Architecting for the Cloud

John Chang (張書源)
Technology Evangelist
Amazon Web Services

amazon web services

Cloud Architecture Principles

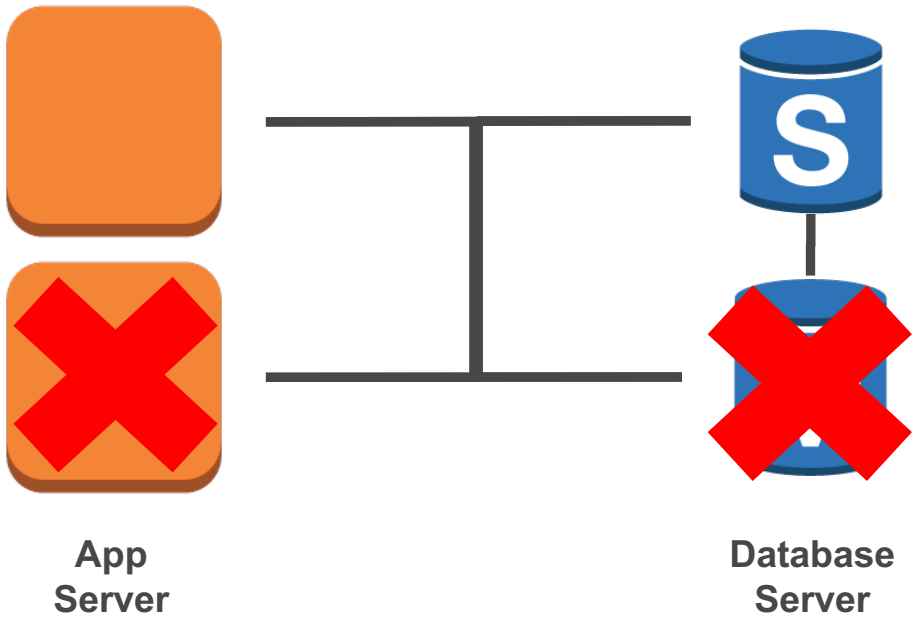
"Everything fails, all the time."

Werner Vogels, CTO, Amazon.com

amazon web services

Cloud Architecture Principles

1. Design for failure and nothing will fail

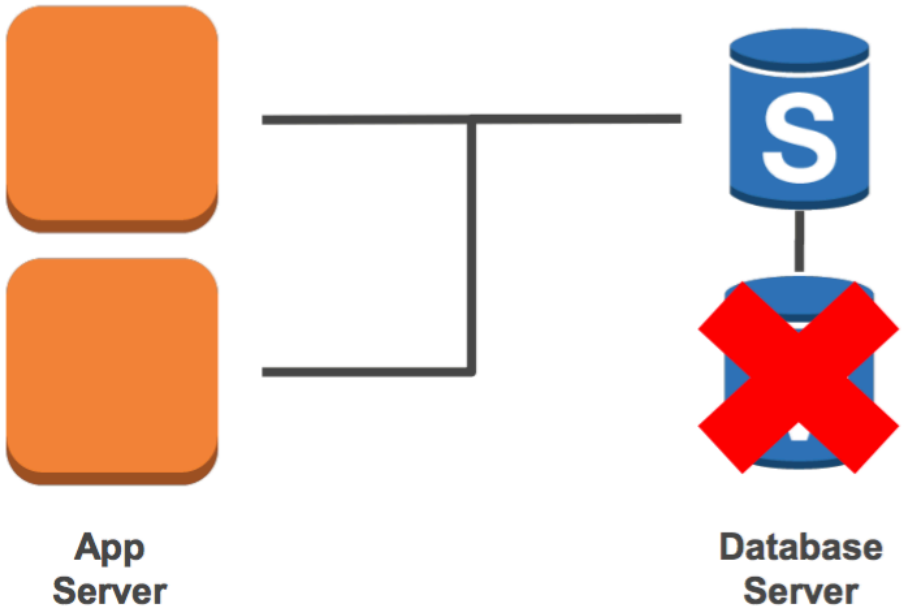


Goal: Applications should continue to function even if the underlying application component fails, communication is lost or physical hardware fails, is removed/replaced.

