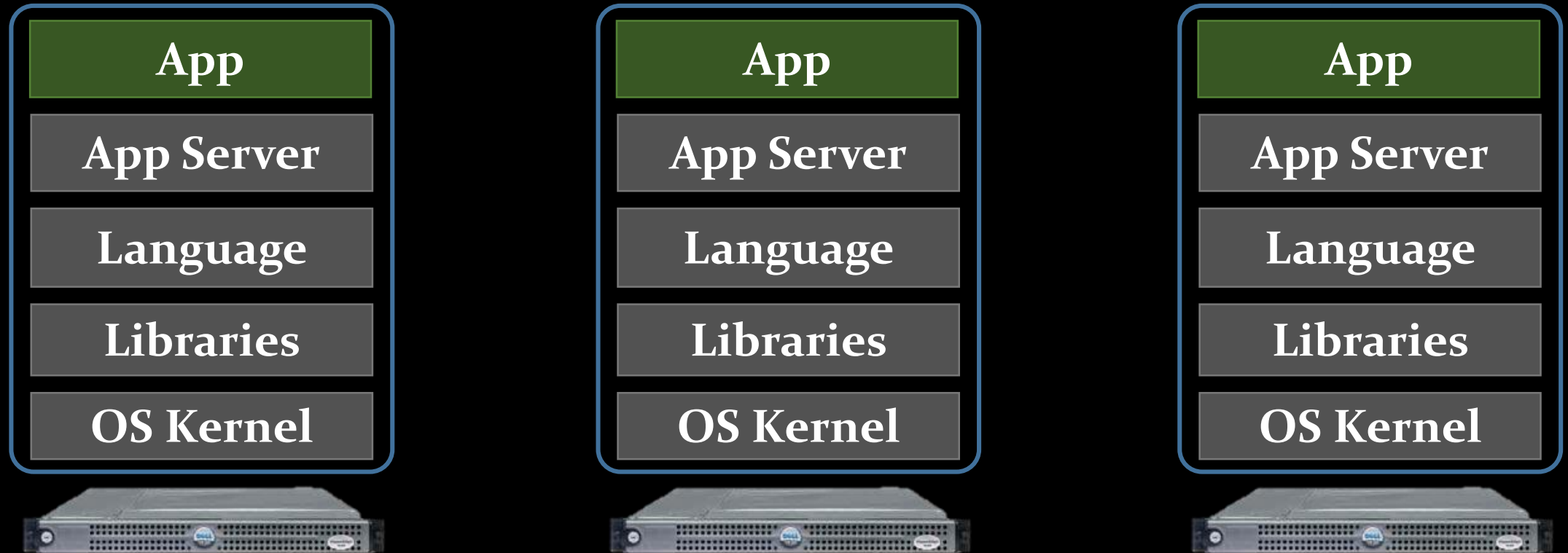
what aren't we holding our servers to the
same standards as our applications ???







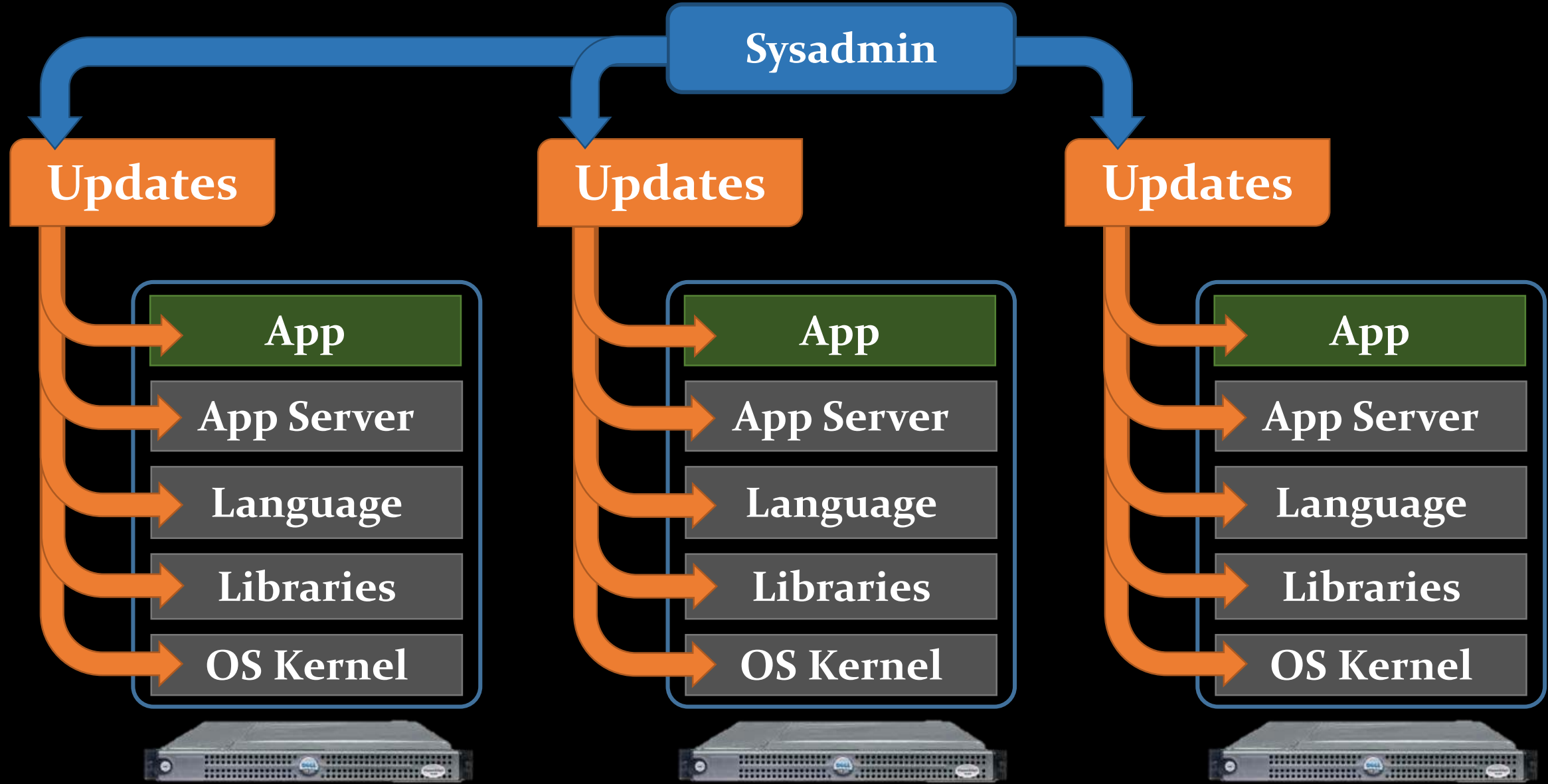
Multiple instances in multiple Environments

