Matt Tavis Solutions Architect mtavis@amazon.com

Architectural Design Patterns in Cloud Computing

Cloud Best Practices Whitepaper

Prescriptive guidance to Cloud Architects

Just Search for “Cloud Best Practices” to find the link

http://media.amazonwebservices.com/AWS_Cloud_Best_Practices.pdf

Cloud Computing Attributes

What makes the Cloud so attractive

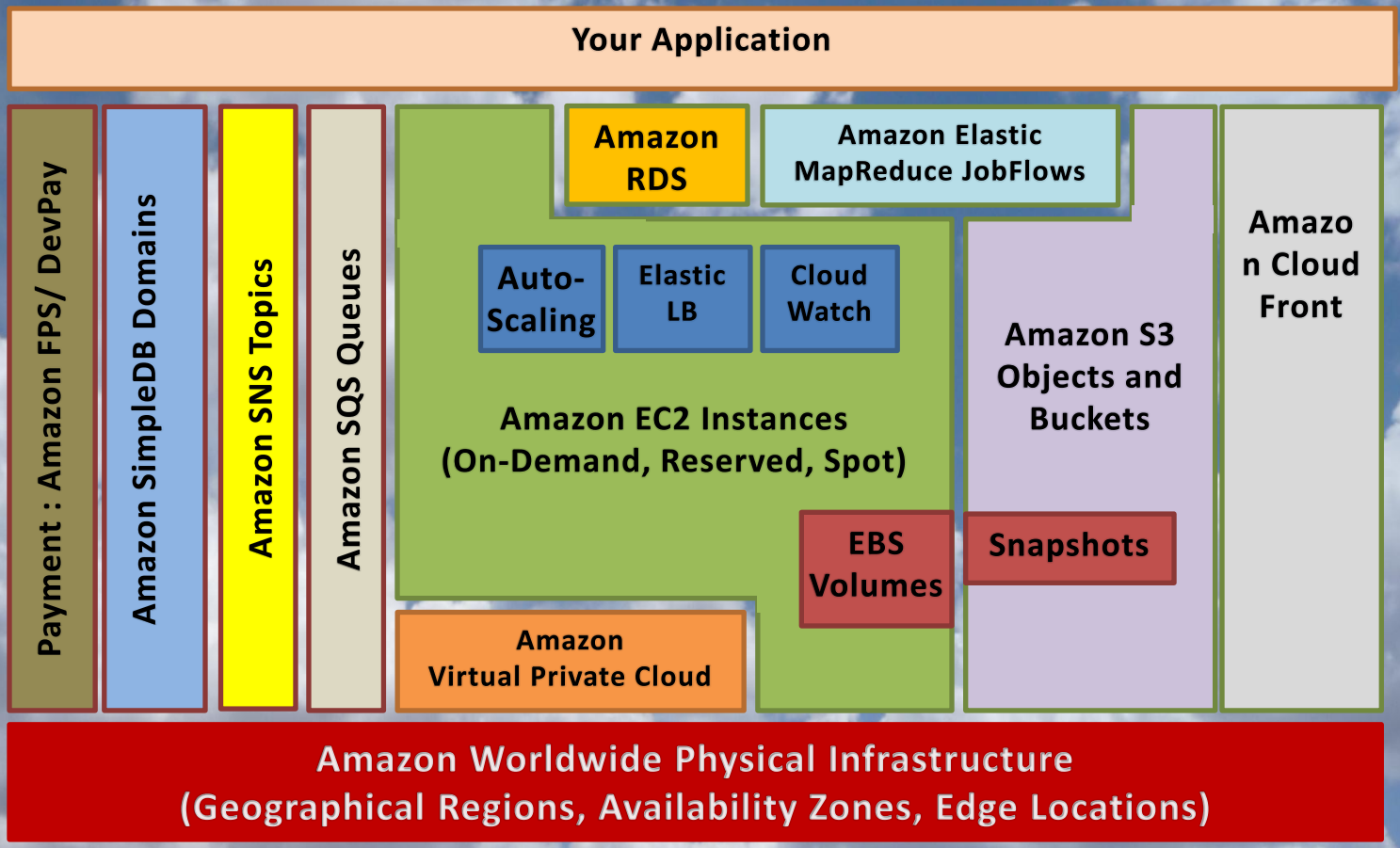
Abstract Resources	Focus on your needs, not on hardware specs. As your needs change, so should your resources.
On-Demand Provisioning	Ask for what you need, exactly when you need it. Get rid of it when you don't need
Scalability in minutes	Scale out or in depending on usage needs.
Pay per consumption	No contracts or long-term commitments. Pay only for what you use.
Efficiency of Experts	Utilize the skills, knowledge and resources of experts.