amazon web services

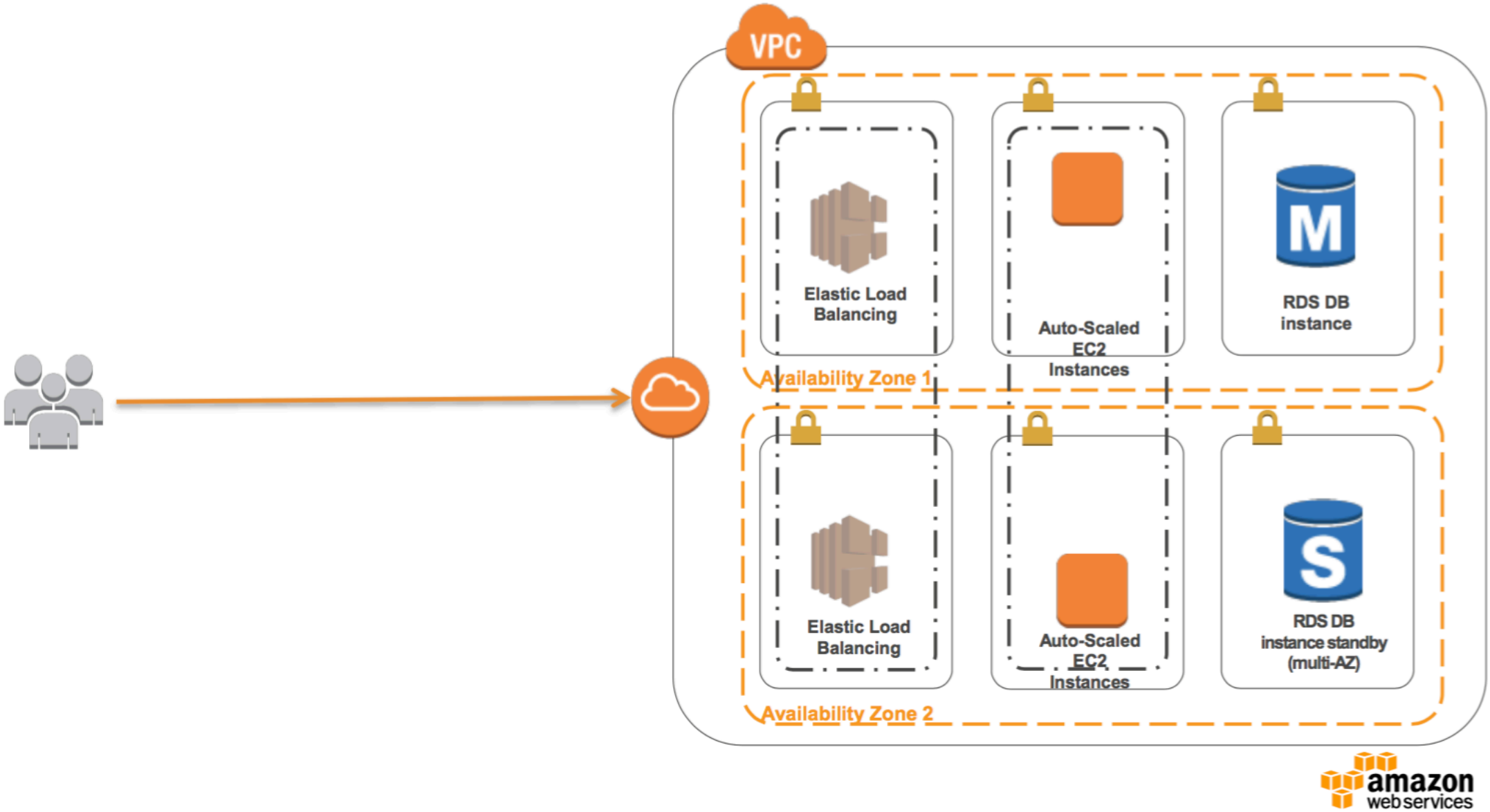
Cloud Architecture Principles

1. Design for failure and nothing will fail



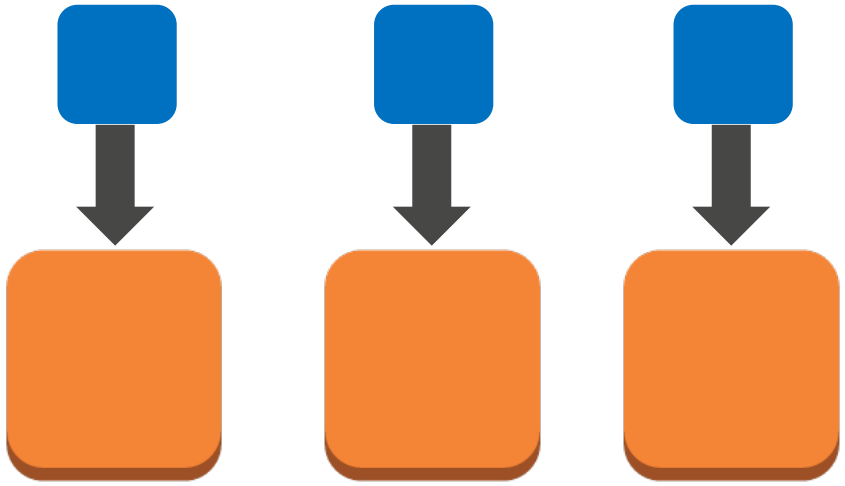
- Have a coherent backup and restore strategy for your data
- Build process that resume on reboot
- Keep pre-configured and pre-optimized virtual images to support the above on launch/boot
- Allow the state of the system to re-sync by reloading messages from queues
- Avoid in-memory sessions or stateful user context, move that to data stores

amazon web services


VPC
Elastic Load Balancing
Auto-Scaled EC2 Instances
RDS DB instance
Availability Zone 1
Elastic Load Balancing
Auto-Scaled EC2 Instances
RDS DB instance standby (multi-AZ)
Availability Zone 2
amazon web services

