

# IMMUTABLE INFRASTRUCTURE

---

RISE OF THE MACHINE IMAGES

# About Axel Fontaine



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

 @axelfontaine





*Flyway*

flywaydb.org



boxfuse

[boxfuse.com](https://boxfuse.com)

about

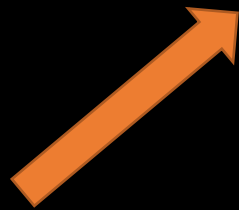
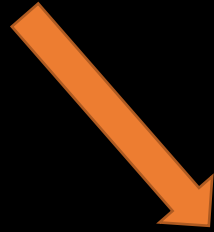
questions

sometime in the 20<sup>th</sup> century ...









# Challenges

ON  
PREM

=



+



+



- ✓ OS Install
- ✓ OS Patching
- ✓ App Install
- ✓ App Updates

- ✓ Procurement
- ✓ Vendor Mgmt
- ✓ Capacity Plan.
- ✓ Financing

- ✓ Power
- ✓ Network
- ✓ Cooling
- ✓ Phys. Security
- ✓ Phys. Space

# Challenges

ON  
PREM

=



+



Lots of undifferentiated heavy lifting

✓ OS In

✓

✓ for Mgmt

Capacity Plan.

✓ Financing

✓ Power

✓ Network

✓ Cooling

✓ Phys. Security

✓ Phys. Space



**FNB** GOLD Credit Card



4000 1234 5678 9010

4127

EXPIRES  
END ▶ 05/84

SARAH CONNOR

**VISA**

# Challenges

**ON  
PREM**

=



+



+



- ✓ OS Install
- ✓ OS Patching
- ✓ App Install
- ✓ App Updates

🕒 Hours

- ✓ Procurement
- ✓ Vendor Mgmt
- ✓ Capacity Plan.
- ✓ Financing

🕒 Days or Weeks

- ✓ Power
- ✓ Network
- ✓ Cooling
- ✓ Phys. Security
- ✓ Phys. Space

🕒 Months

# Challenges



# Challenges

**ROOT  
SERVER =**



- ✓ OS Install
- ✓ OS Patching
- ✓ App Install
- ✓ App Updates

- ✓ Procurement
- ✓ Vendor Mgmt
- ✓ Capacity Plan.
- ✓ Financing

🕒 Hours

🕒 Days or Weeks

# Let's talk about software



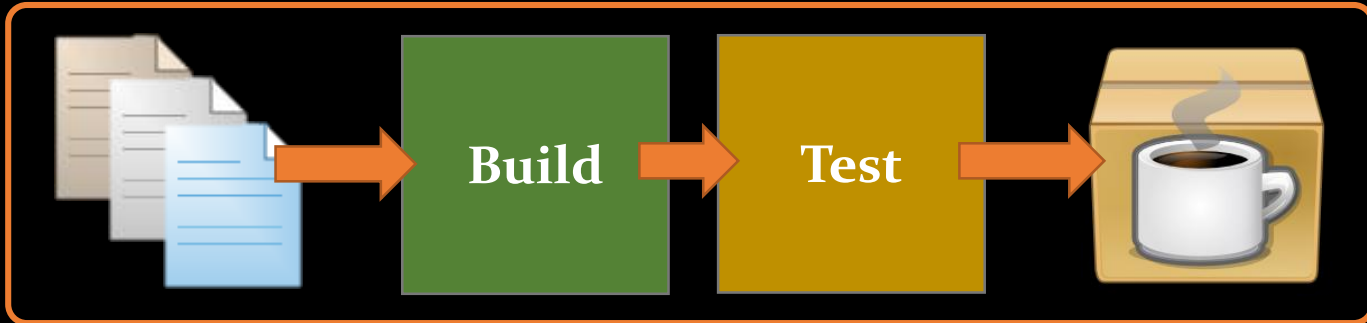
- ✓ OS Install
- ✓ OS Patching
- ✓ App Install
- ✓ App Updates

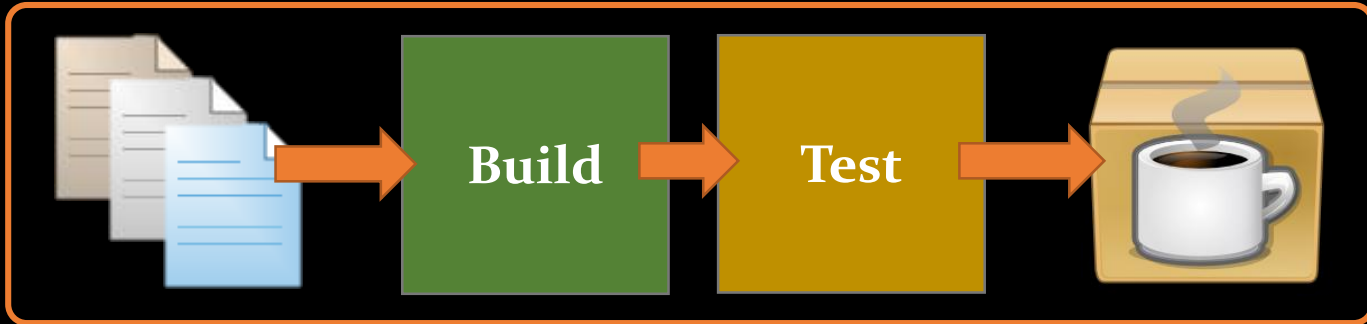


# POLL:

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**

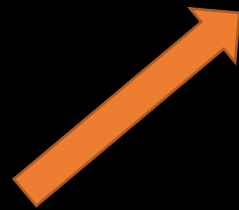
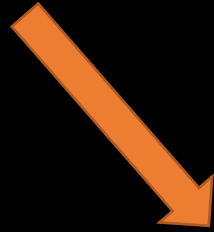
**Language**

**Libraries**

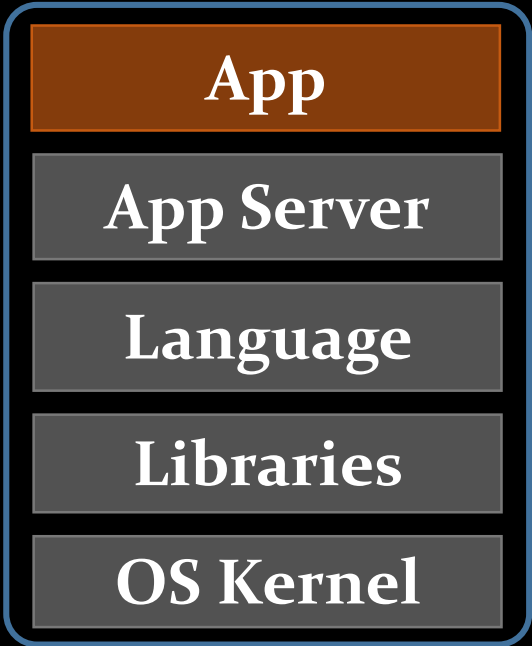
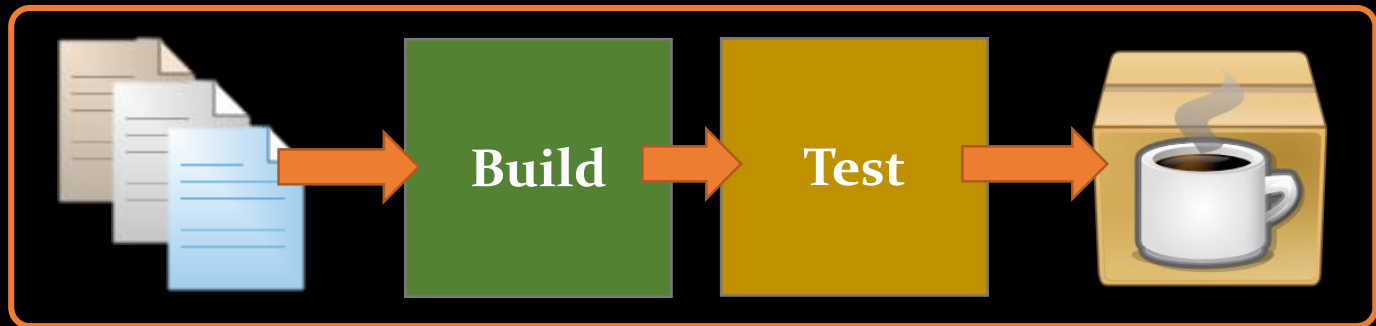
**OS Kernel**

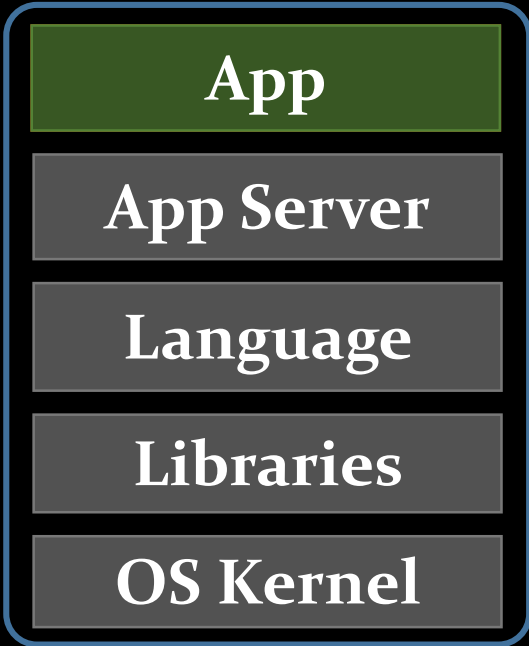
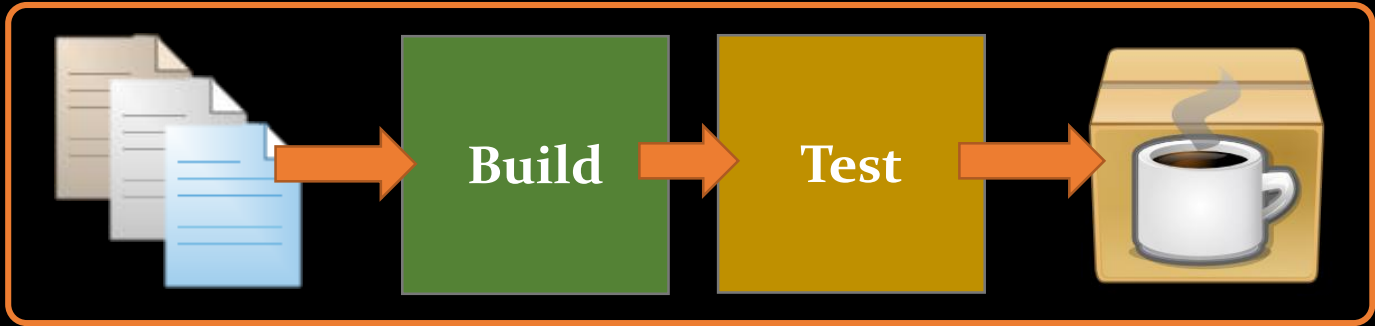


why aren't we doing the same  
for the **layers** this is running on ???