The "Living and Evolving" Cloud

AWS services and basic terminology

Most Applications Need:

1. Compute
2. Storage
3. Messaging
4. Payment
5. Distribution
6. Scale
7. Analytics



Scalability

Build Scalable Architecture on AWS

A scalable architecture is critical to take advantage of a scalable infrastructure

Characteristics of Truly Scalable Service

- Increasing resources results in a proportional increase in performance
- A scalable service is capable of handling heterogeneity
- A scalable service is operationally efficient
- A scalable service is resilient
- A scalable service becomes more cost effective when it grows

Cloud Architecture Lessons

using Amazon Web Services

1. Design for failure and nothing fails
2. Loose coupling sets you free
3. Implement “Elasticity”
4. Build Security in every layer
5. Don't fear constraints
6. Think Parallel
7. Leverage different storage options

1. Design for Failure

and nothing will really fail

"Everything fails, all the time"
Werner Vogels, CTO Amazon.com

Avoid single points of failure
Assume everything fails, and design backwards
Goal: Applications should continue to function even if the underlying physical hardware fails or is removed or replaced.

2. Loose coupling sets you free

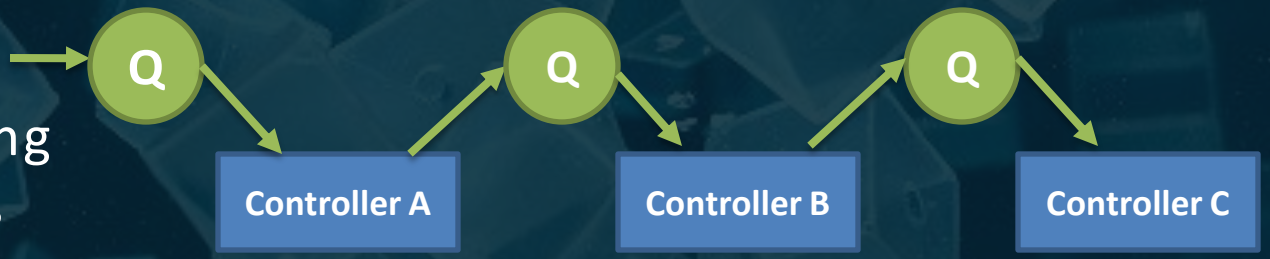
The looser they're coupled, the bigger they scale

Independent components
Design everything as a Black Box
De-coupling for Hybrid models
Load-balance clusters

Use Amazon SQS as Buffers

Tight Coupling

Loose Coupling
using Queues



3. Implement Elasticity

Elasticity is fundamental property of the Cloud

Don’t assume health or fixed location of components
Use designs that are resilient to reboot and re-launch
Bootstrap your instances: Instances on boot will ask a question *“Who am I & what is my role?”*
Enable dynamic configuration

Use Auto-scaling (Free)
Use Elastic Load Balancing on multiple layers
Use configurations in SimpleDB to bootstrap instance