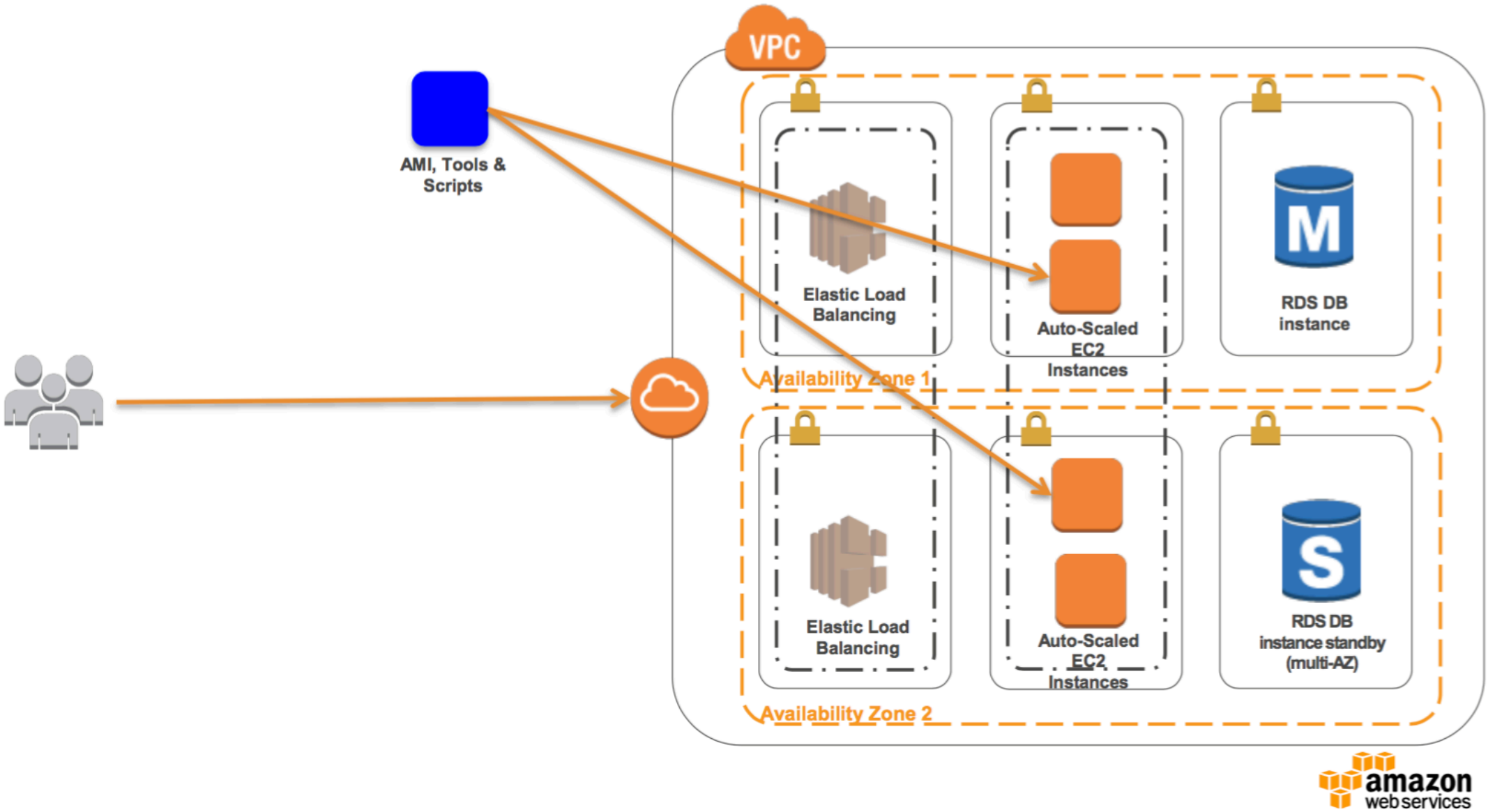
Cloud Architecture Principles

2. Embrace Elasticity & Automate



- Do not assume health, availability or fixed location of components (e.g. fixed IP)
- Automate installation and configuration of environment
- Favor dynamic configuration

amazon web services

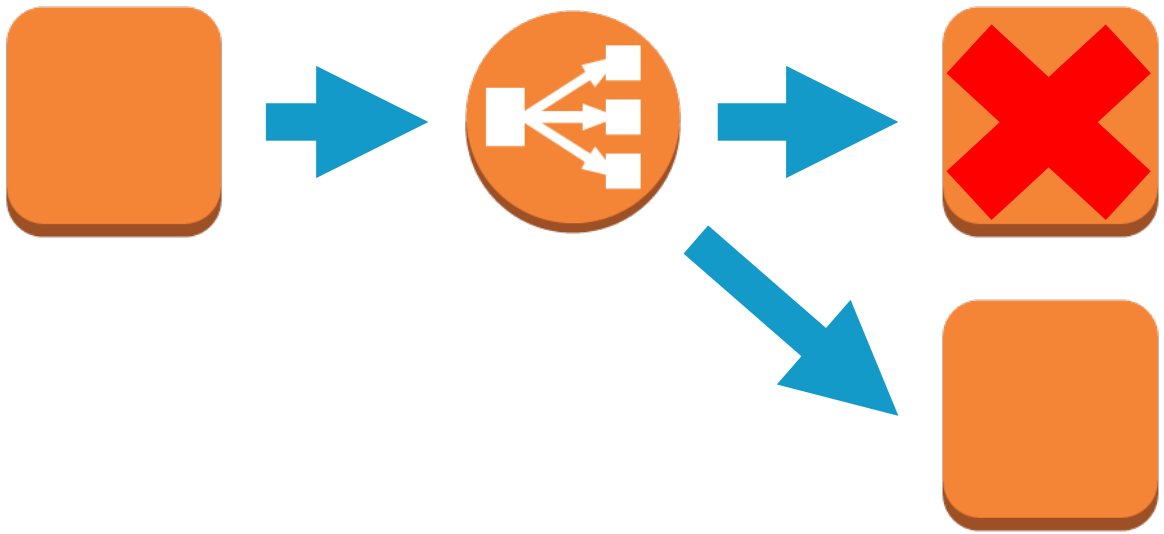

VPC
AMI, Tools &
Scripts
Elastic Load
Balancing
Auto-Scaled
EC2
Instances
RDS DB
instance
Availability Zone 1
Elastic Load
Balancing
Auto-Scaled
EC2
Instances
RDS DB
instance standby
(multi-AZ)
Availability Zone 2
amazon
web services