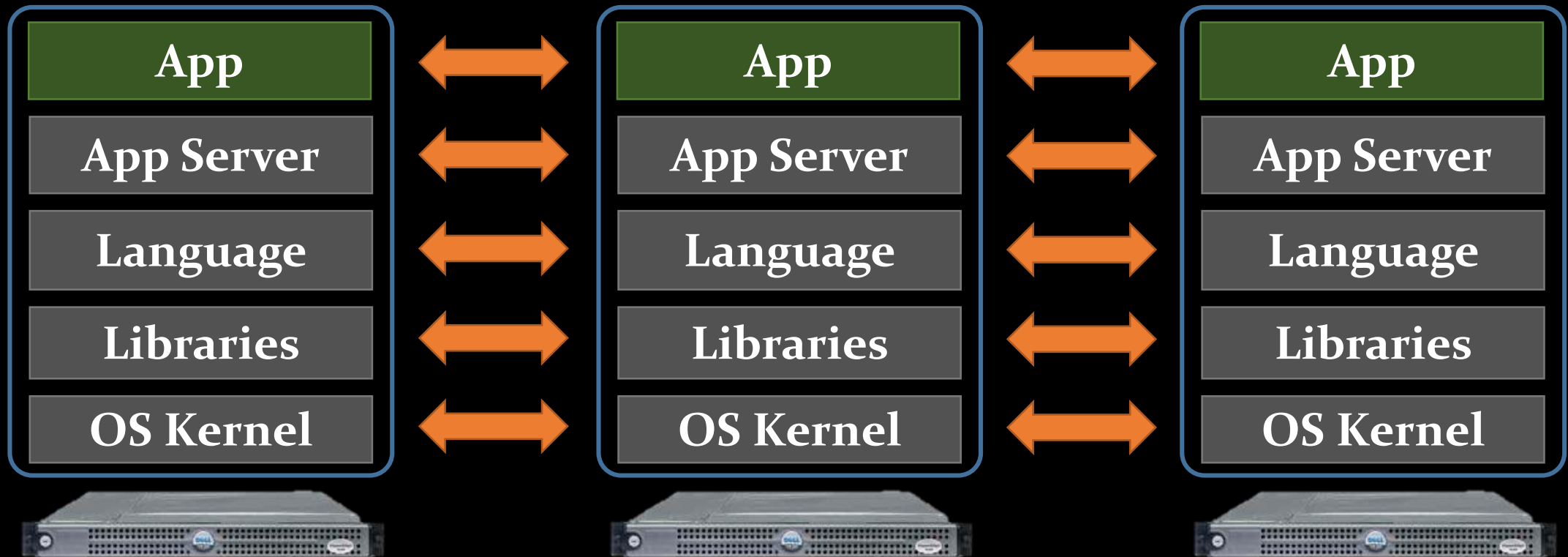


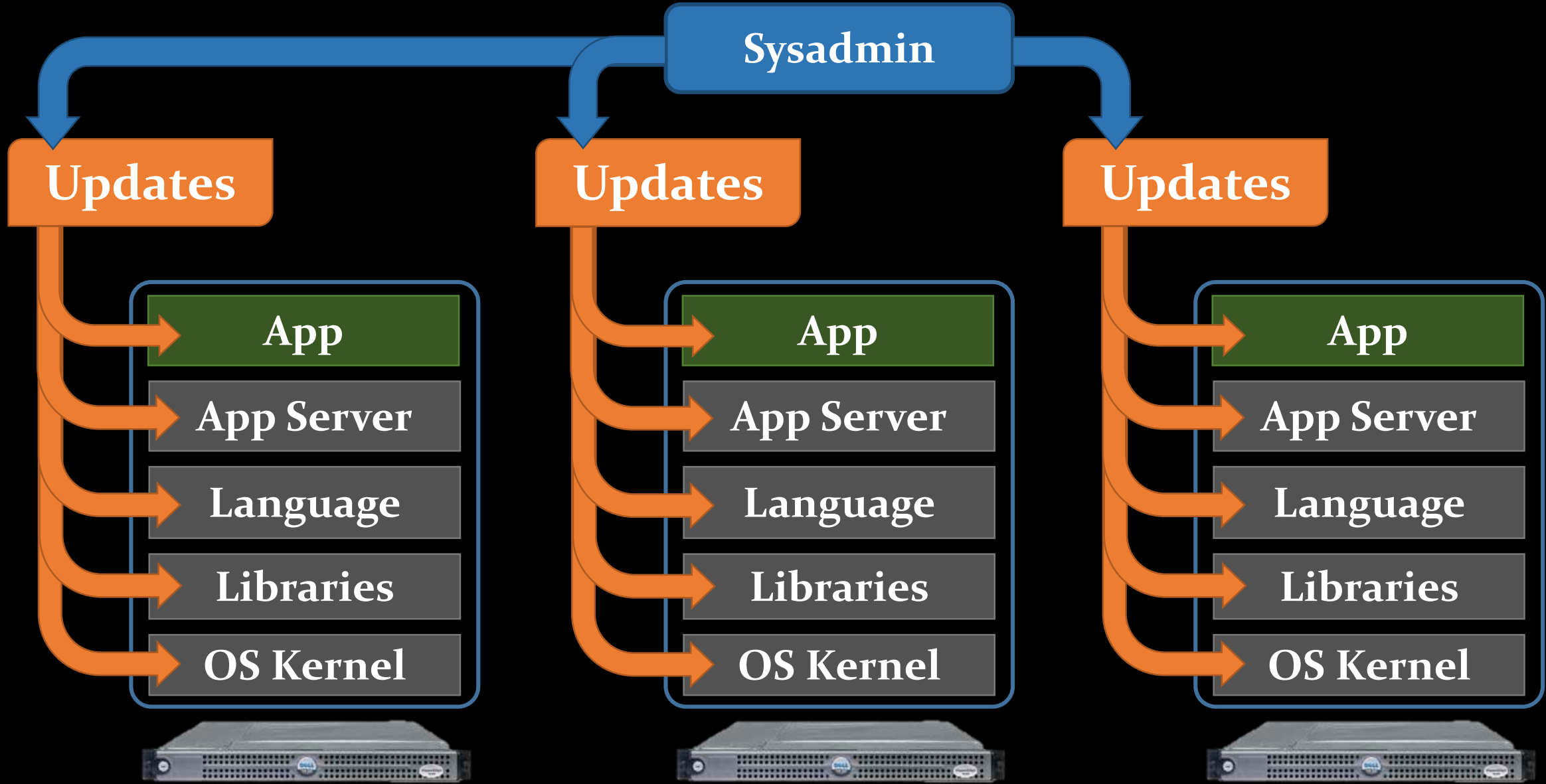


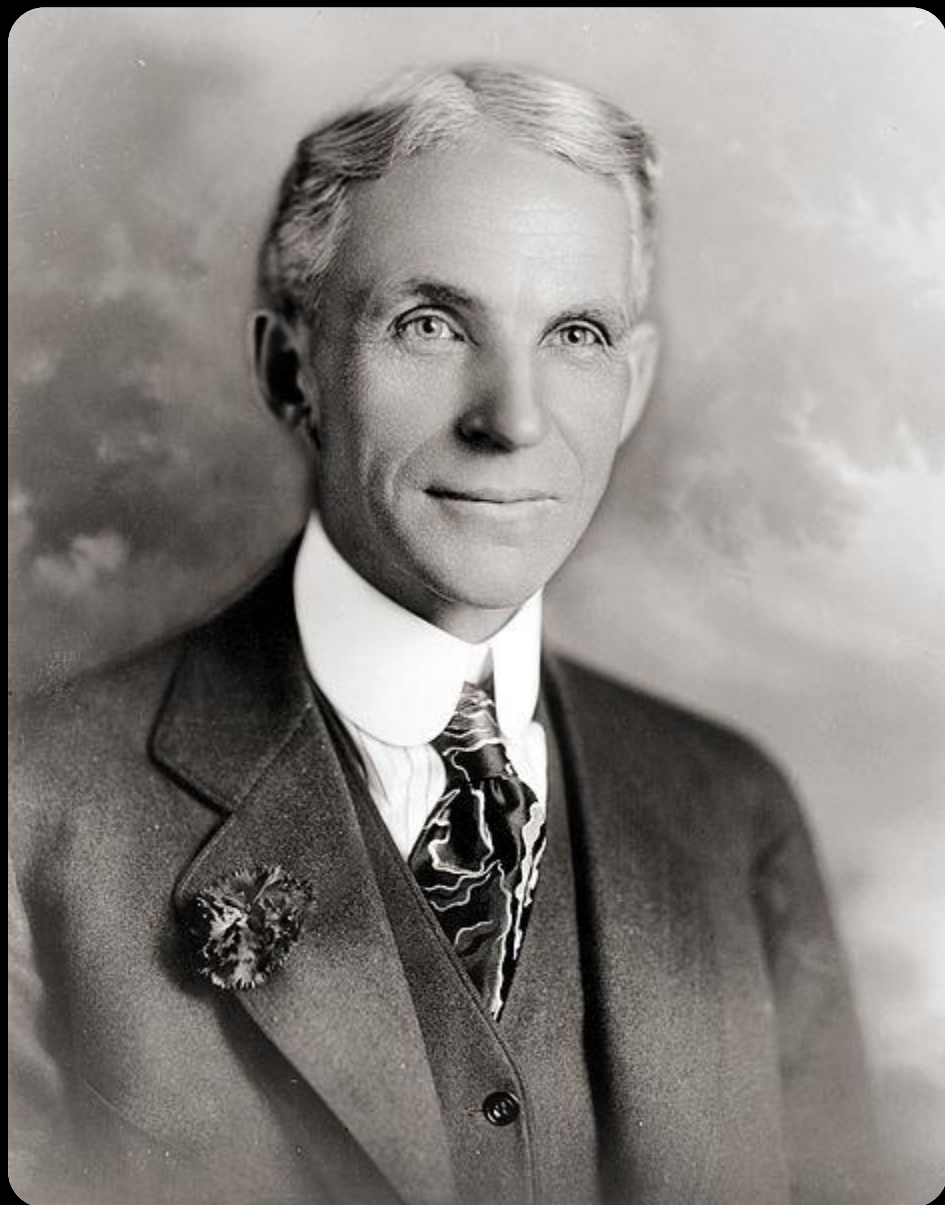




Any **difference** is a potential source of errors

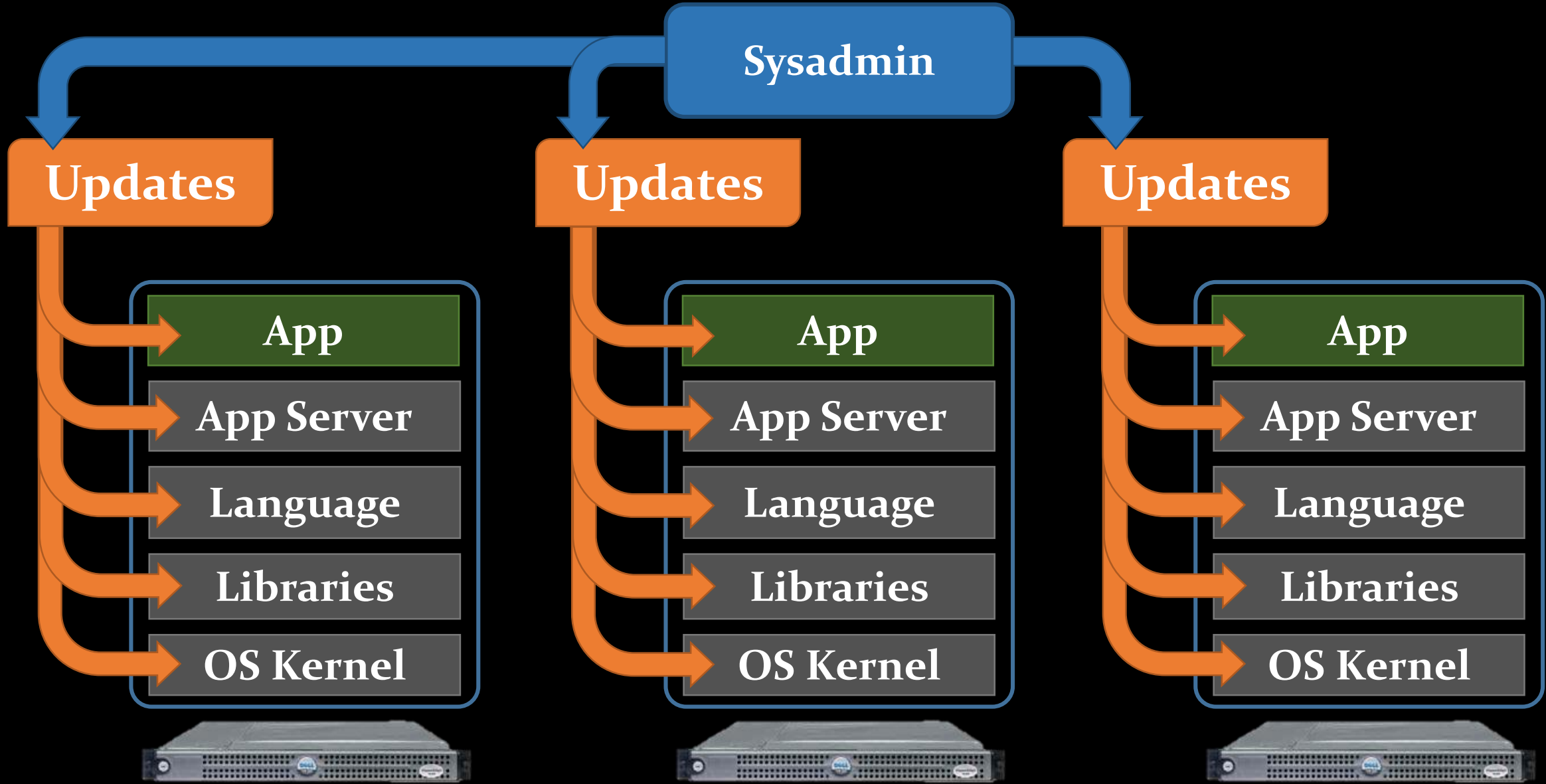


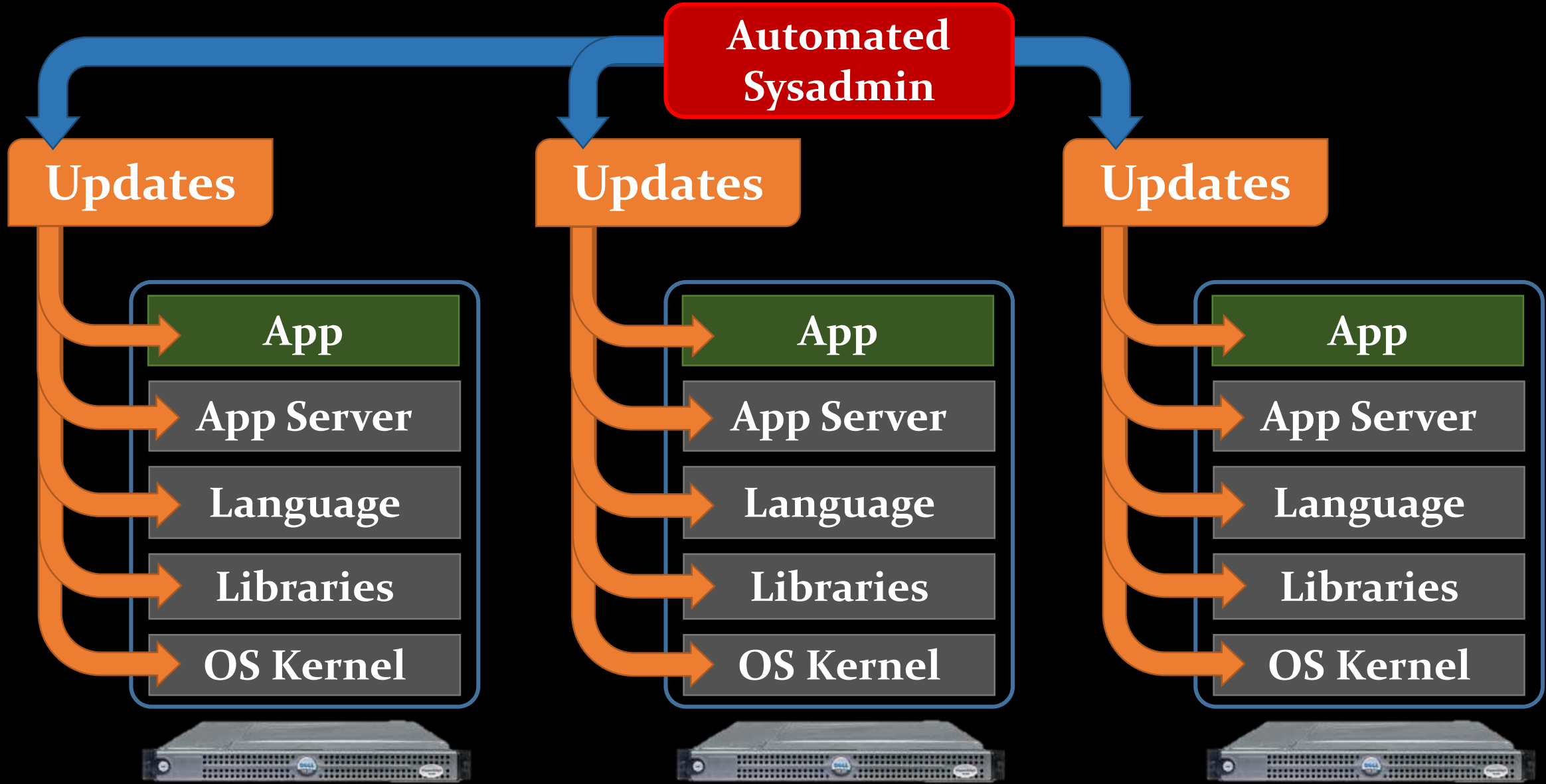




*If I had asked my  
customers what they  
wanted they would have  
said a **faster horse**.*

**Henry Ford**





fast forward to 2016 ...





*Every day, AWS adds enough server capacity to power the whole \$7B enterprise Amazon.com was in 2004. Weekends included.*

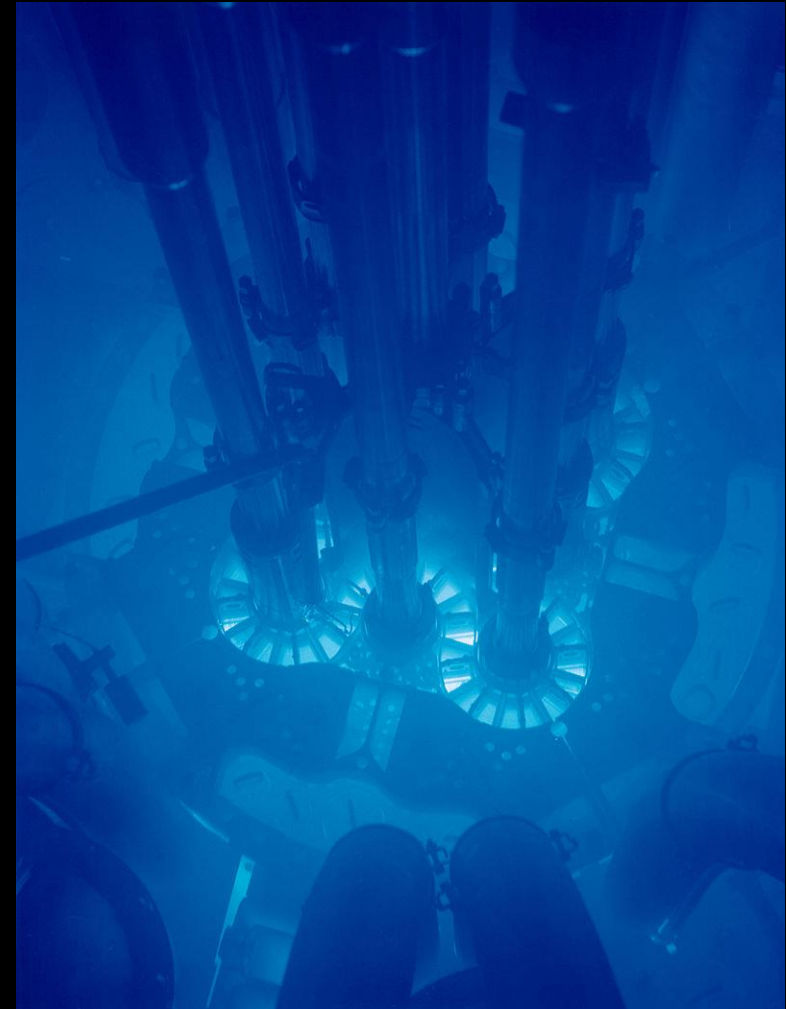


Shift to a world of **abundance**  
(no more resource scarcity)



## Control Plane

"RIAN archive\_341194\_Kursk Nuclear Power Plant" by RIA Novosti archive, image #341194, merged by yakaki - 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](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](http://commons.wikimedia.org/wiki/File:Advanced_Test_Reactor.jpg#mediaviewer/File:Advanced_Test_Reactor.jpg)

The screenshot shows the AWS Management Console interface for the EC2 service in the EU Central (Frankfurt) region. The left sidebar contains navigation options like EC2 Dashboard, INSTANCES, IMAGES, ELASTIC BLOCK STORE, and NETWORK & SECURITY. The main content area is divided into 'Resources', 'Account Attributes', and 'Create Instance' sections. The 'Resources' section provides a summary of current EC2 resources, including 1 running instance and 2 volumes. The 'Create Instance' section includes a 'Launch Instance' button and a note about the region.