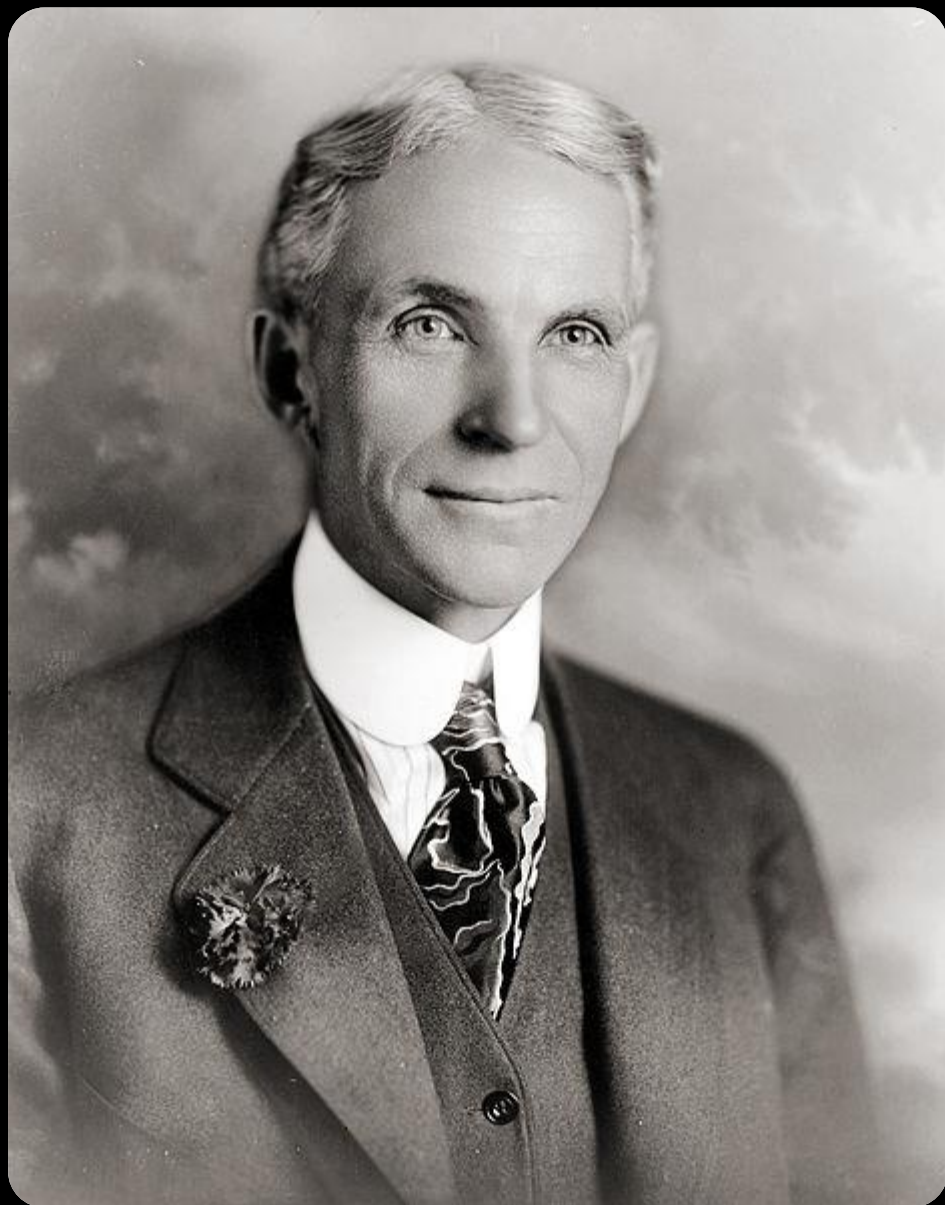


Multiple instances in multiple **Environments**



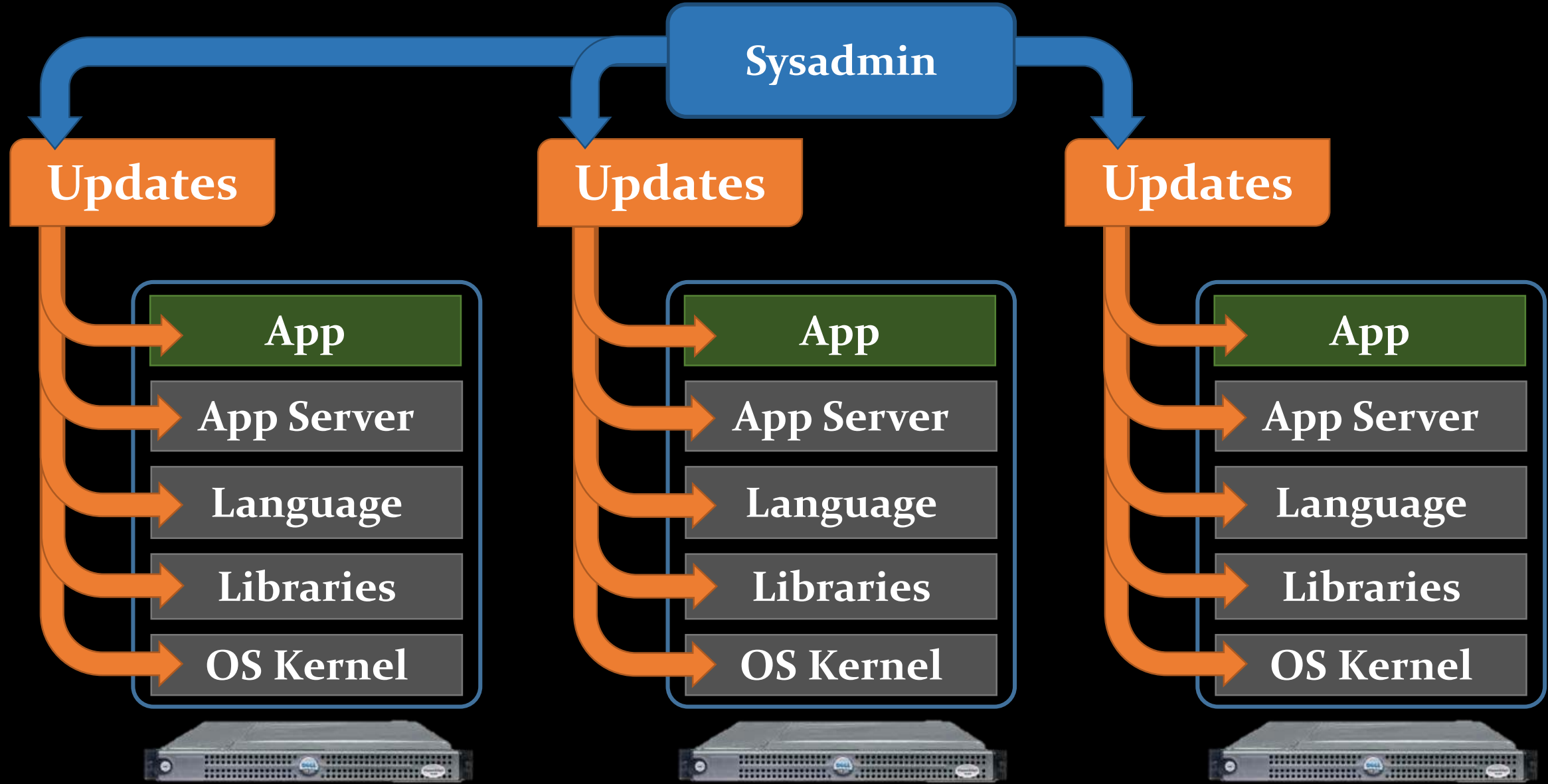
- All instances should be as similar as possible (any difference is a potential source of errors)
- That also includes your local Dev environment!
- Must be able to reliably provision new ones (and recreate existing ones from scratch)

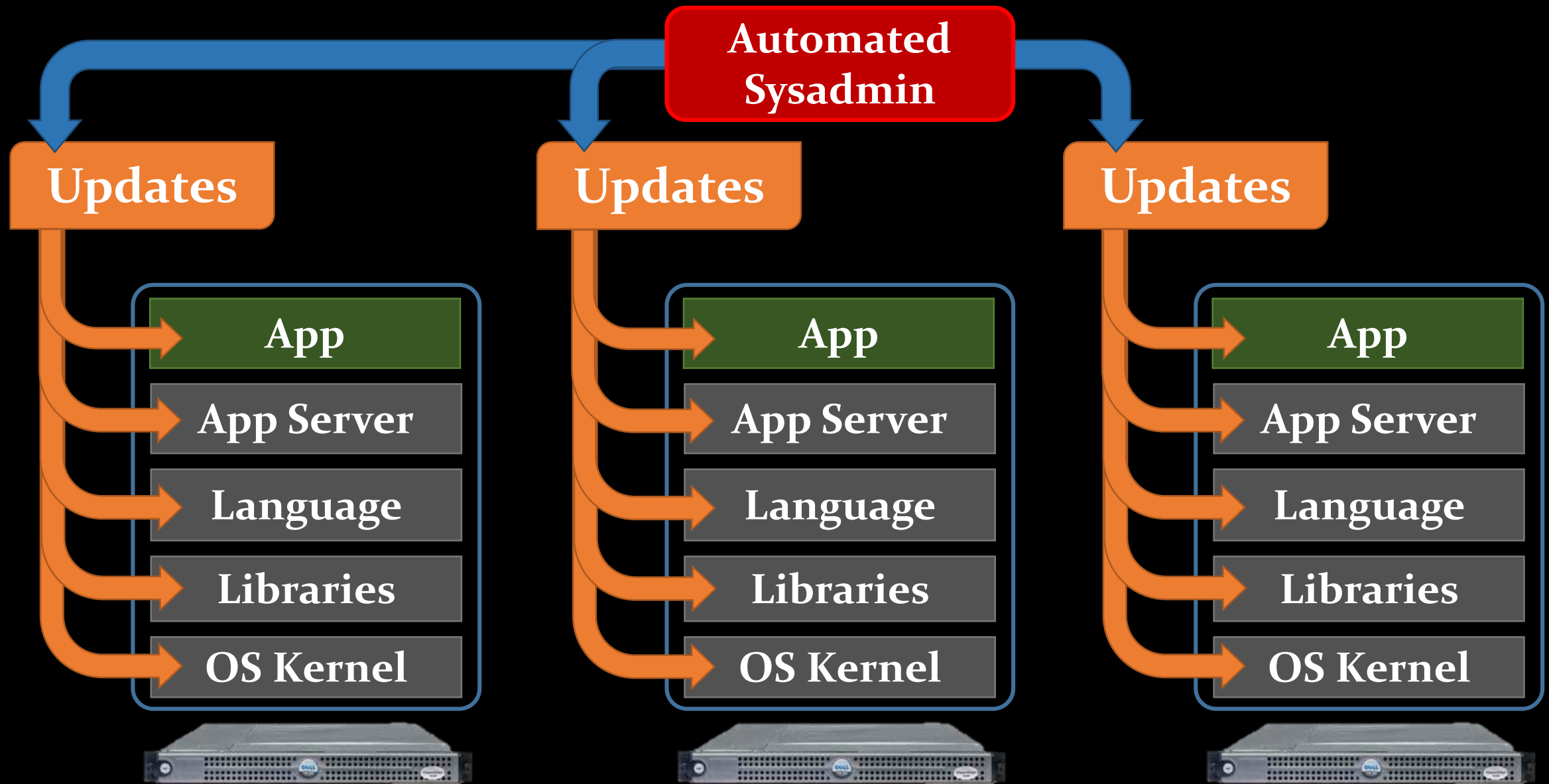




*If I had asked my
customers what they
wanted they would have
said a **faster horse**.*

Henry Ford





fast forward to 2015 ...