Cloud Architecture Principles

3. Loose coupling sets you free

Cloud Architecture Principles

3. Loose coupling sets you free

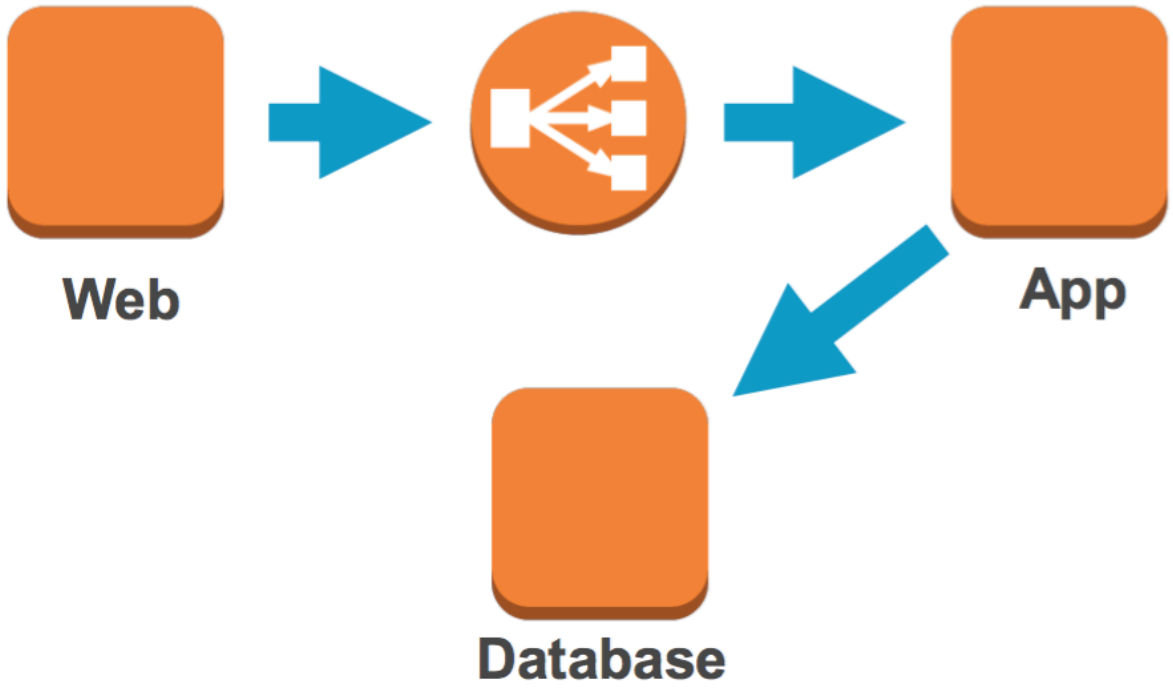


- Design architectures with independent components
- Design every component as a black box
- Load balance clusters