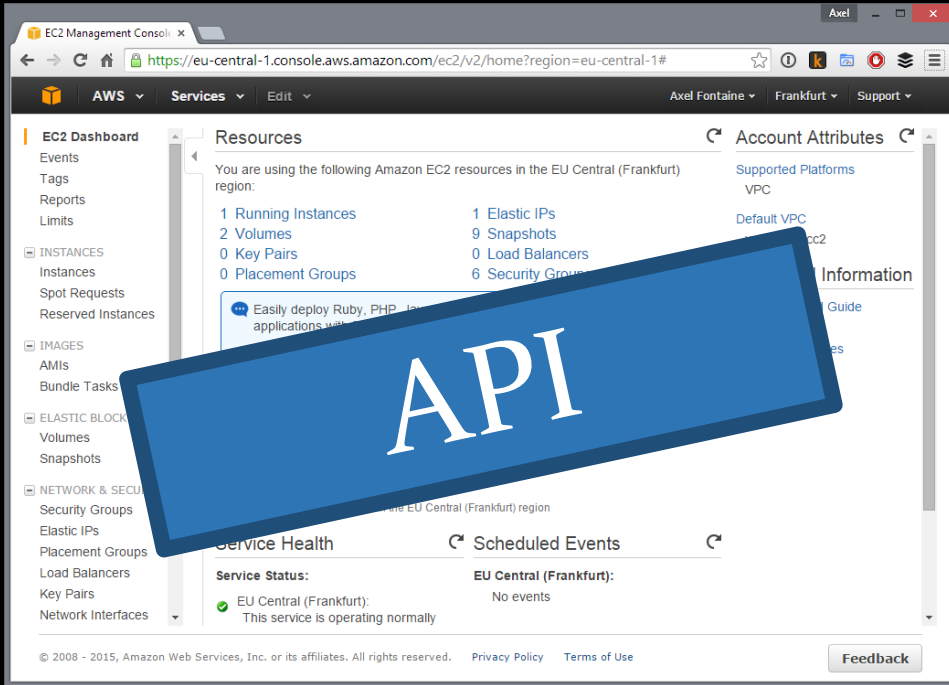
Control Plane

```

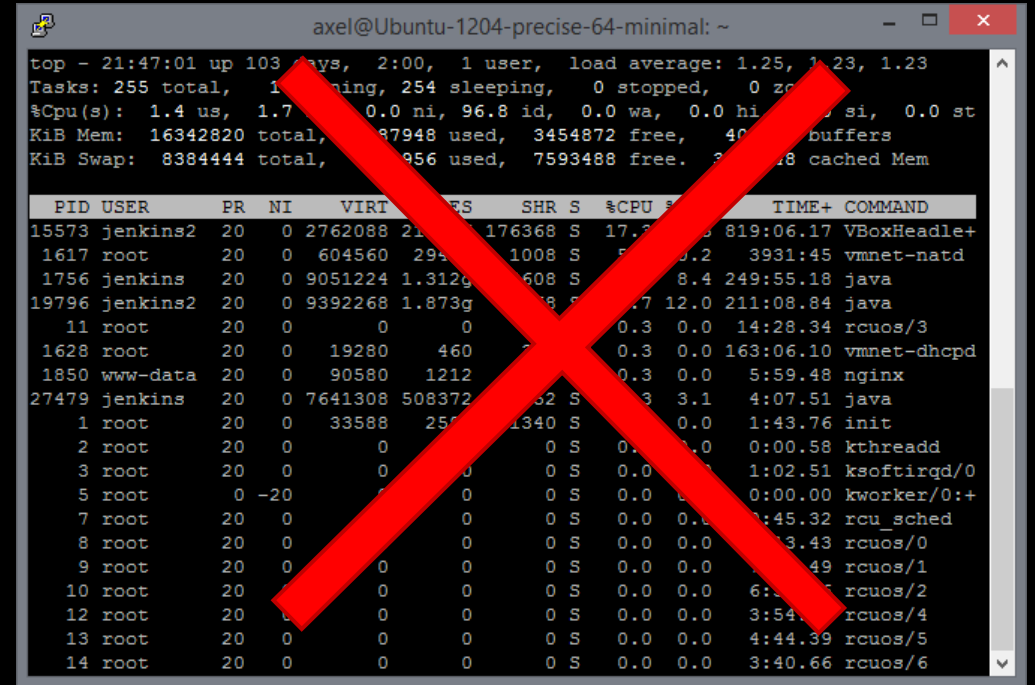
axel@Ubuntu-1204-precise-64-minimal: ~
top - 21:47:01 up 103 days, 2:00, 1 user, load average: 1.25, 1.23, 1.23
Tasks: 255 total, 1 running, 254 sleeping, 0 stopped, 0 zombie
%Cpu(s): 1.4 us, 1.7 sy, 0.0 ni, 96.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 16342820 total, 12887948 used, 3454872 free, 400984 buffers
KiB Swap: 8384444 total, 790956 used, 7593488 free. 3392548 cached Mem

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
15573 jenkins2  20   0 2762088 215316 176368 S  17.3  1.3  819:06.17 VBoxHeadle+
1617  root     20   0 604560 29448  1008 S   5.0  0.2  3931:45 vmnet-natd
1756  jenkins  20   0 9051224 1.312g 8608 S   0.7  8.4  249:55.18 java
19796 jenkins2  20   0 9392268 1.873g 8368 S   0.7 12.0  211:08.84 java
  11  root     20   0 0 0 0 S   0.3  0.0  14:28.34 rcuos/3
1628  root     20   0 19280 460 324 S   0.3  0.0  163:06.10 vmnet-dhcpd
1850  www-data 20   0 90580 1212 928 S   0.3  0.0  5:59.48 nginx
27479 jenkins  20   0 7641308 508372 12652 S  0.3  3.1  4:07.51 java
  1  root     20   0 33588 2592 1340 S  0.0  0.0  1:43.76 init
  2  root     20   0 0 0 0 S  0.0  0.0  0:00.58 kthread
  3  root     20   0 0 0 0 S  0.0  0.0  1:02.51 ksoftirqd/0
  5  root     0 -20 0 0 0 S  0.0  0.0  0:00.00 kworker/0:++
  7  root     20   0 0 0 0 S  0.0  0.0  20:45.32 rcu_sched
  8  root     20   0 0 0 0 S  0.0  0.0  9:13.43 rcuos/0
  9  root     20   0 0 0 0 S  0.0  0.0  7:35.49 rcuos/1
 10  root     20   0 0 0 0 S  0.0  0.0  6:36.96 rcuos/2
 12  root     20   0 0 0 0 S  0.0  0.0  3:54.80 rcuos/4
 13  root     20   0 0 0 0 S  0.0  0.0  4:44.39 rcuos/5
 14  root     20   0 0 0 0 S  0.0  0.0  3:40.66 rcuos/6
  
```

Data Plane

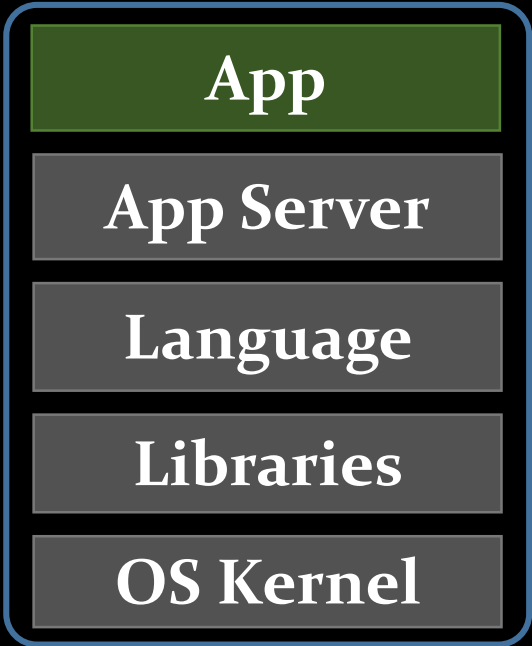
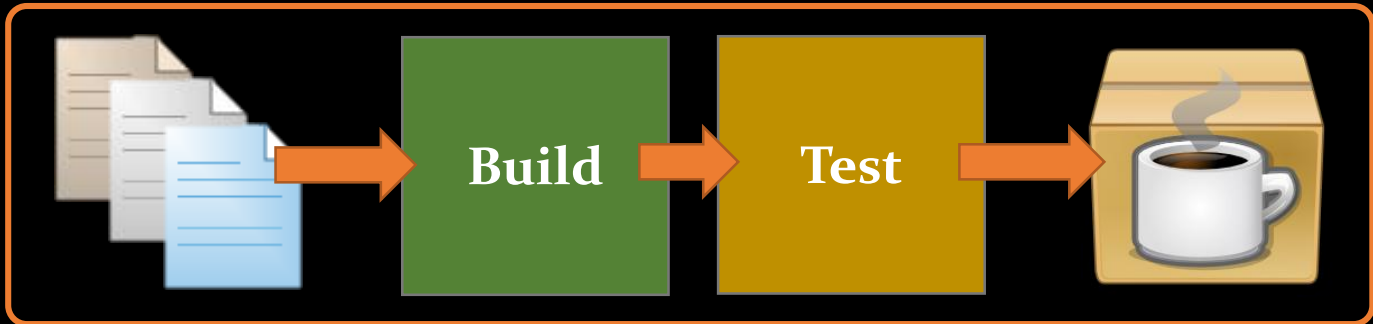


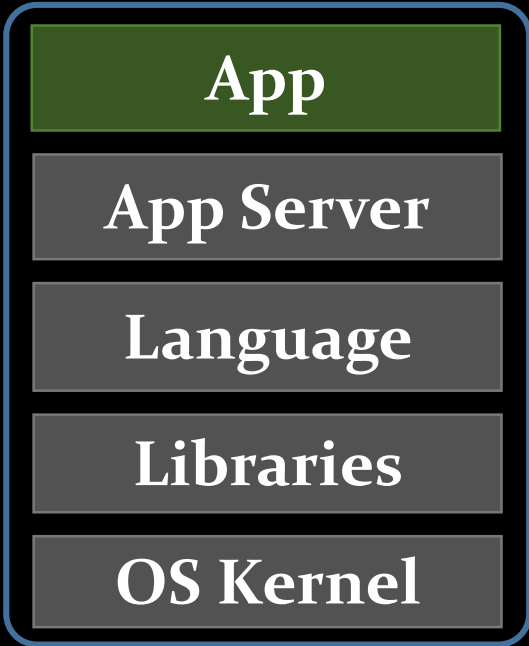
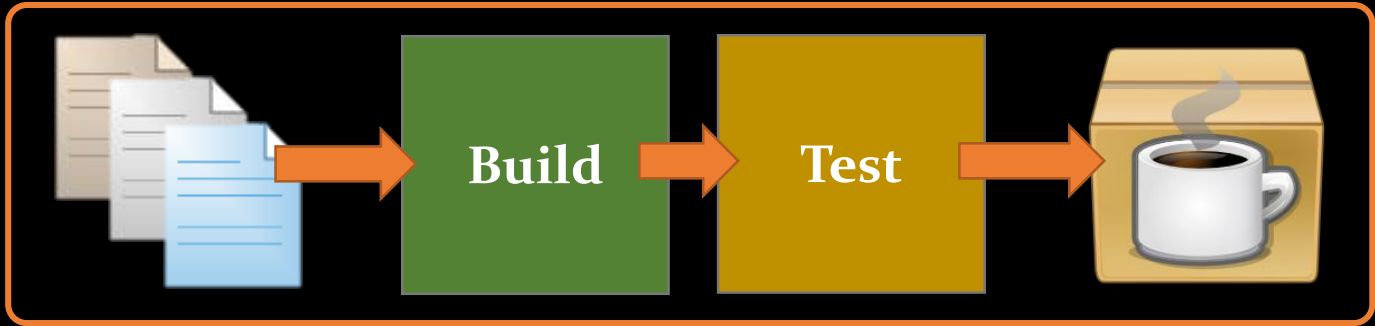
Automated  
Provisioning