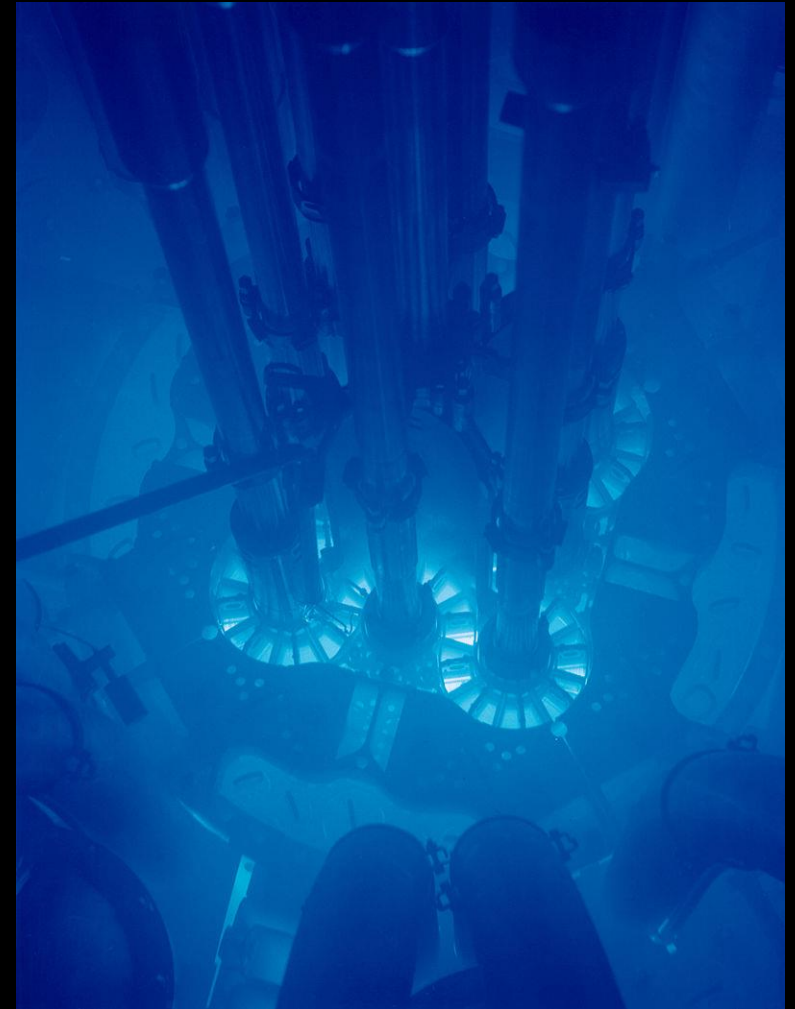
*Every day, AWS adds
enough server capacity
to power the whole \$5B
enterprise Amazon.com
was in 2003.
Weekends included.*





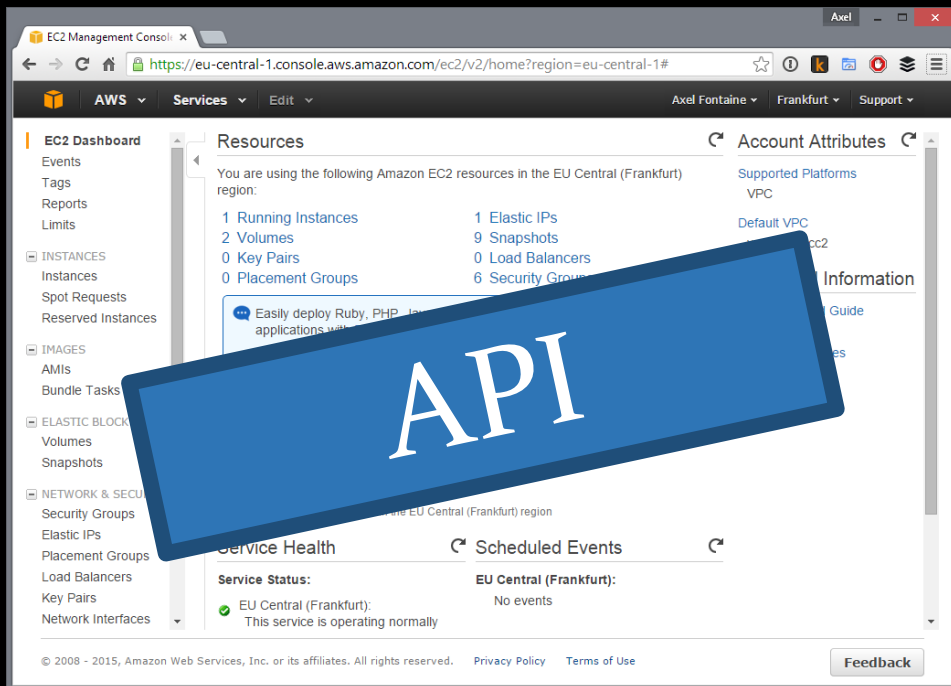
Control Plane

"RIAN archive 341194 Kursk Nuclear Power Plant" by RIA Novosti archive, image #341194, merged by yakovlev, CC-BY-SA 3.0, Licensed under CC BY-SA 3.0 via Wikimedia Commons - http://commons.wikimedia.org/wiki/File:RIAN_archive_341194_Kursk_Nuclear_Power_Plant.jpg#mediaviewer/File:RIAN_archive_341194_Kursk_Nuclear_Power_Plant.jpg

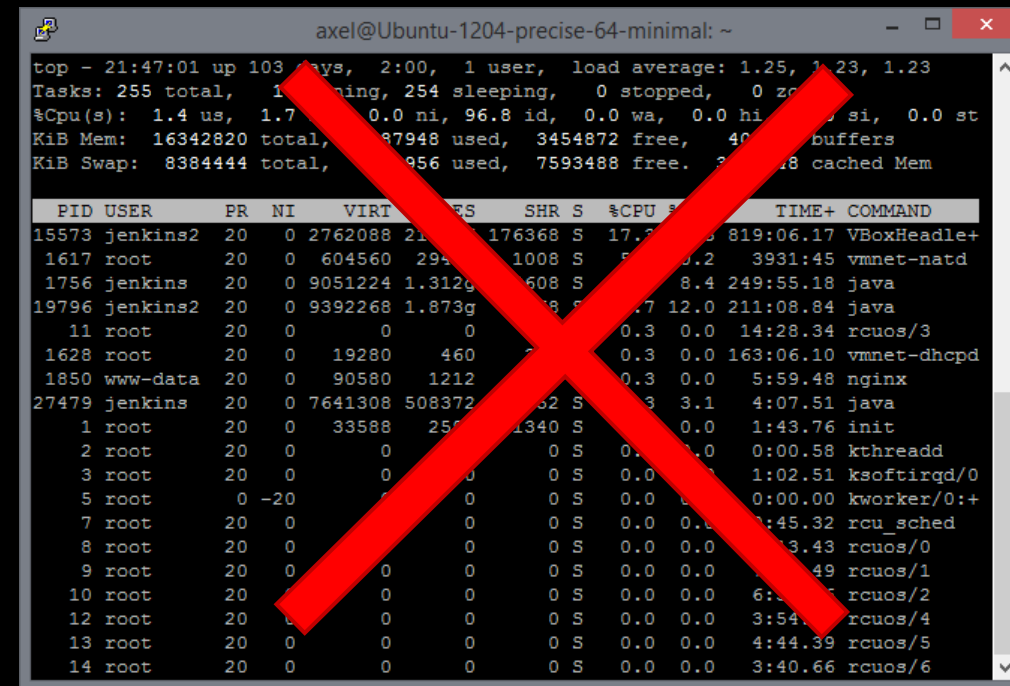


Data Plane

"Advanced Test Reactor" by Argonne National Laboratory - originally posted to Flickr as Advanced Test Reactor Core, Idaho National Laboratory, uploaded using FlickrCommons. Licensed under CC BY-SA 2.0 via Wikimedia Commons - http://commons.wikimedia.org/wiki/File:Advanced_Test_Reactor.jpg#mediaviewer/File:Advanced_Test_Reactor.jpg