amazon web services

Cloud Architecture Principles

3. Loose coupling sets you free

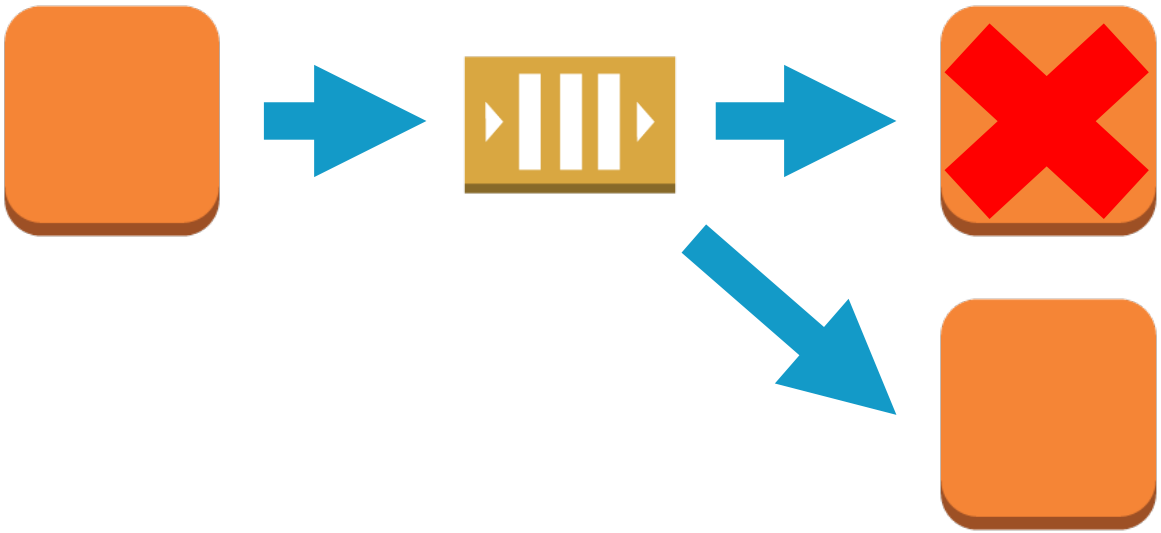


- Separate application into independent tiers

amazon web services

Cloud Architecture Principles

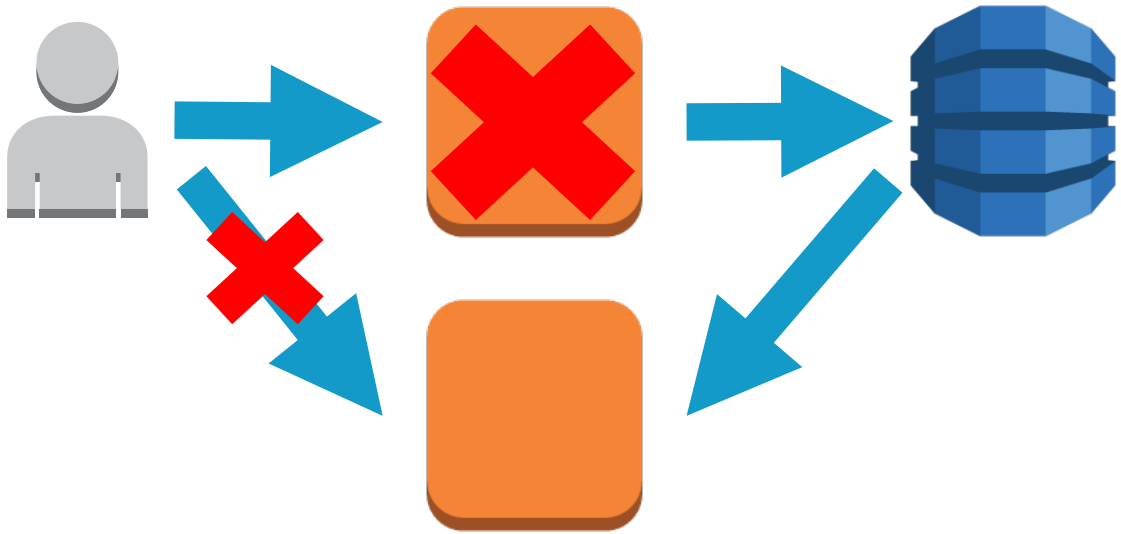
3. Loose coupling sets you free