Cost-driven  
Architectures

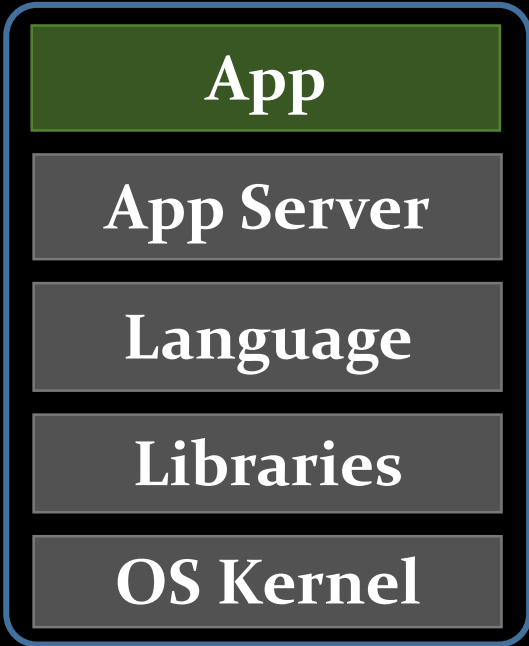
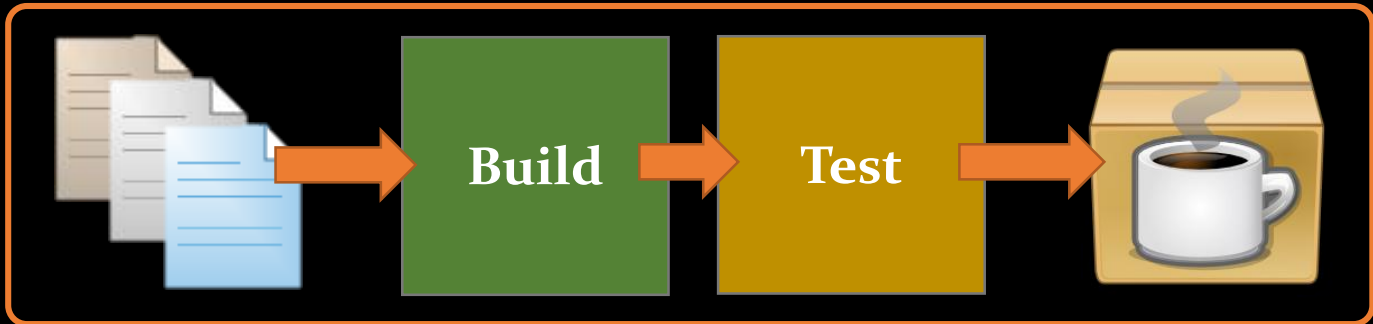
it is time to **rethink** the faster horse

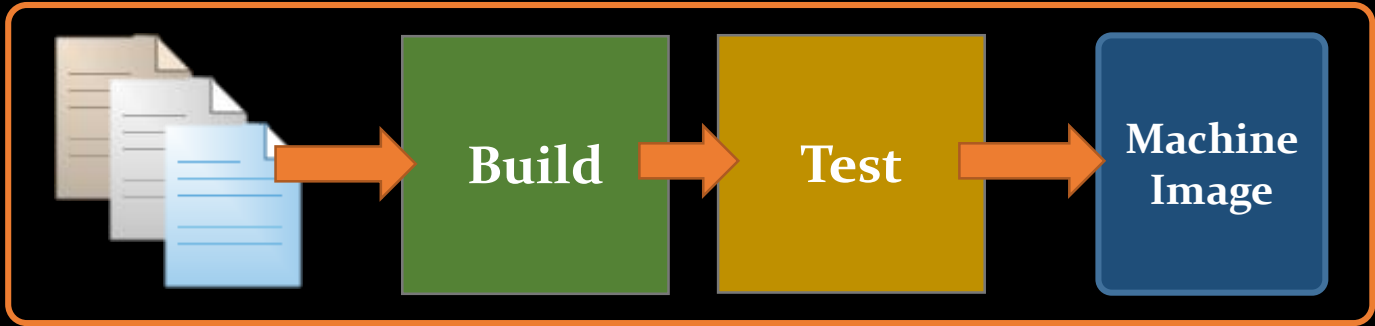




Undifferentiated  
Heavy lifting







Machine  
Image



Machine  
Image



Machine  
Image



~~Unifies~~

Machine Image



Machine Image



Machine Image



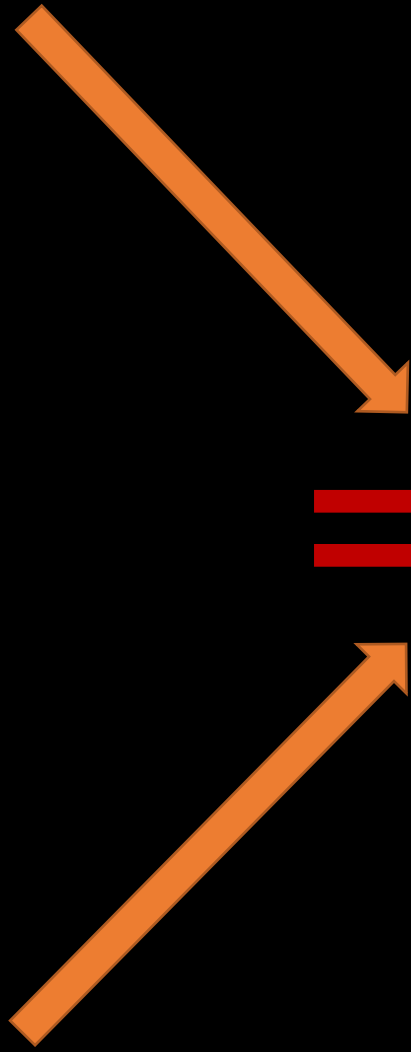
but there is one **big** problem left ...

Machine  
Image



Network Cable

Multiple  
GB



Network Cable



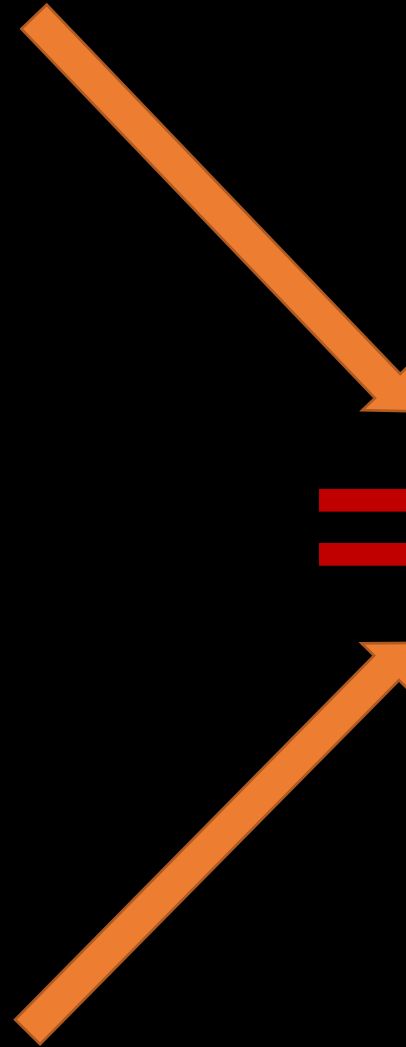
*Running servers in production should be like going **backpacking**. You take the bare minimum with you. Anything else is going to hurt.*

**A Wise Man**



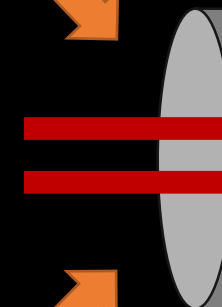
what is really adding **business value** ???