4. Build Security in every layer

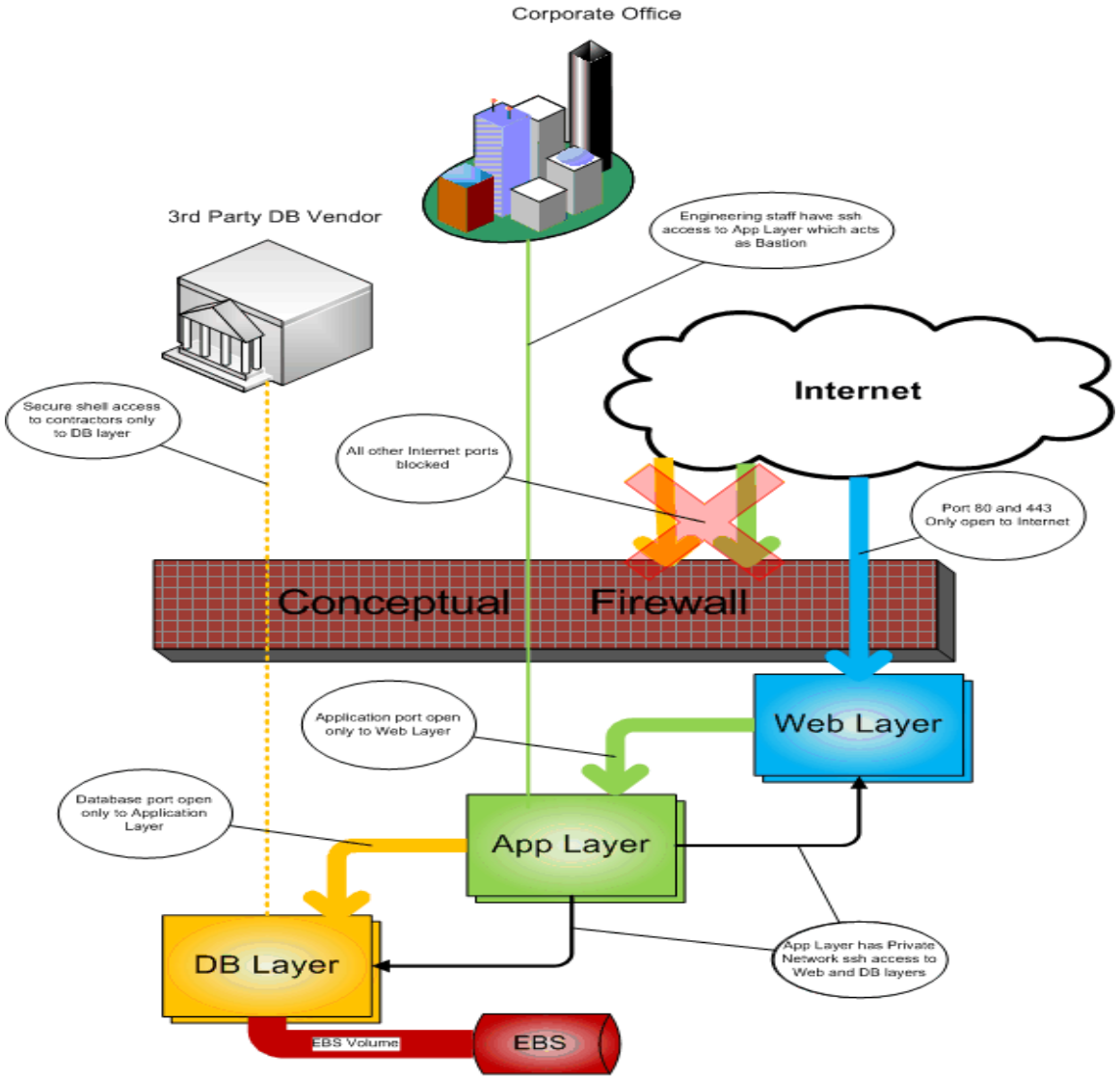
Design with Security in mind

With cloud, you lose a little bit of physical control but not your ownership

Create distinct Security Groups for each Amazon EC2 cluster
Use group-based rules for controlling access between layers
Restrict external access to specific IP ranges
Encrypt data “at-rest” in Amazon S3
Encrypt data “in-transit” (SSL)
Consider encrypted file systems in EC2 for sensitive data
Rotate your AWS Credentials, Pass in as arguments encrypted
Use MultiFactor Authentication

4. Build Security in every layer

Design with Security in mind



"Web" Security Group:

TCP 80 0.0.0.0/0
TCP 443 0.0.0.0/0
TCP 22 "App"

"App" Security Group:

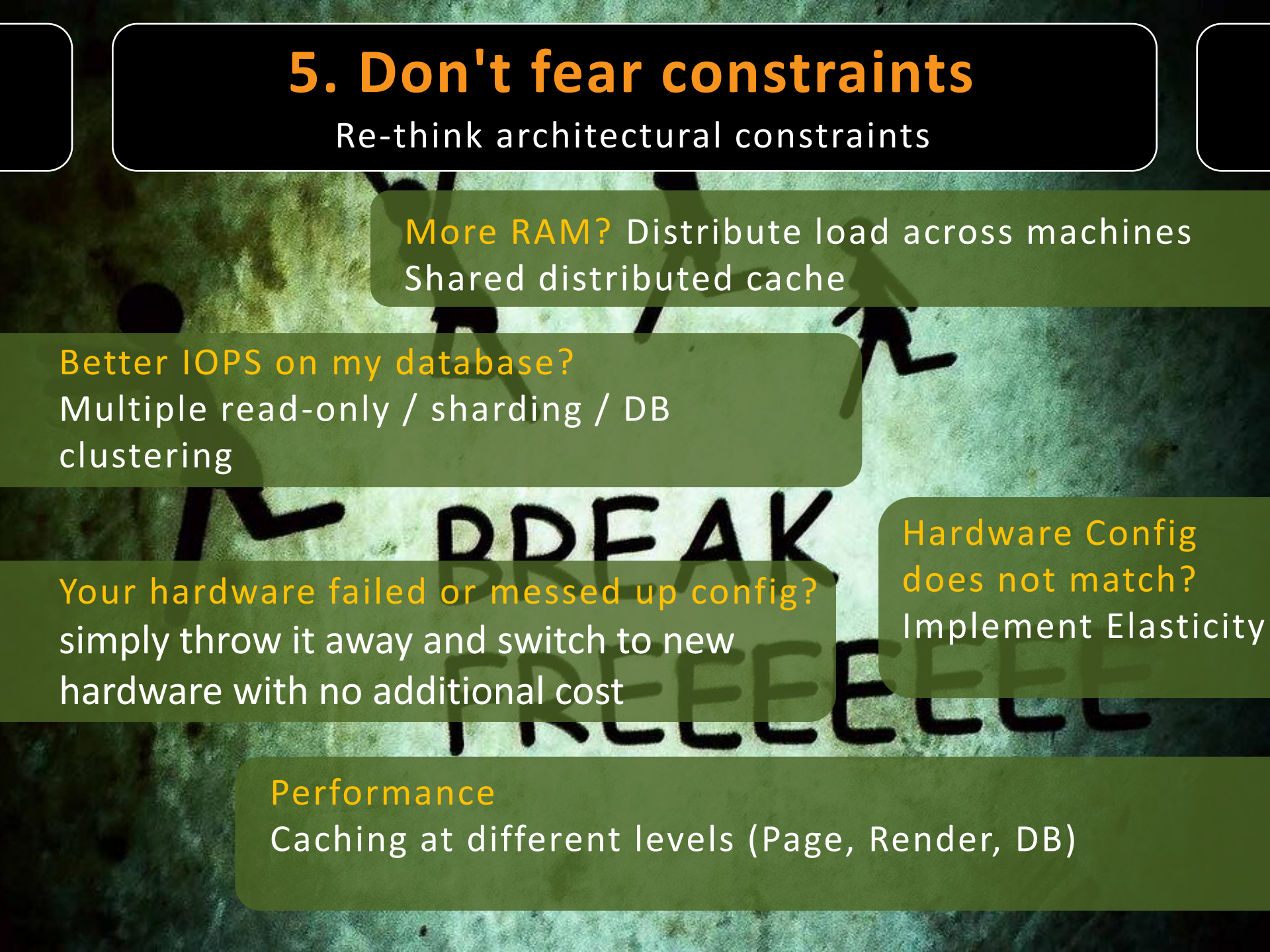
TCP 8080 "Web"
TCP 22 172.154.0.0/16
TCP 22 "App"

"DB" Security Group:

TCP 3306 "App"
TCP 3306 163.128.25.32/32
TCP 22 "App"

5. Don't fear constraints

Re-think architectural constraints



More RAM? Distribute load across machines
Shared distributed cache

Better IOPS on my database?
Multiple read-only / sharding / DB clustering

Hardware Config does not match?
Implement Elasticity

Your hardware failed or messed up config?
simply throw it away and switch to new hardware with no additional cost

Performance
Caching at different levels (Page, Render, DB)