- Use queues to pass messages between components

Cloud Architecture Principles

4. Become Stateless

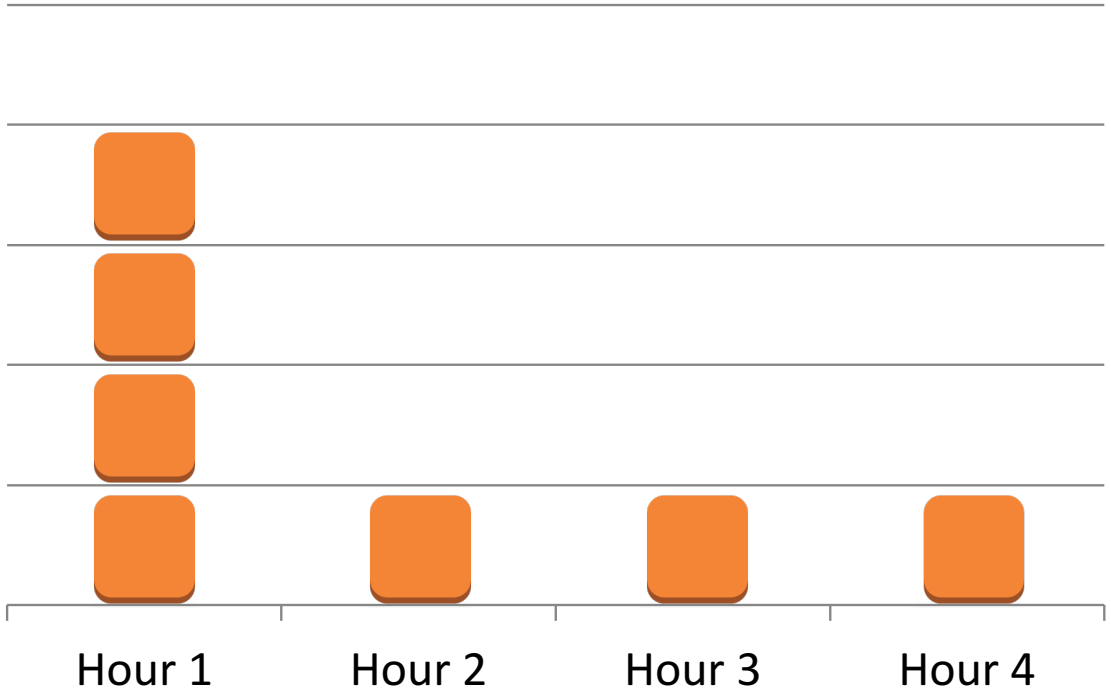


- Don’t store state in server
- Leverage services to hold state information
- Application functions regardless of which application node processes the request