Machine  
Image

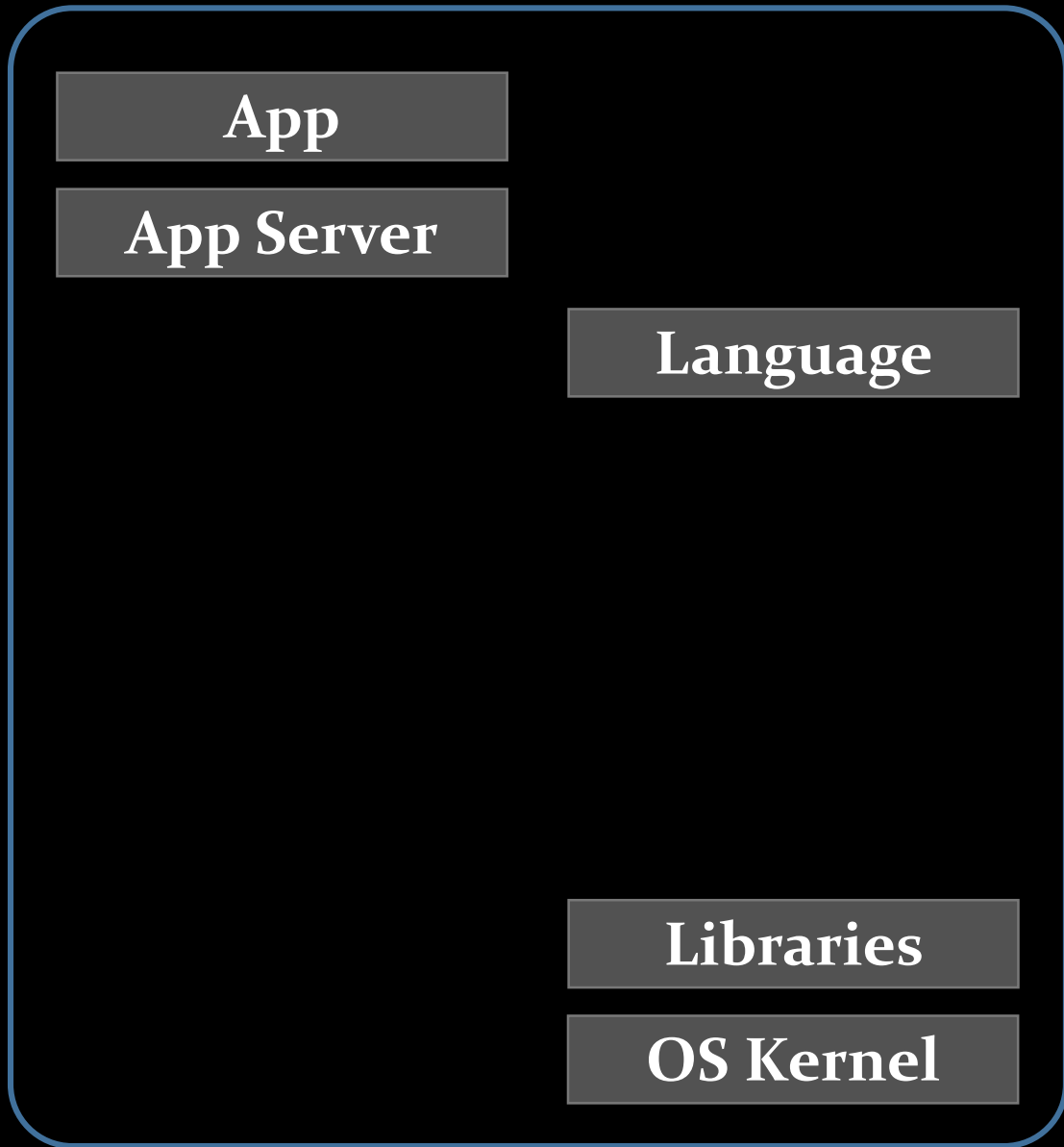


Network Cable

Machine  
Image



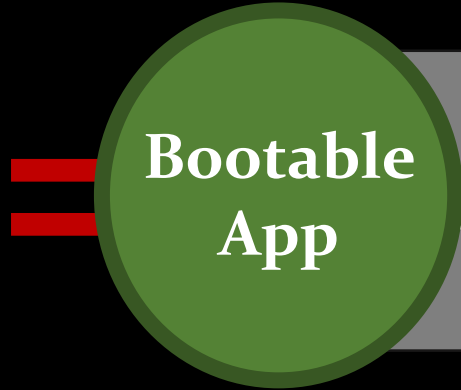
Network Cable



15  
MB

Multiple  
GB

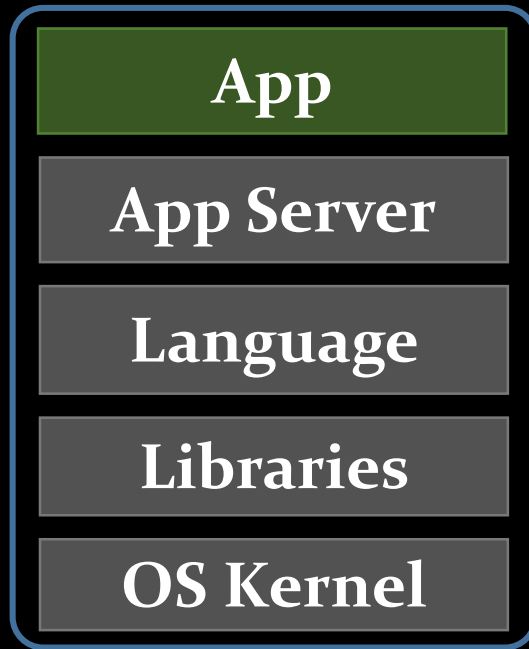
15  
MB



Network Cable

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



C ~~RU~~ ID

for servers is **dead!**

high **uptime** is a liability



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



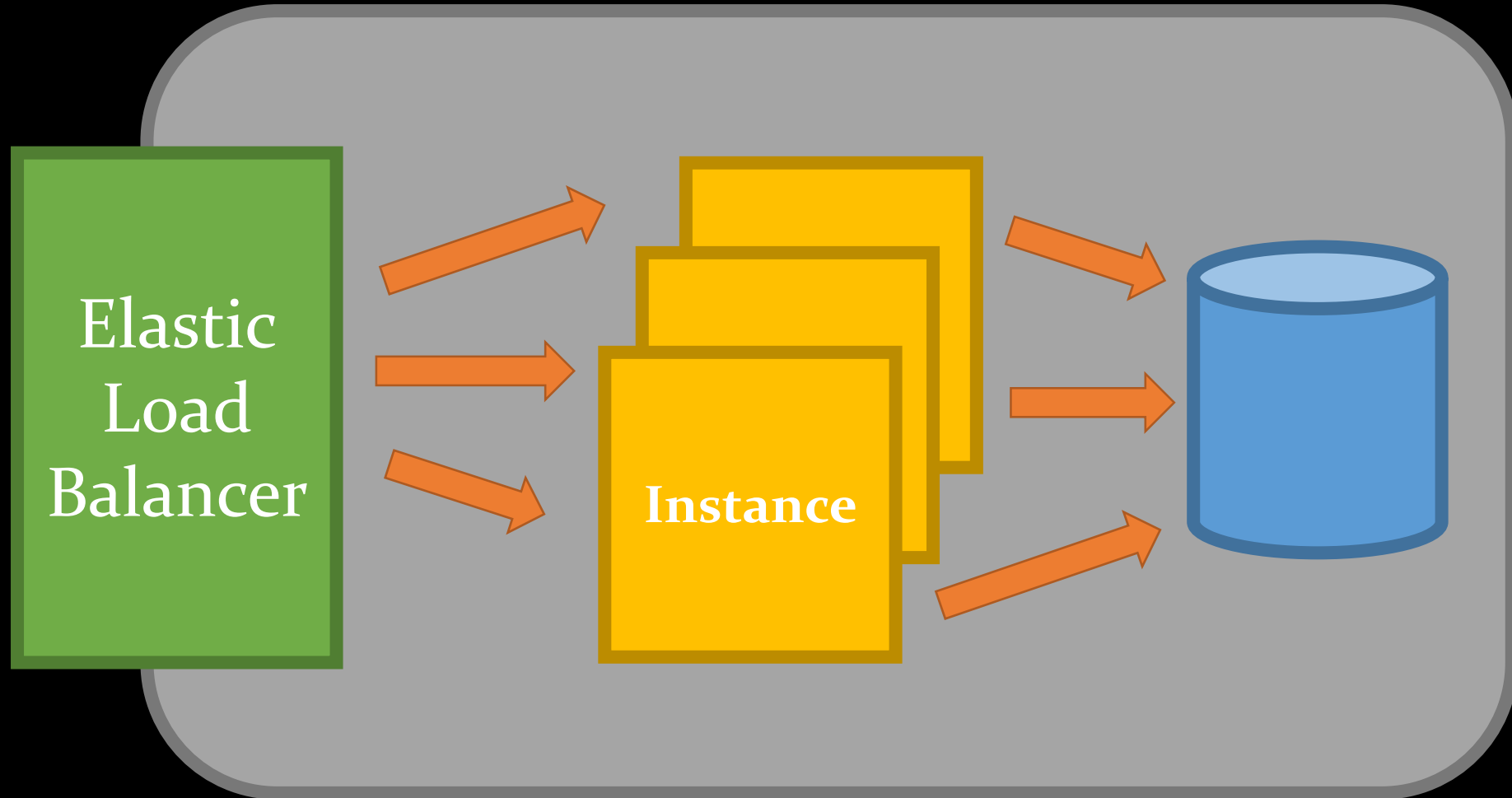
```
axel@Ubuntu-1204-precise-64-minimal:~$ uptime -p  
up 14 weeks, 5 days, 2 hours, 47 minutes  
axel@Ubuntu-1204-precise-64-minimal:~$
```

**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** ?



Complexity is the Enemy of Security

# What about **security** ?

Bootable  
App

- 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

# what about configuration ???

- Bake as much configuration as possible for all environments directly in the Bootable App

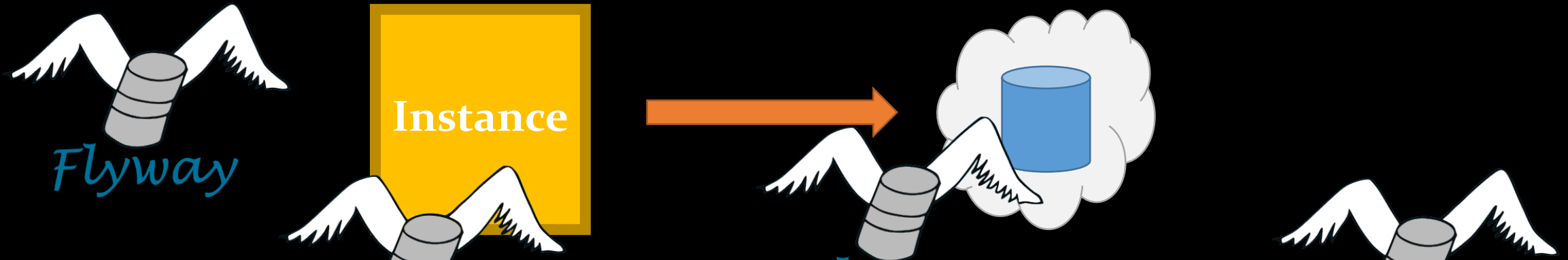
Key	Value
JDBC_URL	jdbc:...
ENV	prod

- Use environment detection and auto-configuration
- Pass remaining configuration at startup and expose it as environment variables

Bootable App



# what about the database ???



- Keep persistent state out of the instance, including the database

- Use one of the many good hosted solutions available like Amazon RDS or Google Cloud SQL

- Use a database migration tool to update the schema on application startup



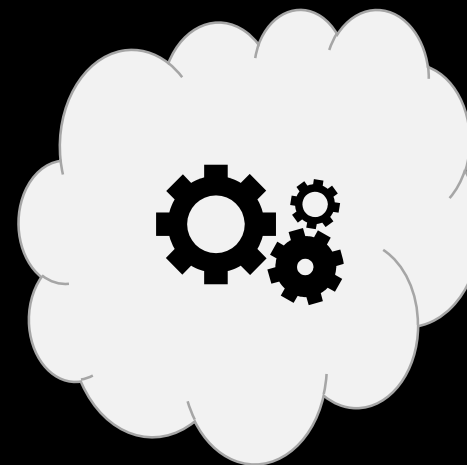
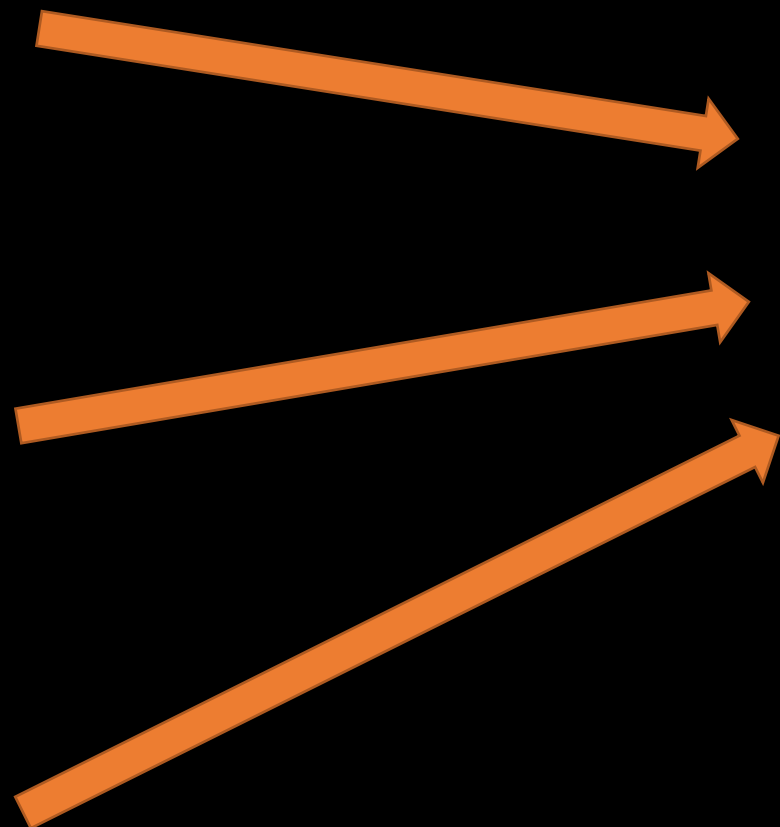
what about the logs ???  
~~ssh me@myserver1~~  
~~tail -f server.log~~



~~ssh me@myserver2~~  
~~tail -f server.log~~



~~ssh me@myserver3~~  
~~tail -f server.log~~



log server

where logs can be

- aggregated
- stored and backuped
- indexed
- searched



what about **sessions** ???