6. Think Parallel

Serial and Sequential is now history

Experiment different architectures in parallel

Multi-treading and Concurrent requests to cloud services

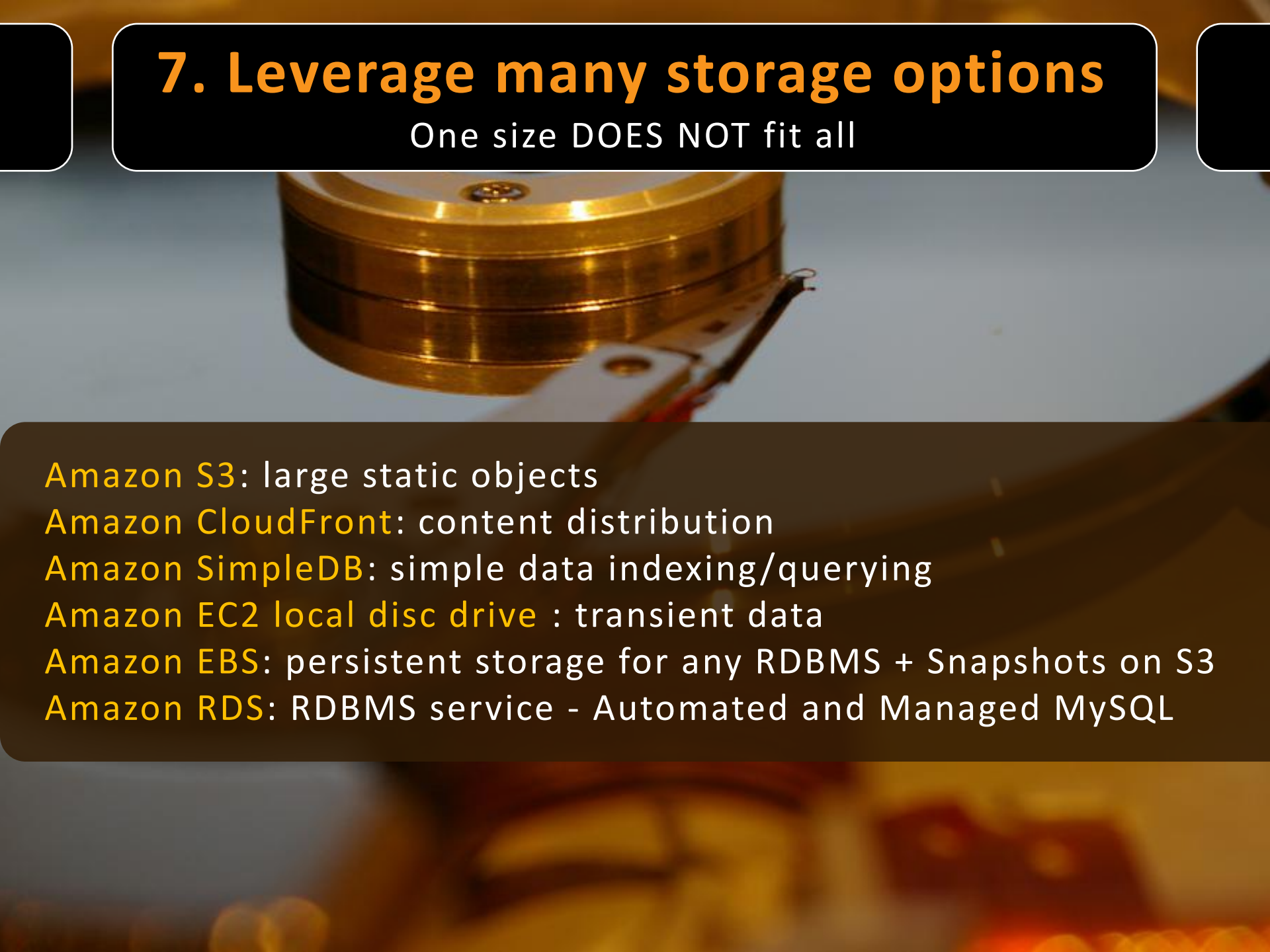
Run parallel MapReduce Jobs

Use Elastic Load Balancing to distribute load across multiple servers

Decompose a Job into its simplest form

7. Leverage many storage options

One size DOES NOT fit all



Amazon S3: large static objects
Amazon CloudFront: content distribution
Amazon SimpleDB: simple data indexing/querying
Amazon EC2 local disc drive : transient data
Amazon EBS: persistent storage for any RDBMS + Snapshots on S3
Amazon RDS: RDBMS service - Automated and Managed MySQL

7. Leverage many storage options

Which storage option to use when?