Control Plane

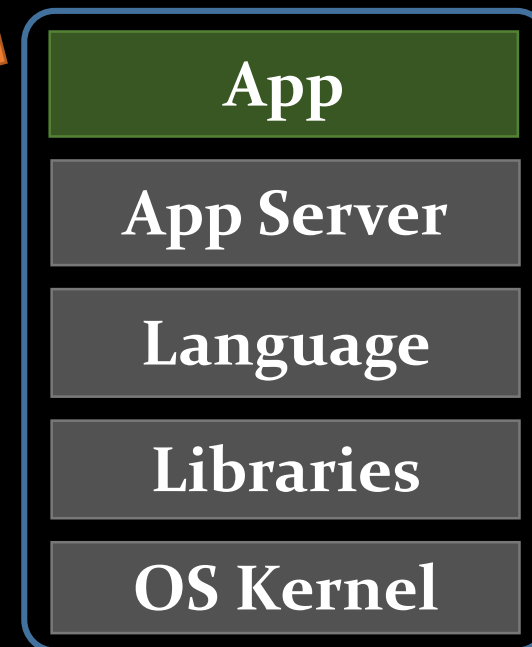
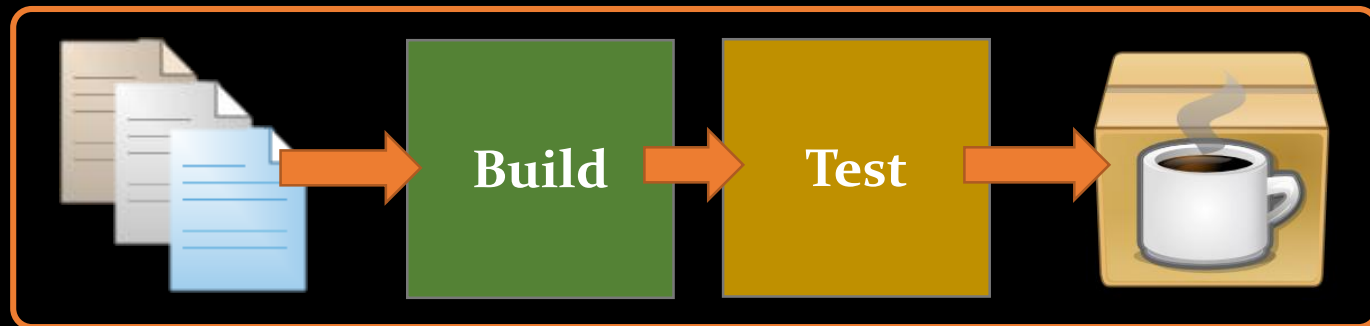


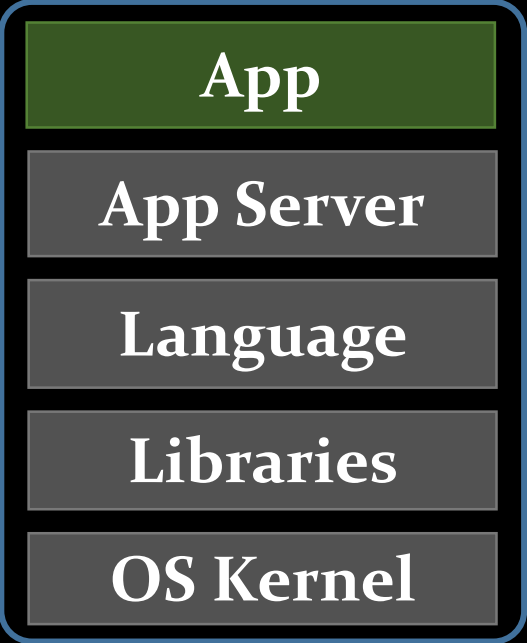
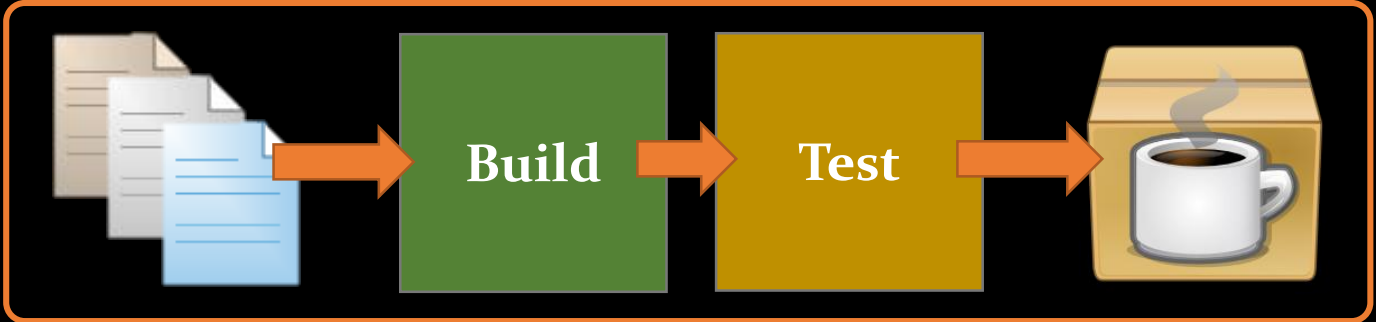
Data Plane

Benefits of the cloud

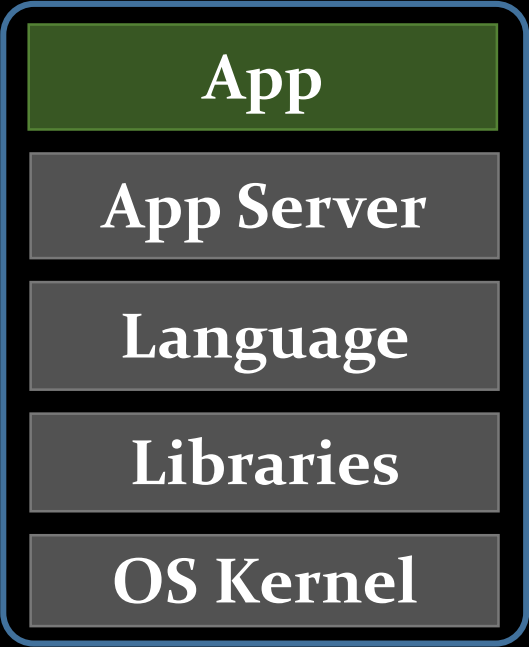
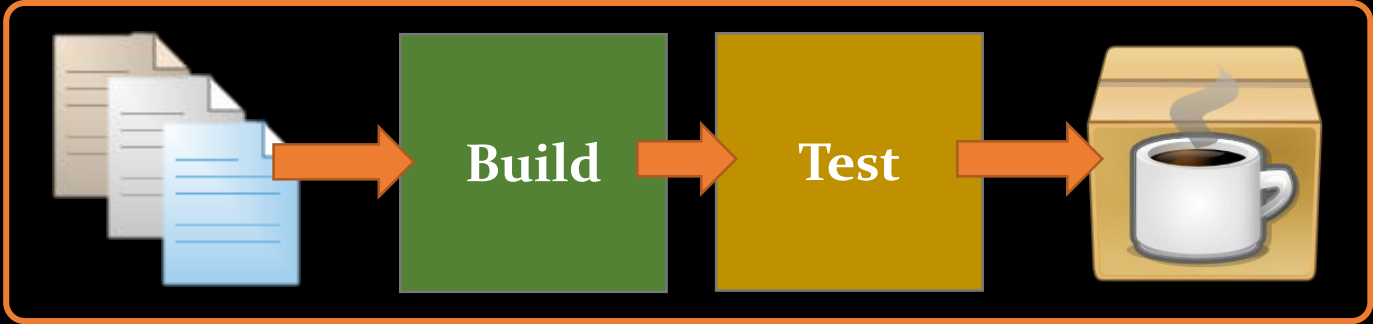
- Shift to a world of abundance
(no more resource scarcity)
- Clean Control Plane/Data Plane split
with API-based provisioning
- Cost-based Architectures
with the ability to turn infrastructure off

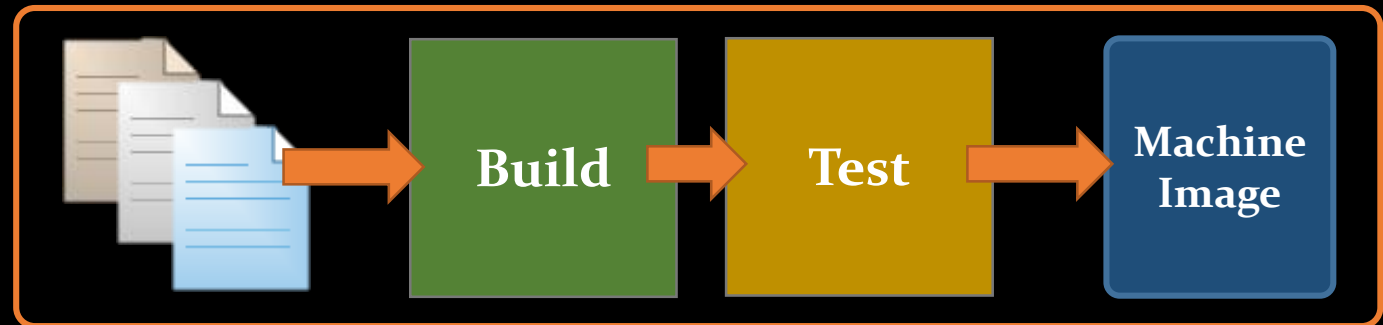
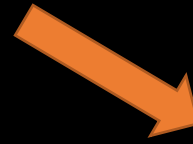
it is time to rethink the faster horse