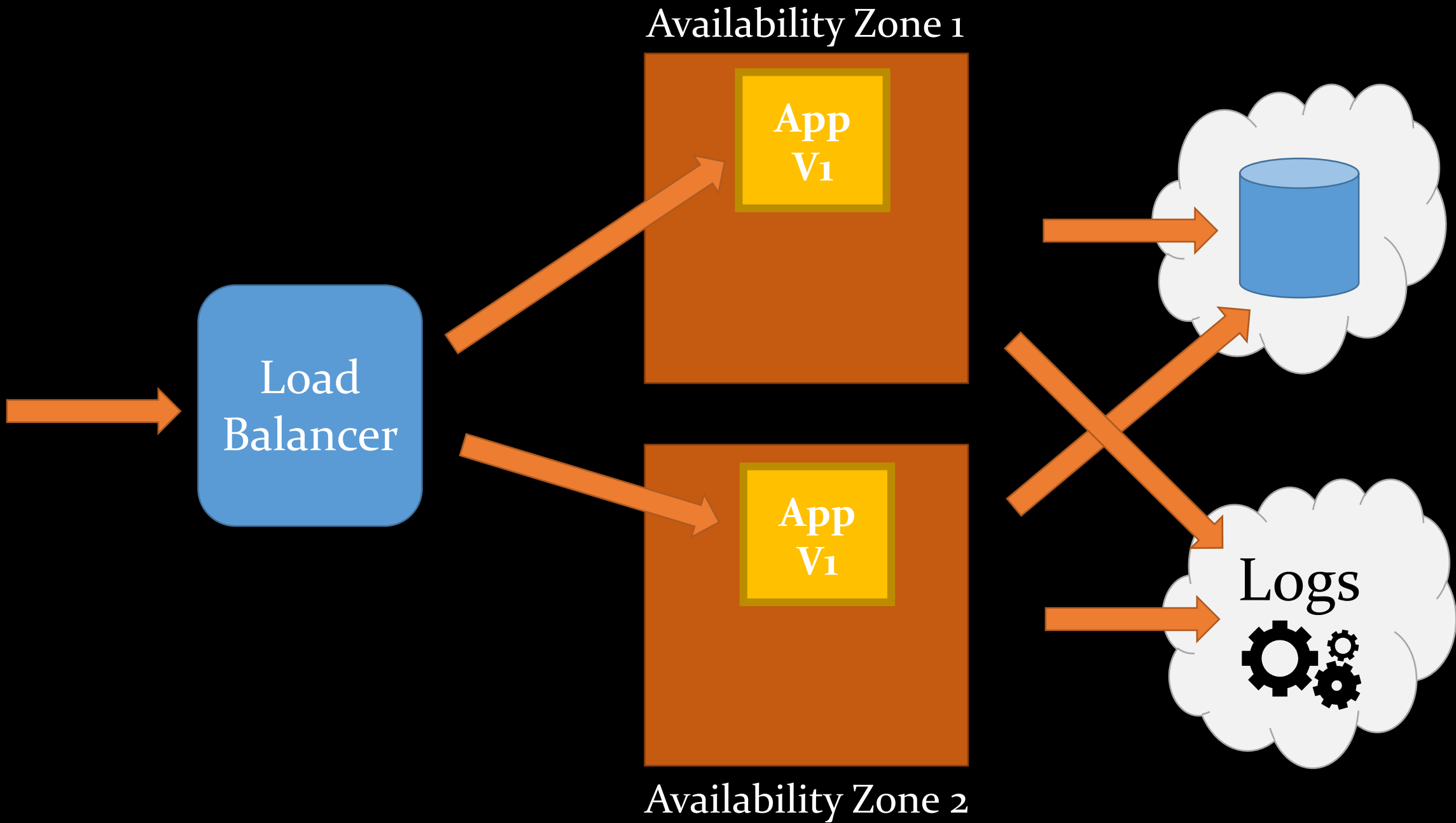


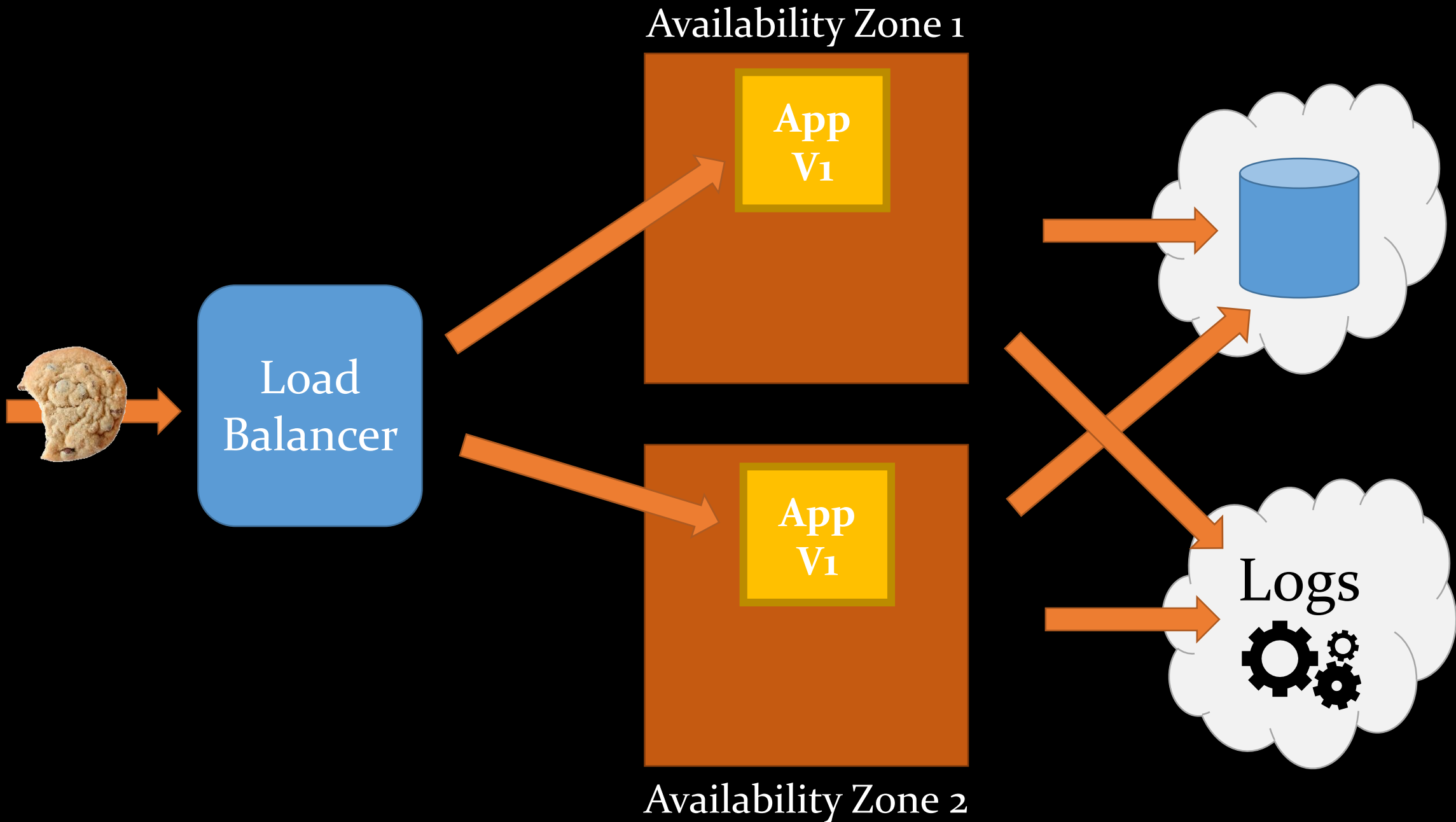
Instance

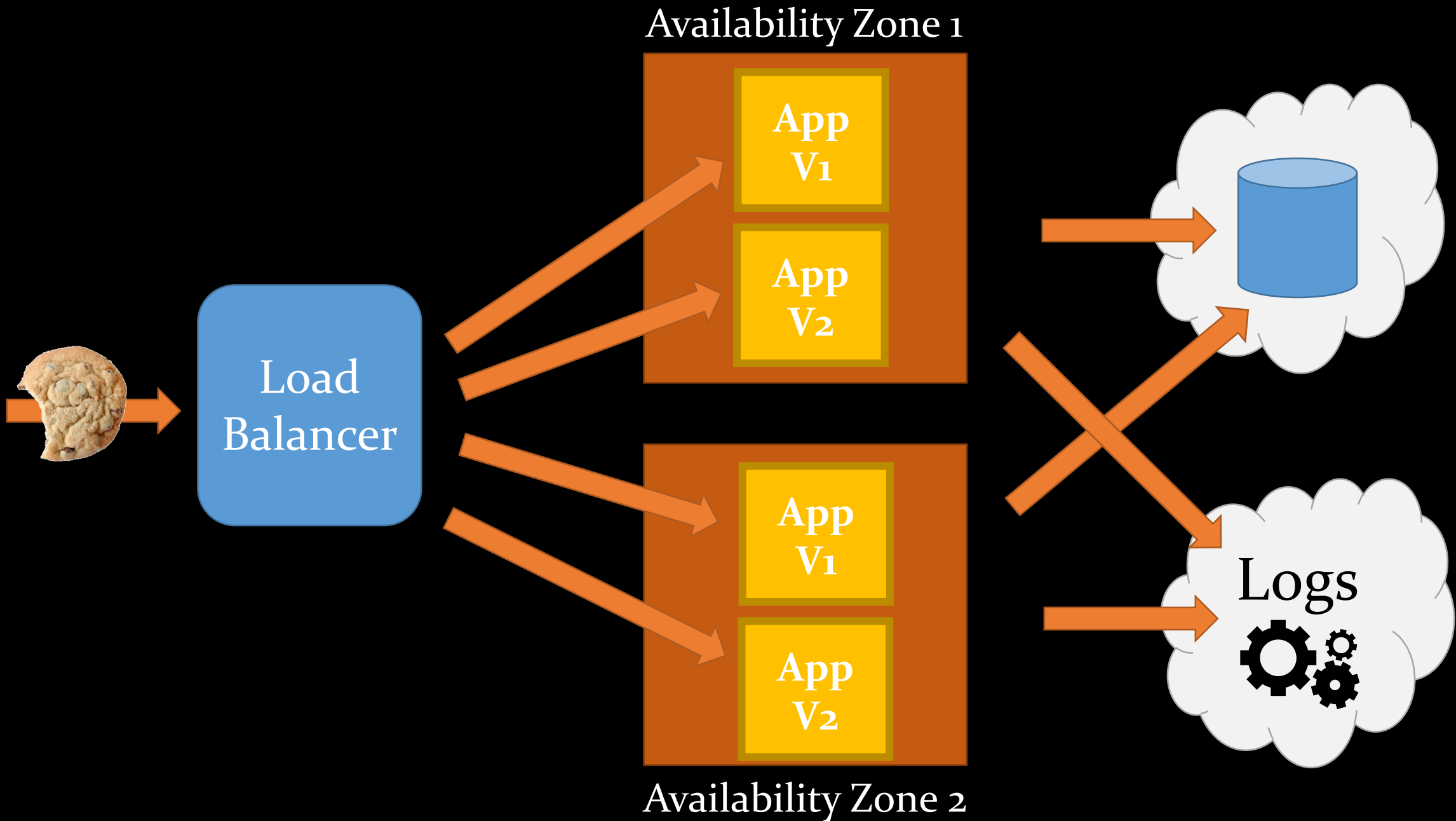
Keep session in an encrypted and signed **cookie**