	Amazon S3 + CF	Amazon EC2 Ephemeral Store	Amazon EBS	Amazon SimpleDB	Amazon RDS
Ideal for	Storing Large write-once, read-many types of objects, Static Content Distribution	Storing non-persistent transient updates	Off-instance persistent storage for any kind of data,	Querying light-weight attribute data	Storing and querying structured Relational and referential Data
Ideal examples	Media files, audio, video, images, Backups, archives, versioning	Config Data, scratch files, TempDB	Clusters, boot data, Log or data of commercial RDBMS like Oracle, DB2	Querying, Mapping, tagging, click-stream logs, metadata, shared-state management, indexing	Complex transactional systems, inventory management and order fulfillment systems
Not recommended for	Querying, Searching	Storing Database logs or backups, customer data		Relational (joins) query	
Not recommended examples	Database, File Systems	Sensitive data	Content Distribution	OLTP, DW cube rollups	Simple lookups

Applying Cloud Architecture Lessons

Moving a Web Architecture to the Cloud

Exterior Firewall Hardware or Software Solution to open standard Ports (80, 443)

Web Load Balancer
Hardware or Software solution to distribute traffic over web servers

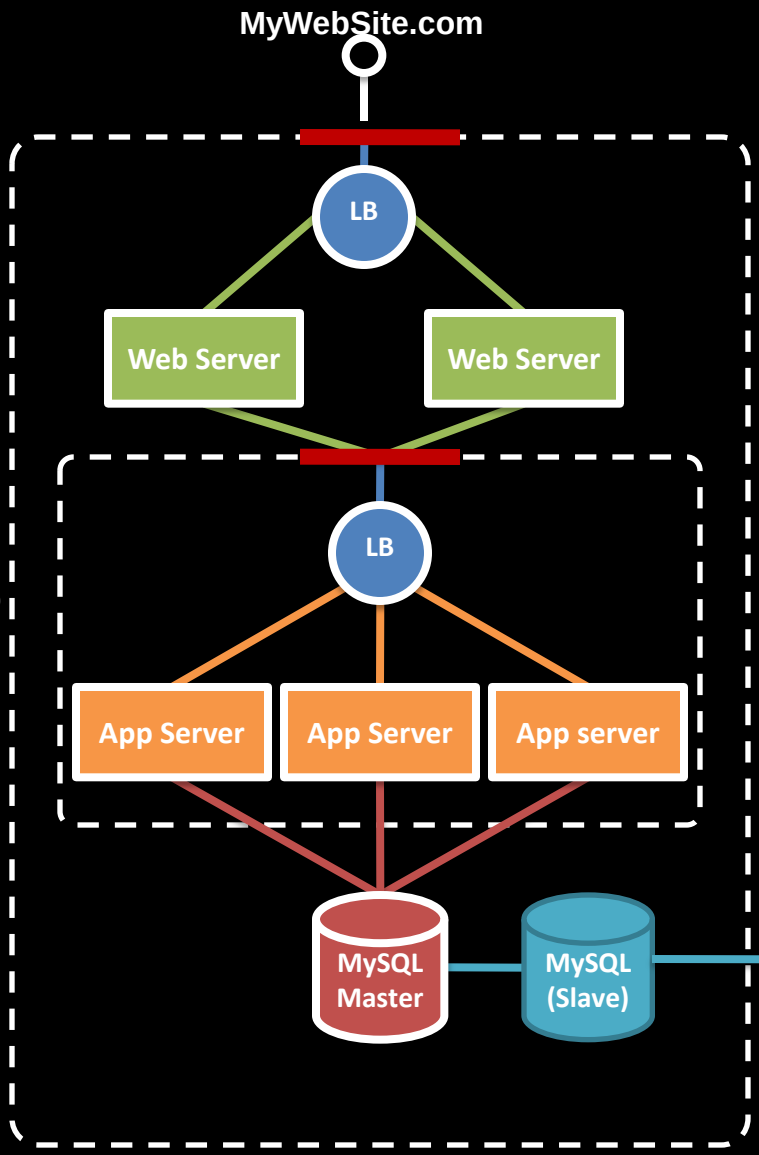
Web Tier
Fleet of machines handling HTTP requests.