amazon web services

Cloud Architecture Principles

5. Think Parallel

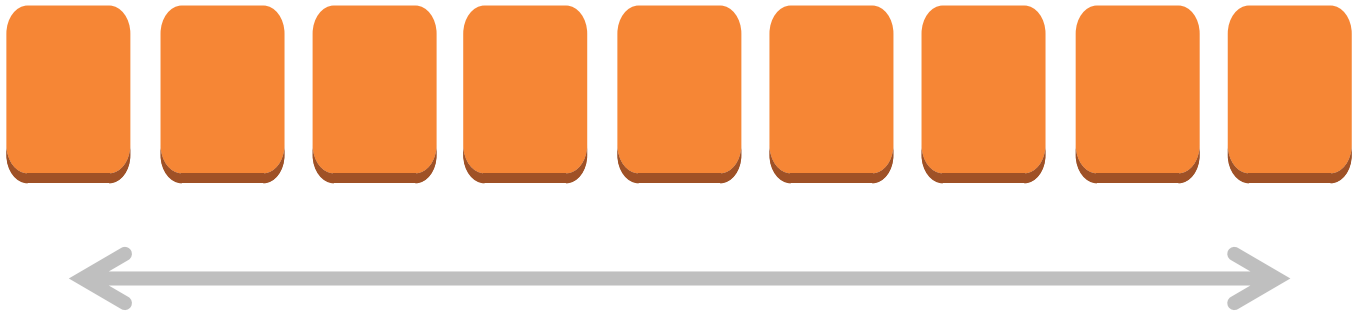


- One Server working for Four hours costs the same as Four servers working for One hour
- Combine with elasticity to increase capacity when you need it most

amazon web services

Having done that…

Having decomposed into
small, loosely coupled,
stateless building blocks

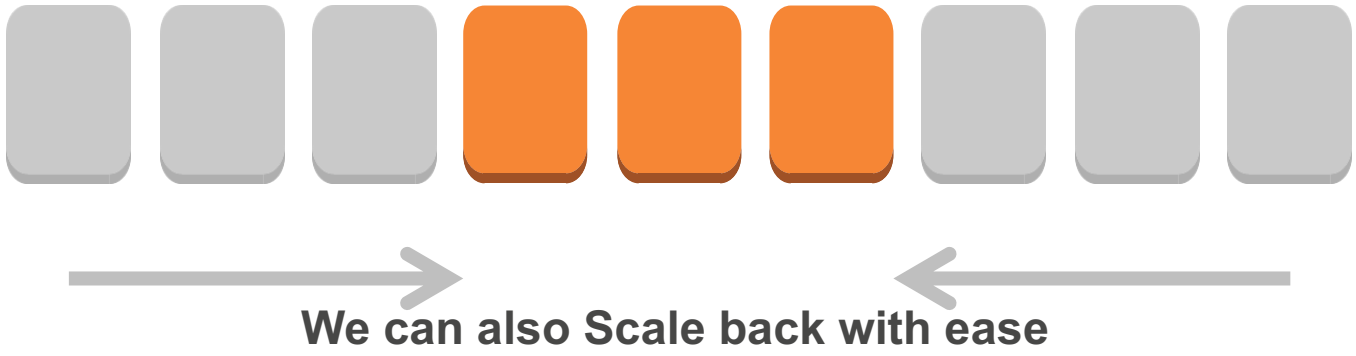


You can now Scale out with ease

amazon web services

Having done that…

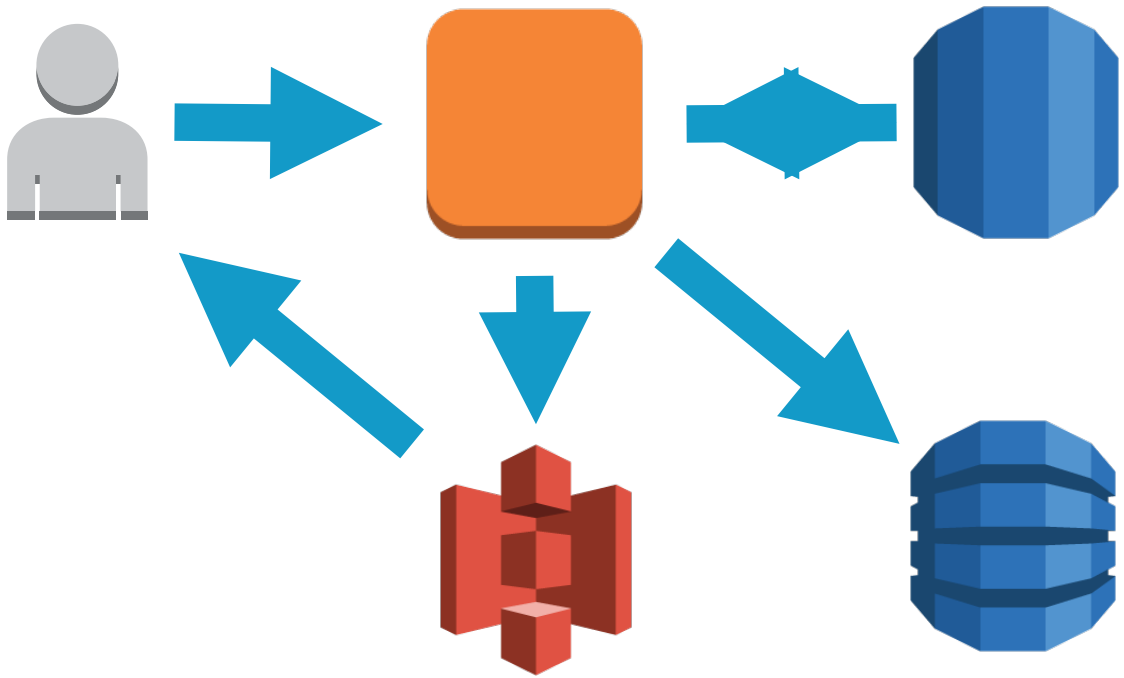
Having decomposed into small, loosely coupled, stateless building blocks