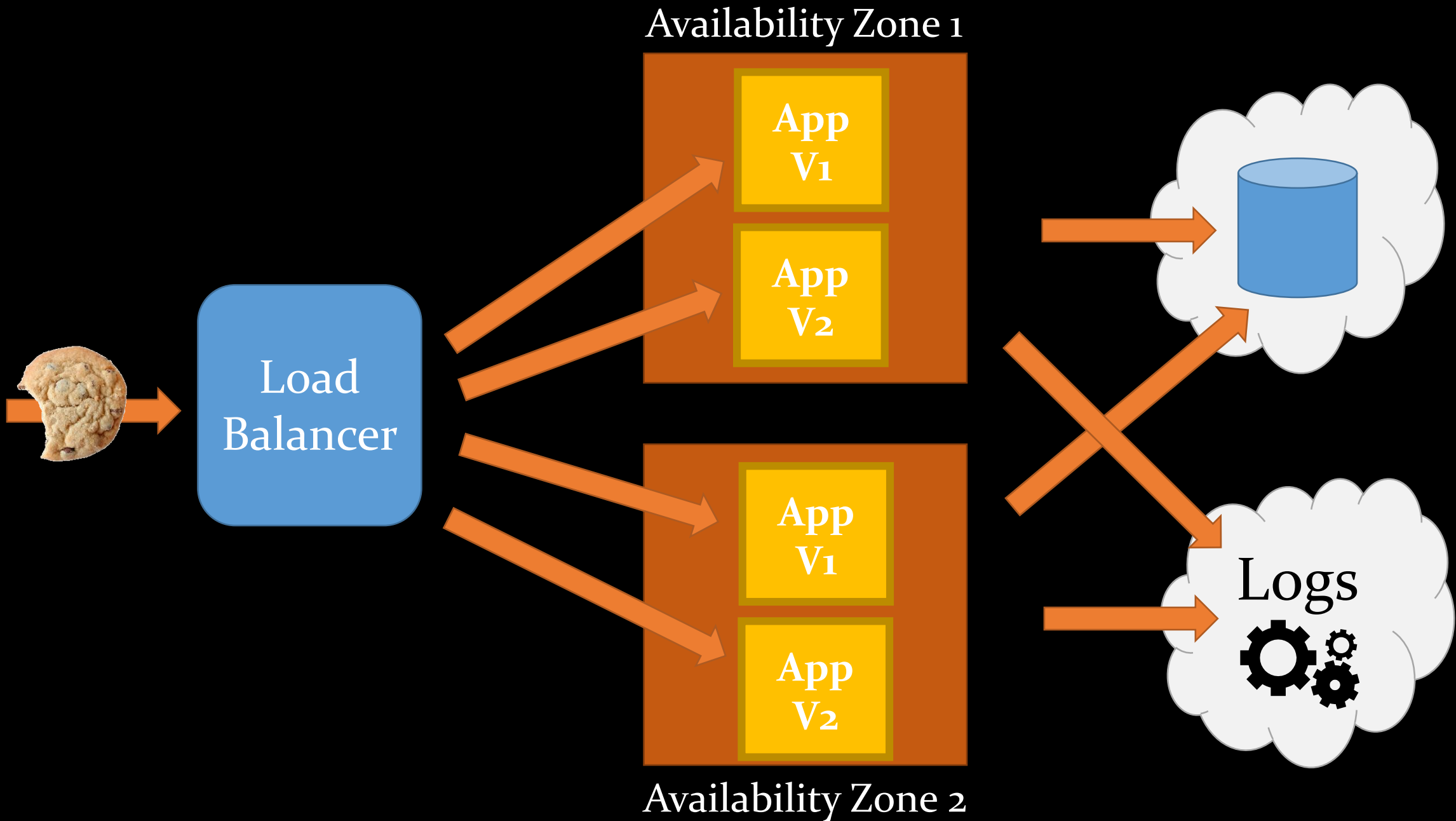
- avoids session timeouts
- avoids server clustering & session replication
- avoids sticky sessions & server affinity

what about rolling out **new versions** ???









what about **containers** ???  
(as in OS-level virtualization)

# understanding modern CPUs

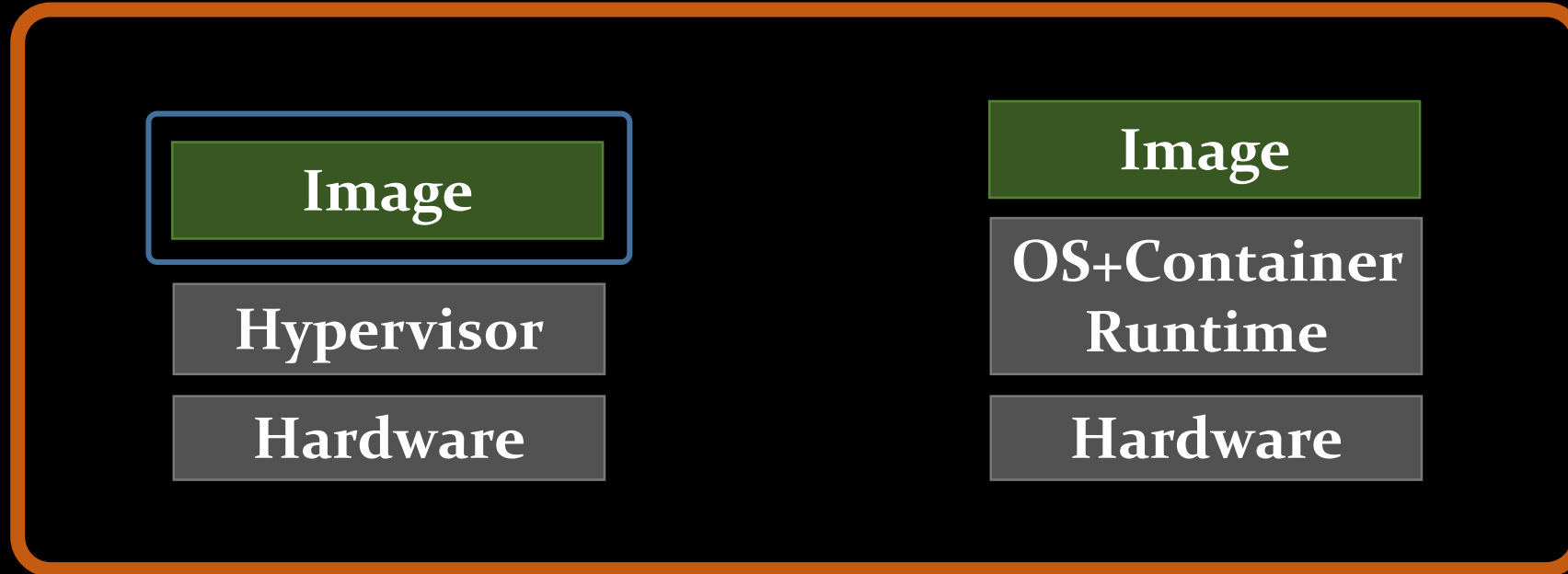


Both Intel and AMD have hardware support for virtualization

- isolation
- performance penalty



on prem

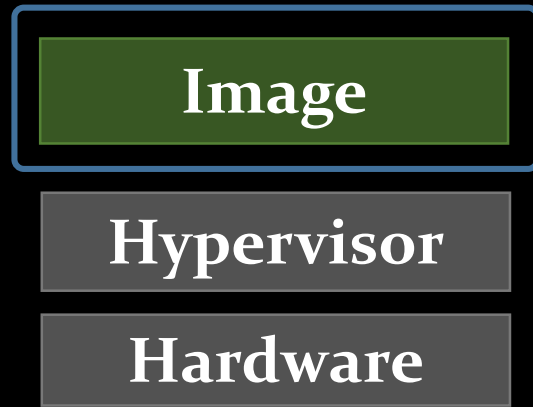


VM

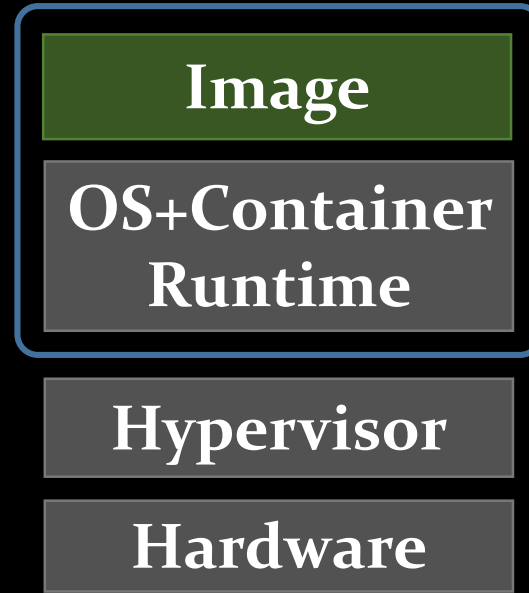
Container

*your  
responsibility*

cloud



VM

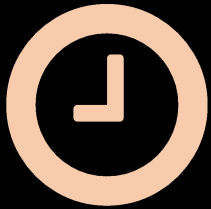


Container

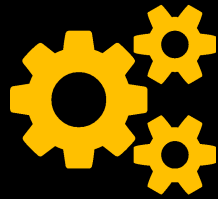
# cloud



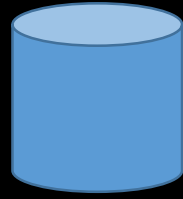
container  
images



container  
scheduling



containers



container  
volumes

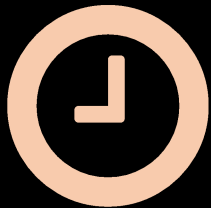


container  
networking

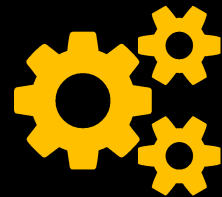
*your  
responsibility*



machine  
images



instance  
scheduling



instances



instance  
volumes



instance  
networking

*cloud  
responsibility*



1 month of t2.nano



1 hour of t2.nano

# cloud

Only makes sense if you cannot afford  
**0.5 Rappen/hour**  
granularity



container images    container scheduling    containers    container volumes    container networking

**your responsibility**

machine images    instance scheduling    instances    instance volumes    instance networking

**cloud responsibility**

summary



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

**Classic Mistake:** Build per Environment



## Bootable App

- One **immutable** unit
  - **Regenerated** after every change
  - **Promoted** from Environment to Environment
  - Use **Minimal** Images
  - Focus on **Cost** in your architecture
- Classic Mistake:** Build per Environment



boxfuse

[boxfuse.com](https://boxfuse.com)

**THANKS**



AXEL FONTAINE

 @axelfontaine

I'LL BE BACK



[boxfuse.com](https://boxfuse.com)