Backend Firewall Limits access to application tier from web tier

App Load Balancer
Hardware or Software solution to spread traffic over app servers

App Server Tier
Fleet of machines handling Application specific workloads Caching server machines can be implemented at this layer

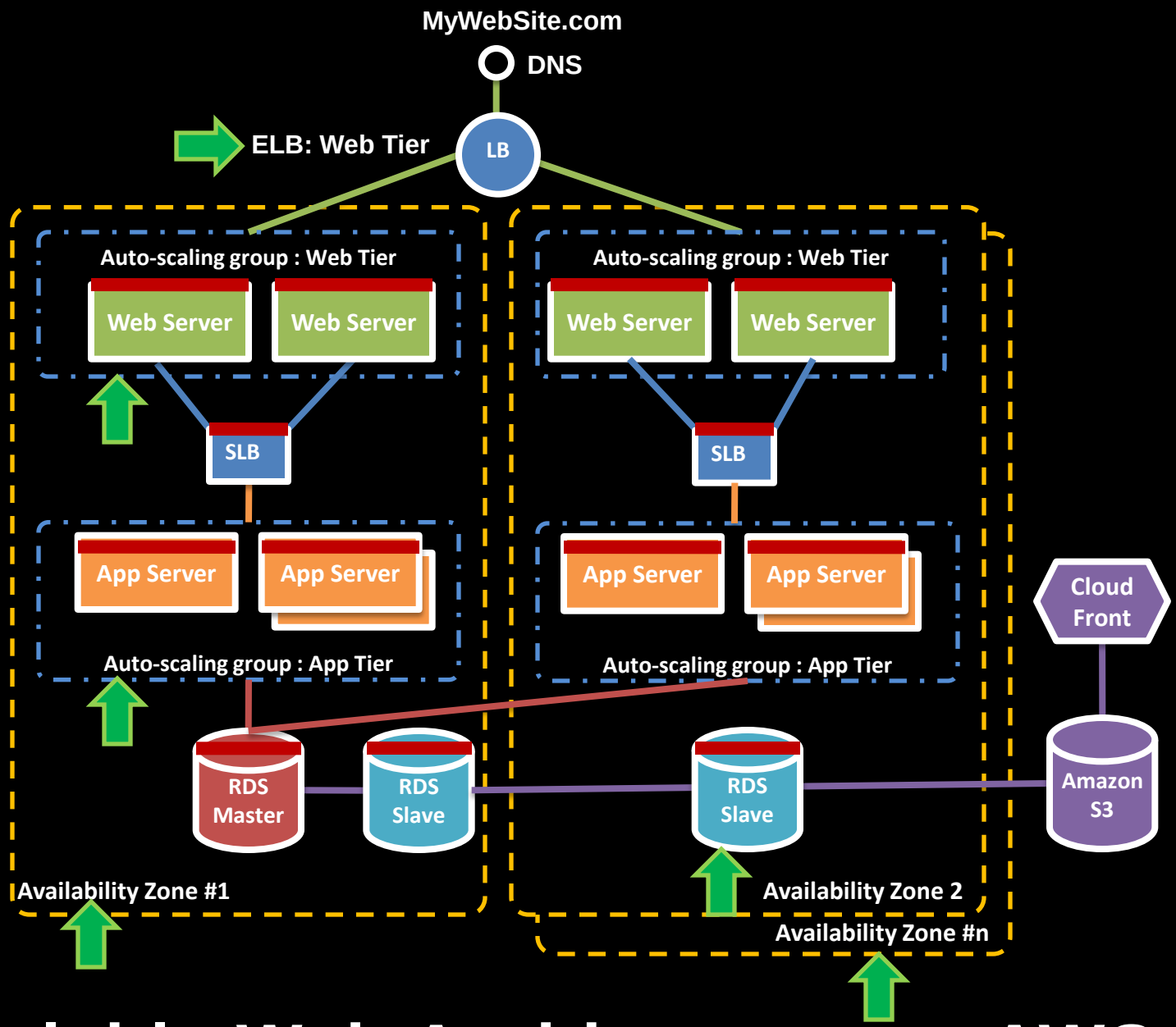
Data Tier
Database Server machines with master and local running separately, Network storage for Static objects