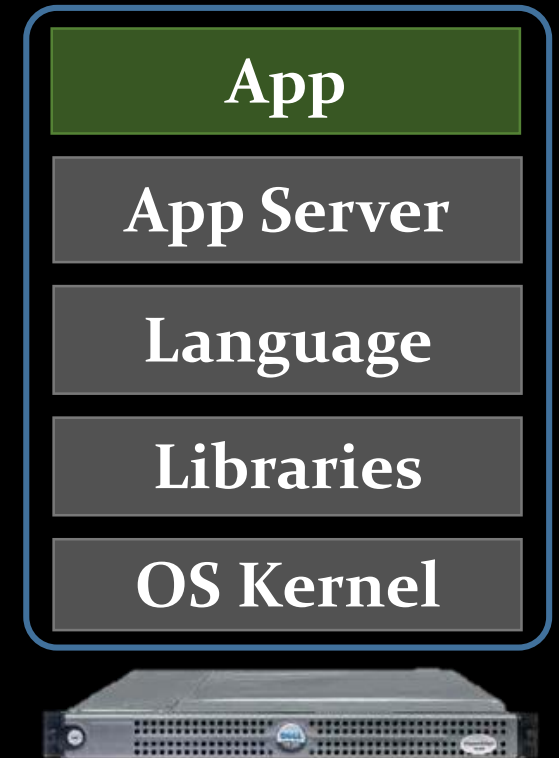
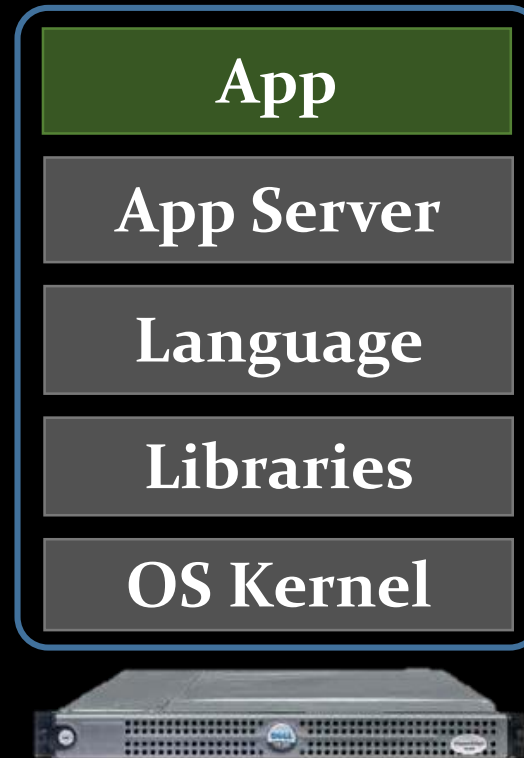
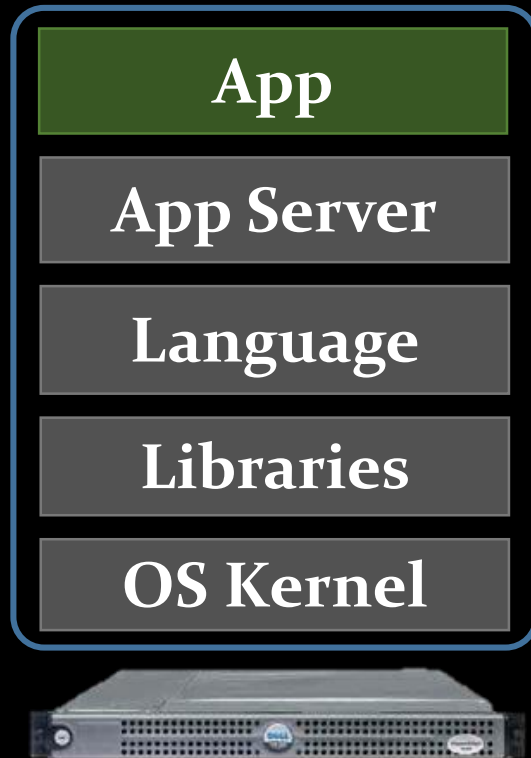


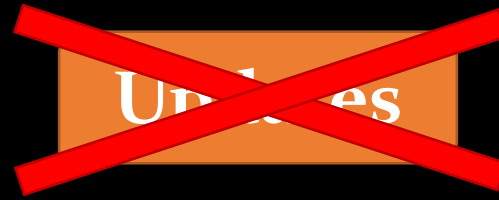
Undifferentiated
Heavy lifting





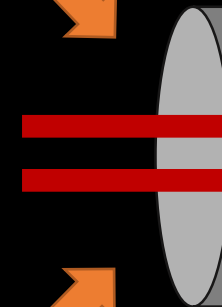
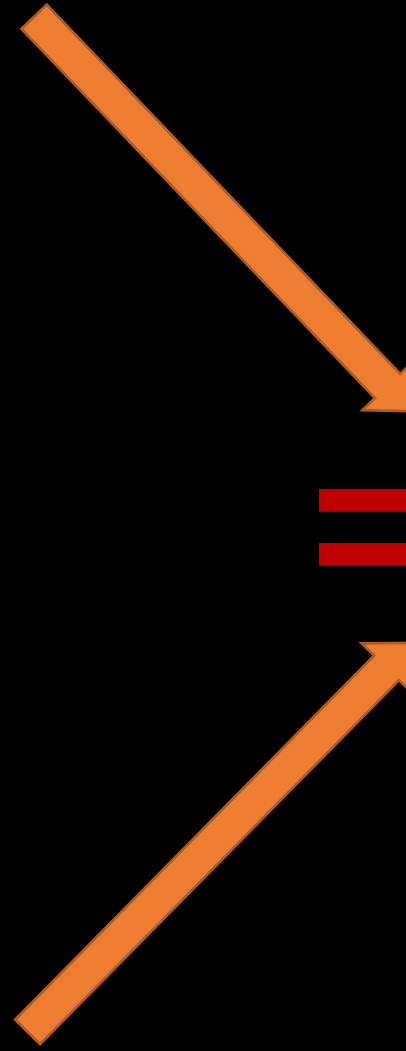
Updates





but there is one **big** problem left ...

Machine Image



Network Cable

Multiple
GB

Network Cable

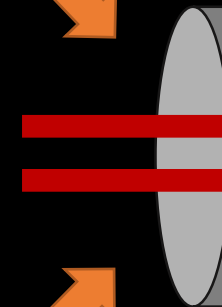
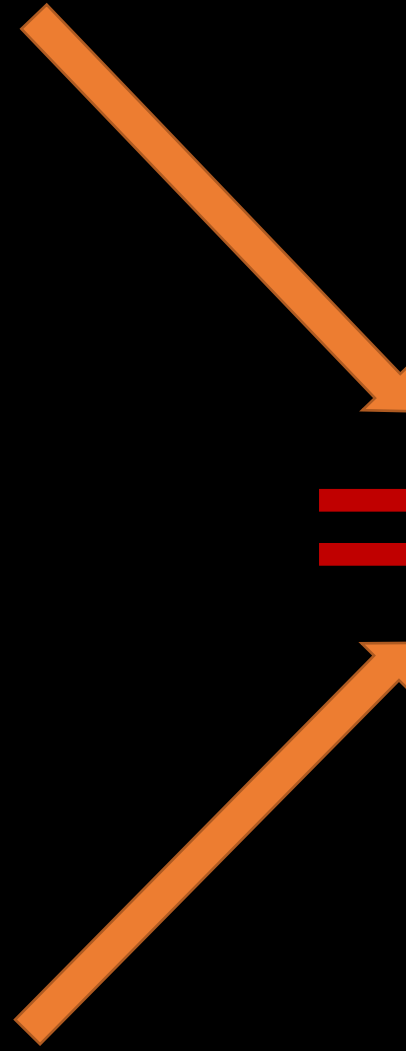


*Running servers in production should be like going **backpacking**. You take the bare minimum with you. Anything else is going to hurt.*

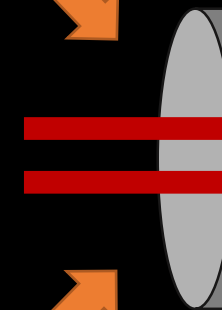
A Wise Man

what is really adding business value ???