amazon web services

Cloud Architecture Principles

6. Leverage different storage options

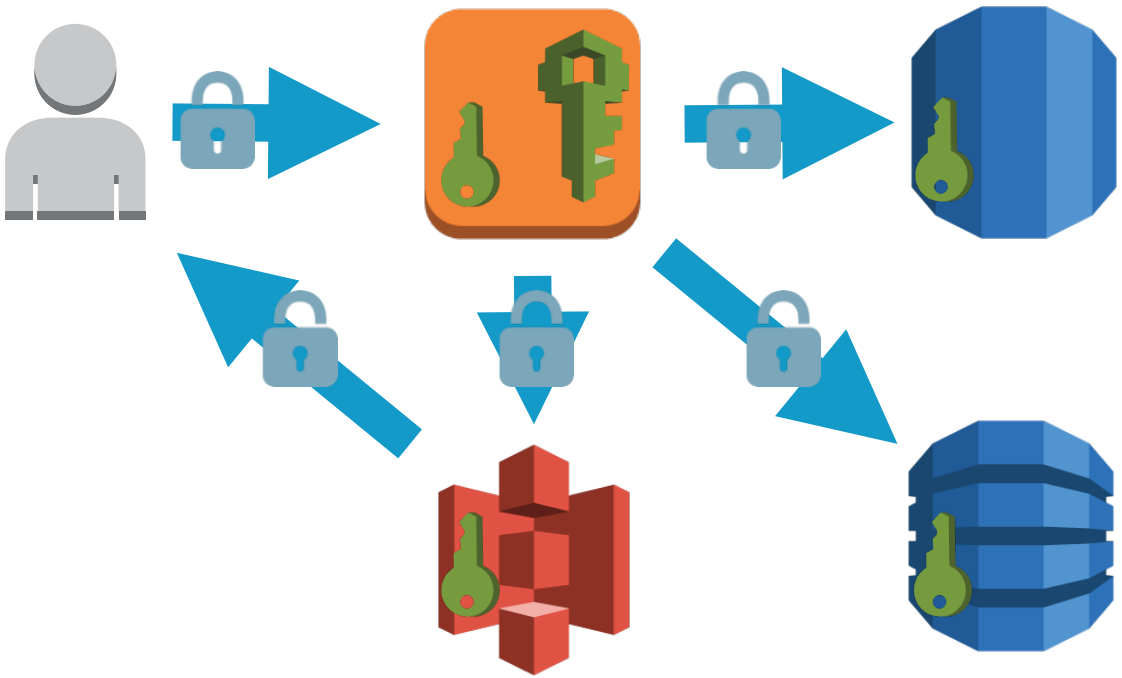


- Don’t log clicks to RDBMS, use NoSQL data store
- Don’t store images in RDBMS, use object store
- Offload log files to scalable object storage

amazon web services

Cloud Architecture Principles

7. Build Security into every layer



- Encrypt data in transit and rest between application tiers
- Enforce principle of least privilege across every service
- Automatically rotate security keys frequently

amazon web services

Cloud Architecture Principles

1. Design for failure and nothing will fail
2. Embrace Elasticity & Automate
3. Loose coupling sets you free
4. Become Stateless
5. Think Parallel
6. Leverage different storage options
7. Build Security into every layer