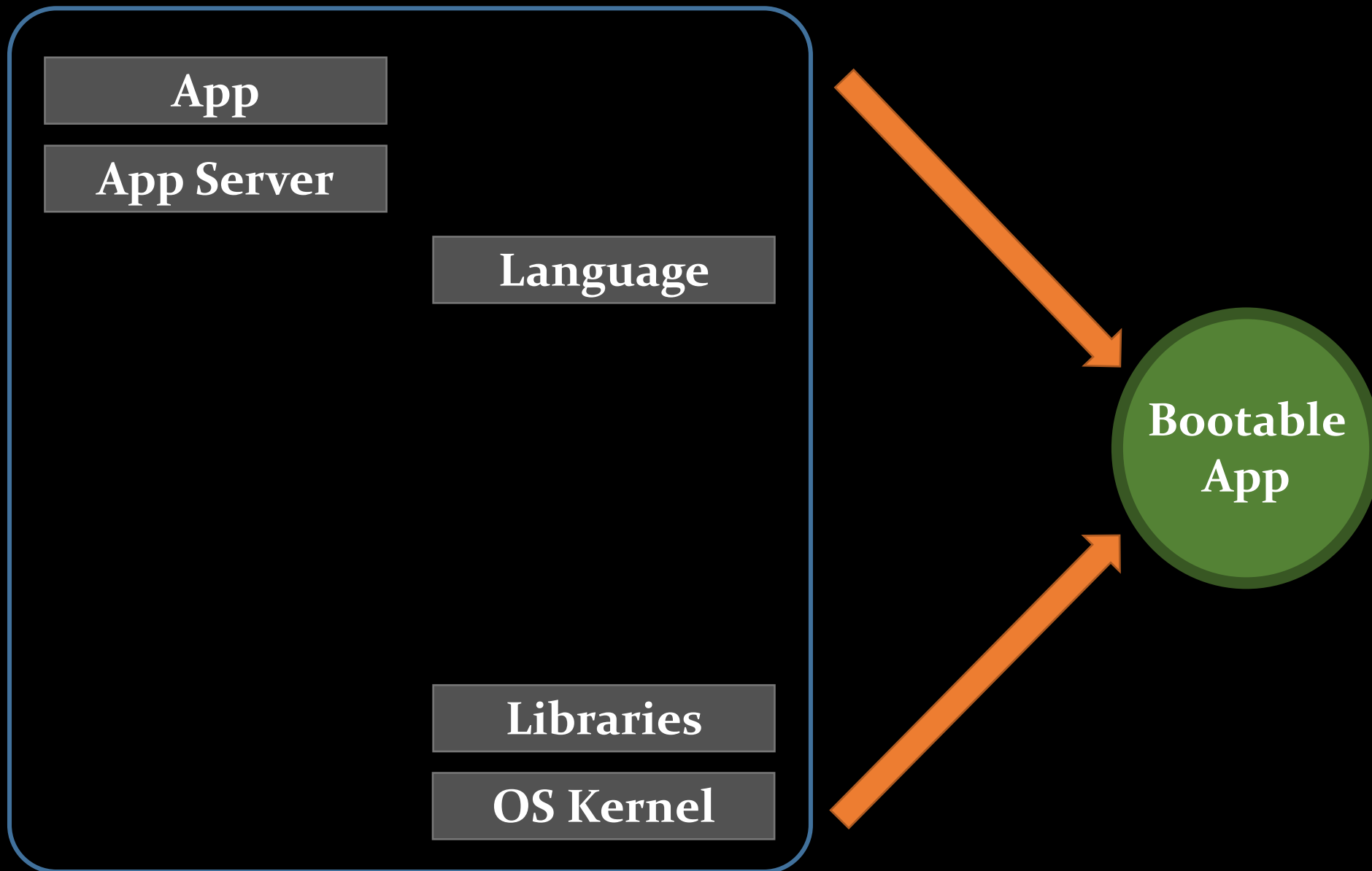
Machine Image



Network Cable



Network Cable

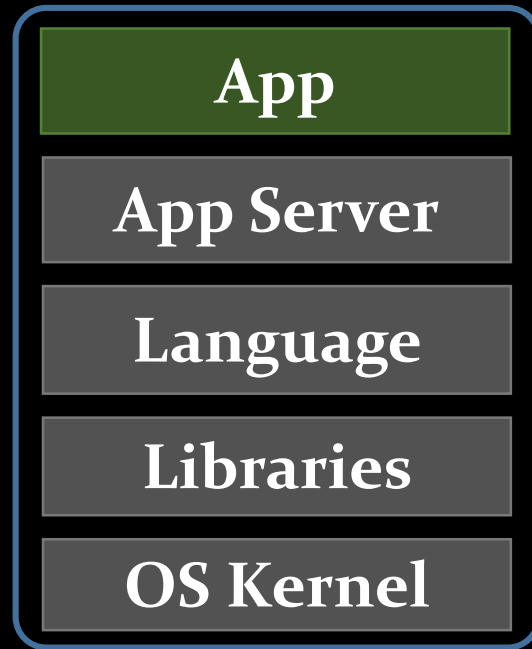


Multiple
GB

40 – 80
MB



who is this for ???



12-factor app

demo

What are the implications ???

Focus **shift**

Instance



Service

Individual instances become **disposable**

Treat servers like **cattle** instead of pets



high **uptime** is a liability



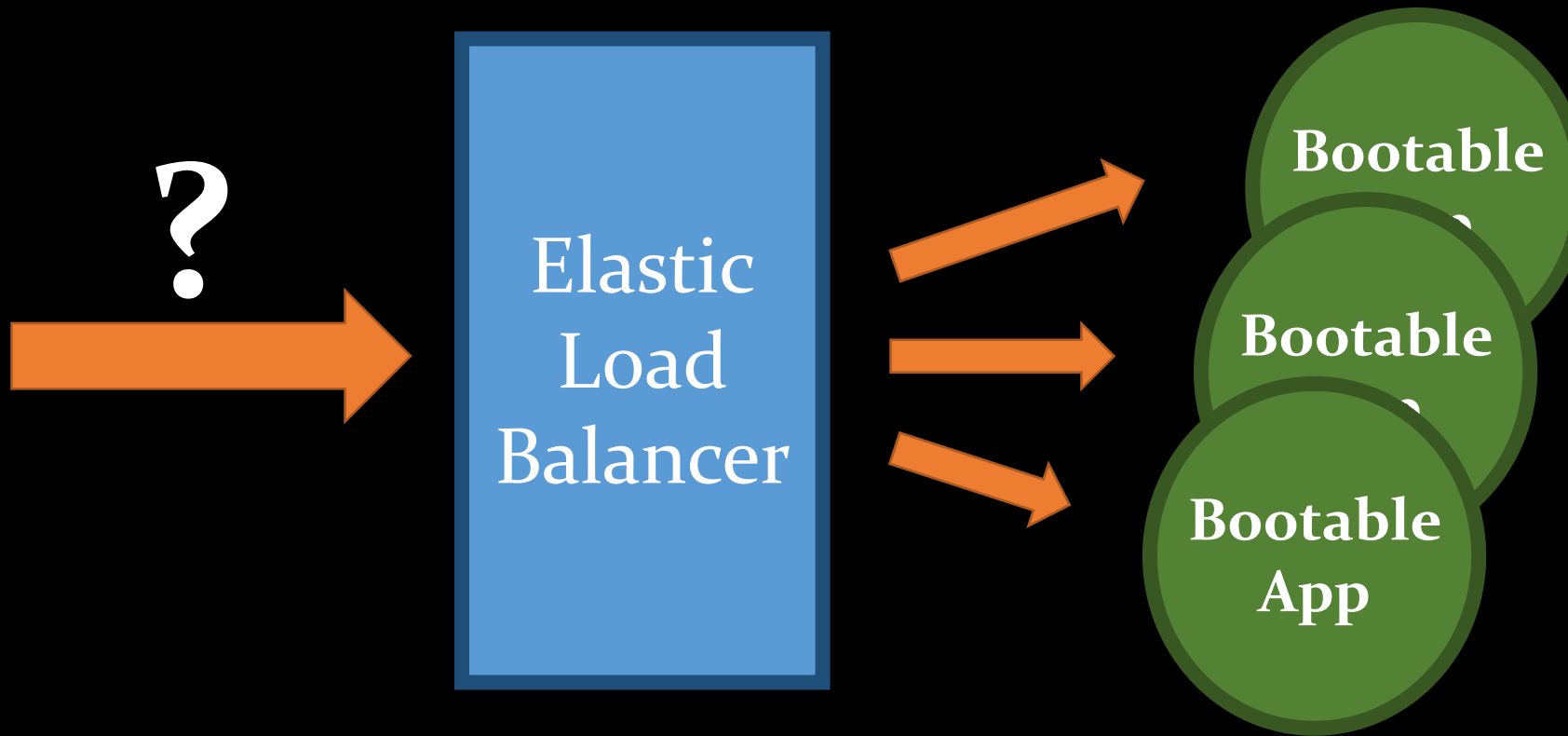
axel@Ubuntu-1204-precise-64-minimal: ~



```
axel@Ubuntu-1204-precise-64-minimal:~$ uptime -p  
up 14 weeks, 5 days, 2 hours, 47 minutes  
axel@Ubuntu-1204-precise-64-minimal:~$
```

**The longer an instance is up,
the harder it becomes to recreate exactly
(and it will fail eventually!)**

How to solve **service discovery** ?



Use a stable entry point with an internal registry

What about **security** ?



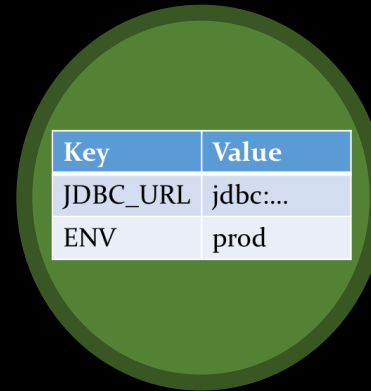
When was the last time your toaster got hacked?

What about security ?

- Smallest possible attack surface
- Vastly reduced implications due to low uptime and transient nature of instances
- Very difficult to exploit other systems because essential tooling is missing

what about configuration ???

- Bake as much configuration as possible for all environments directly in the Bootable App
- Use environment detection and auto-configuration
- Pass remaining configuration at startup and expose it as environment variables



Key	Value
JDBC_URL	jdbc:...
ENV	prod



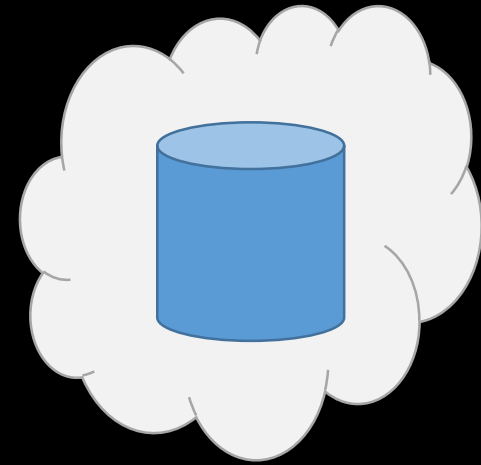
what about the database ???

what about the database ???



what about the database ???

- Keep all persistent state, including the database, out of the instance
- Many good hosted solutions available like Amazon RDS or Google Cloud SQL
- Use a database migration tool like Flyway to update on application startup



what about the **logs** ???



Ship logs to a **central log server**

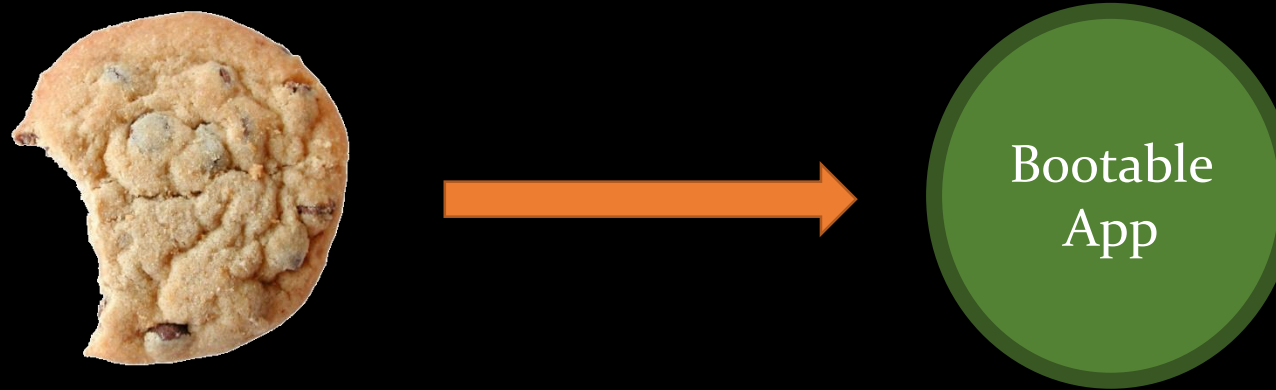
where they can be

- aggregated
- stored and backuped
- indexed
- searched through a nice web UI

Many good hosted solutions

- Loggly
- Logentries
- Papertrail
- ...

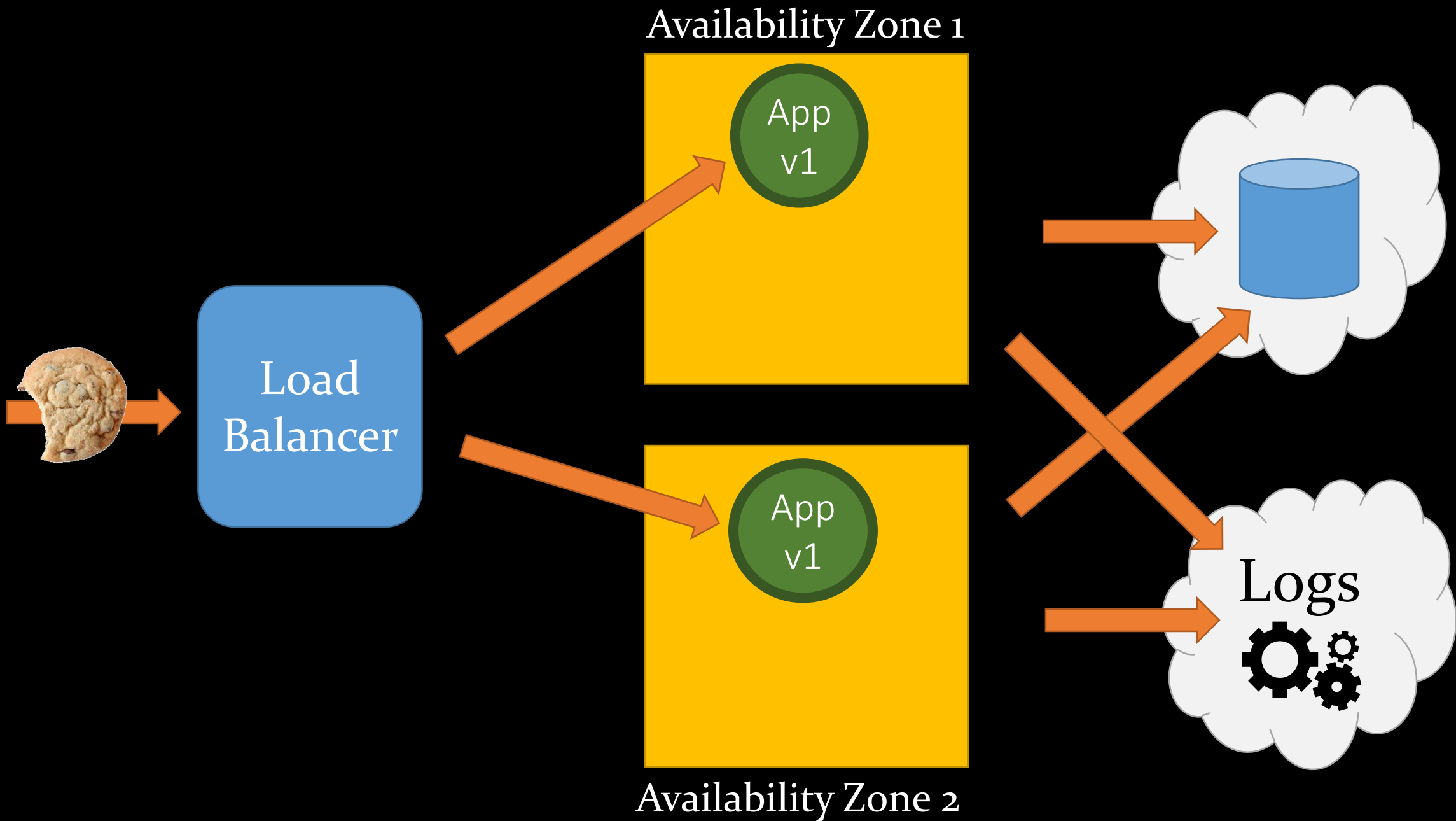
what about sessions ???

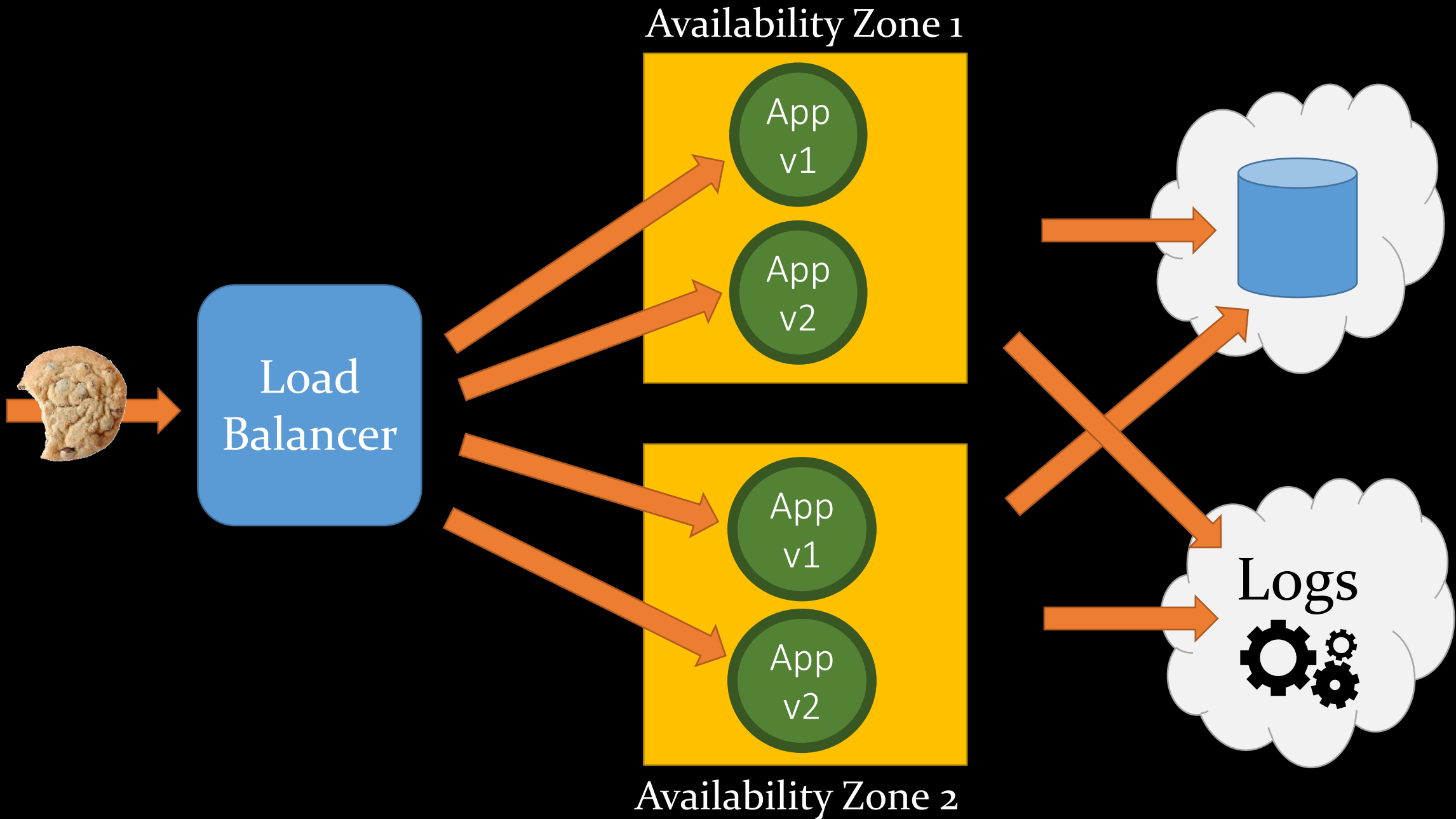


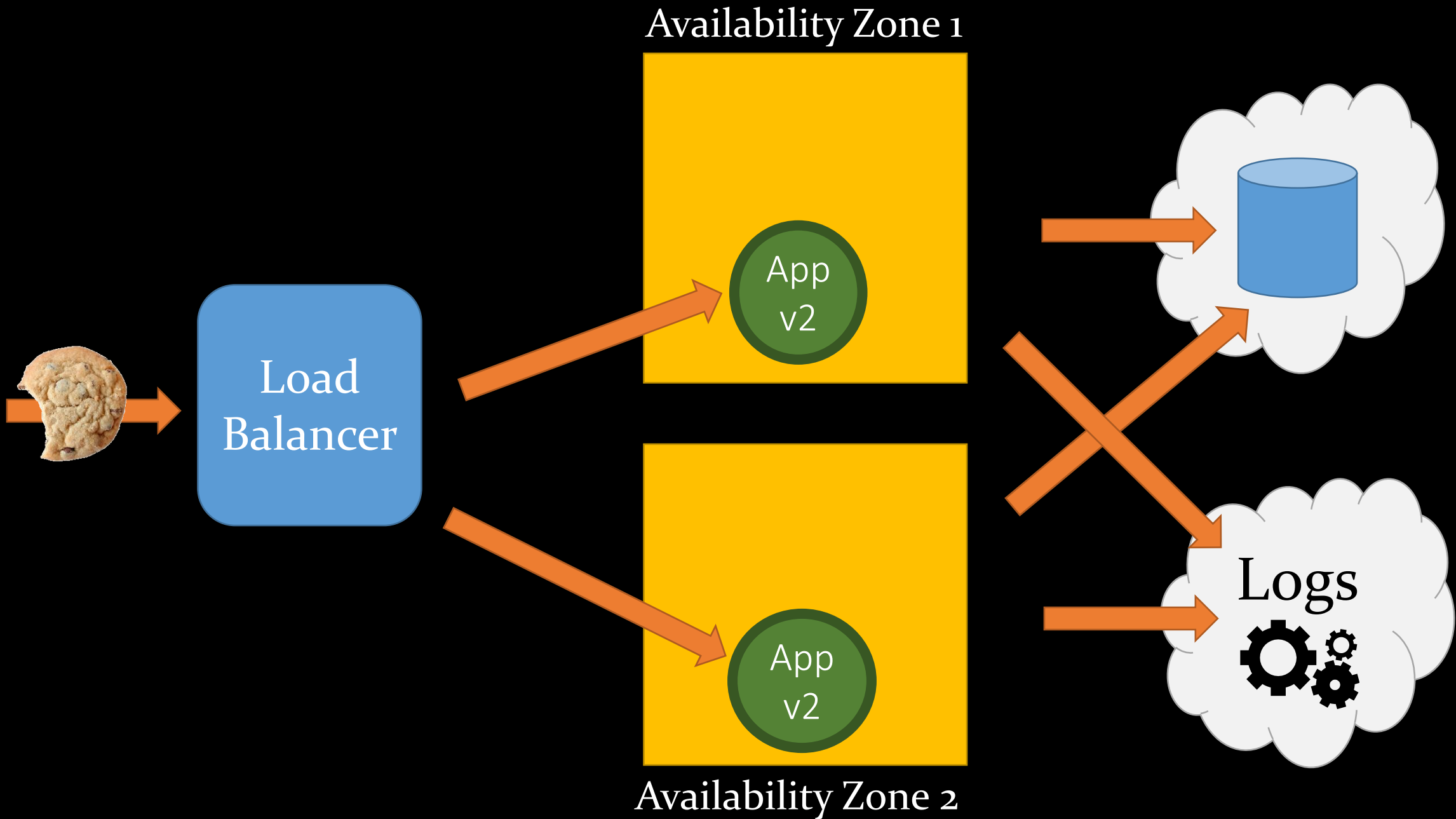
Keep session in an encrypted and signed **cookie**

- avoids session timeouts
- avoids server clustering & session replication
- avoids sticky sessions & server affinity

what about rolling out new versions ???







what about containers ???

understanding modern CPUs



Both Intel and AMD have hardware support for virtualization

- isolation
- performance

Bootable App

Hypervisor

Hardware

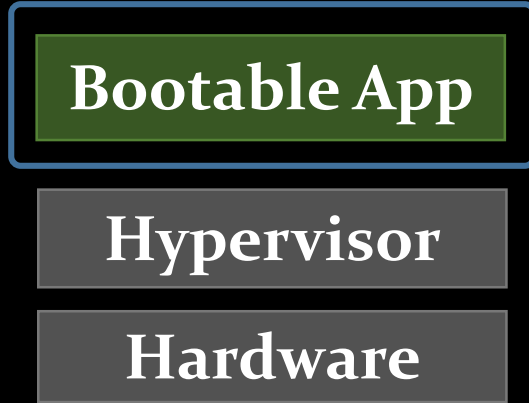
On Prem

Bootable App

**OS+Container
Runtime**

Hardware

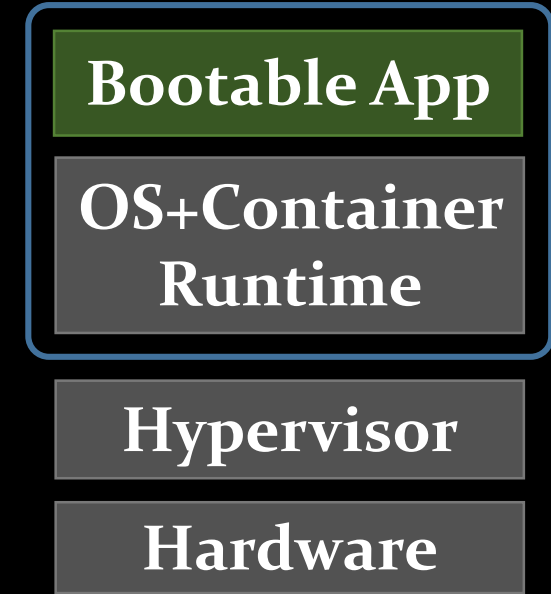
On Prem



On Prem /
Cloud



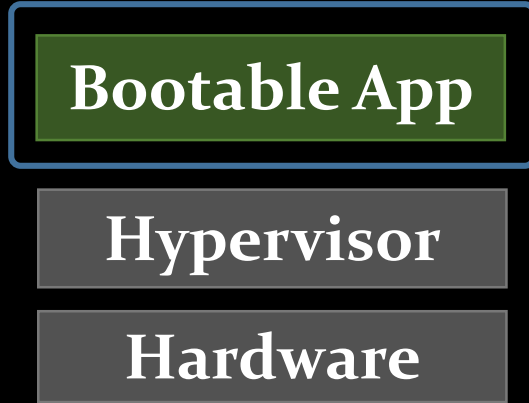
On Prem



Cloud



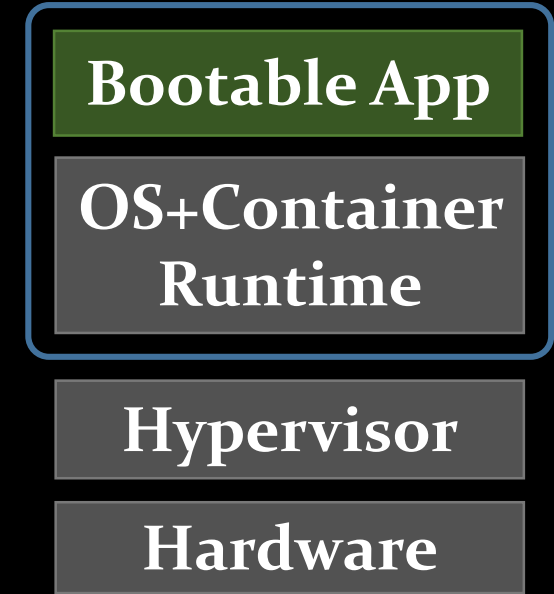
Only makes sense if
you cannot afford
\$9.60/month
granularity



On Prem /
Cloud



On Prem



Cloud



Only makes sense if
you cannot afford
1.3 cents /hour
granularity

summary



- One **immutable** unit
- **Regenerated** after every change
- **Promoted** from Environment to Environment

Classic Mistake: Build per Environment

A green circle with a dark green border, containing the text "Bootable App" in white.

Bootable App

- One **immutable** unit
- **Regenerated** after every change
- **Promoted** from Environment to Environment

Classic Mistake: Build per Environment



boxfuse

boxfuse.com



AXEL FONTAINE



@axelfontaine

Thanks !



boxfuse.com