Backups on Tapes Periodic backups stored on Tapes usually managed by 3rd party at their site

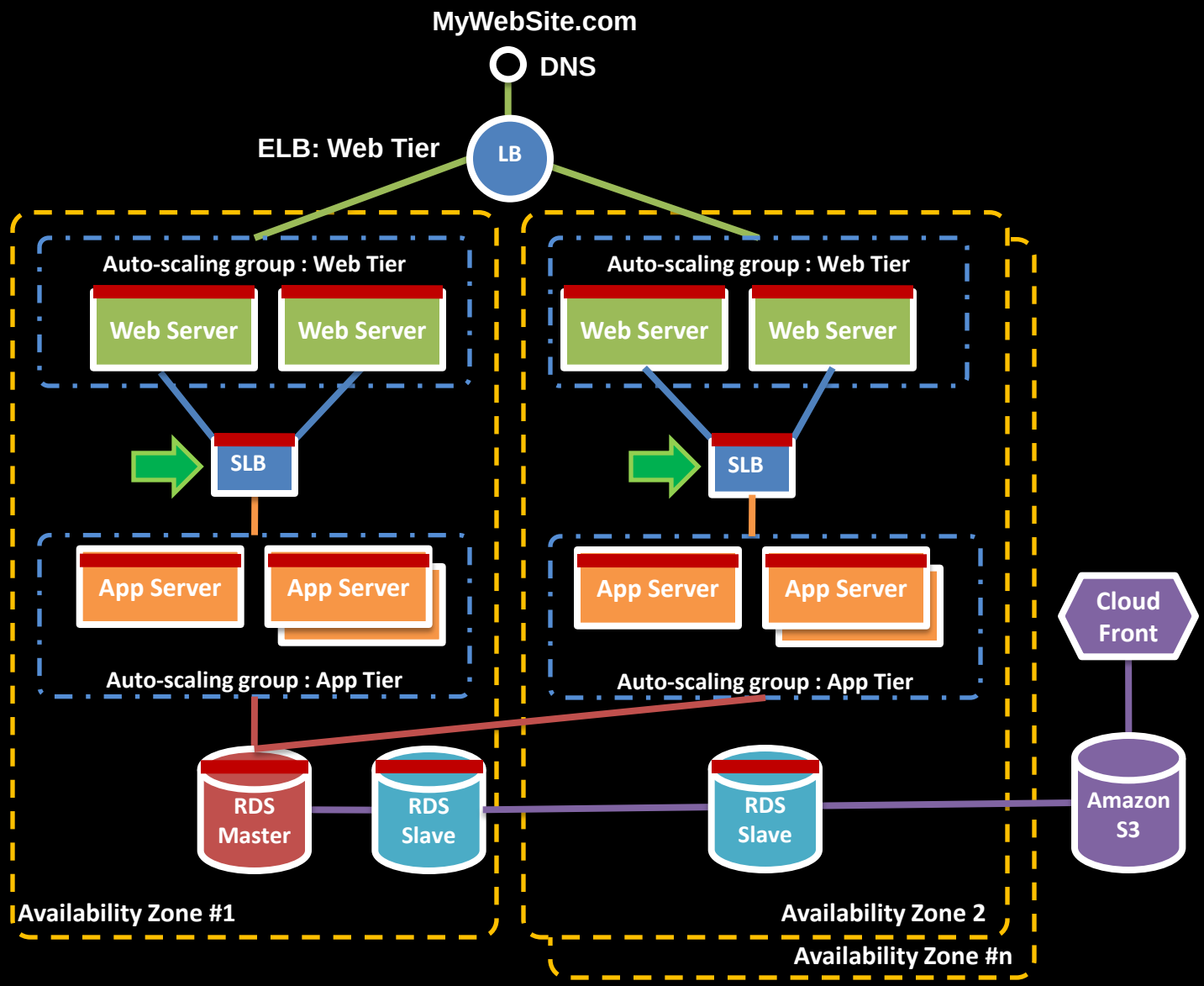
A Classic Web Architecture

Design for failure and nothing fails



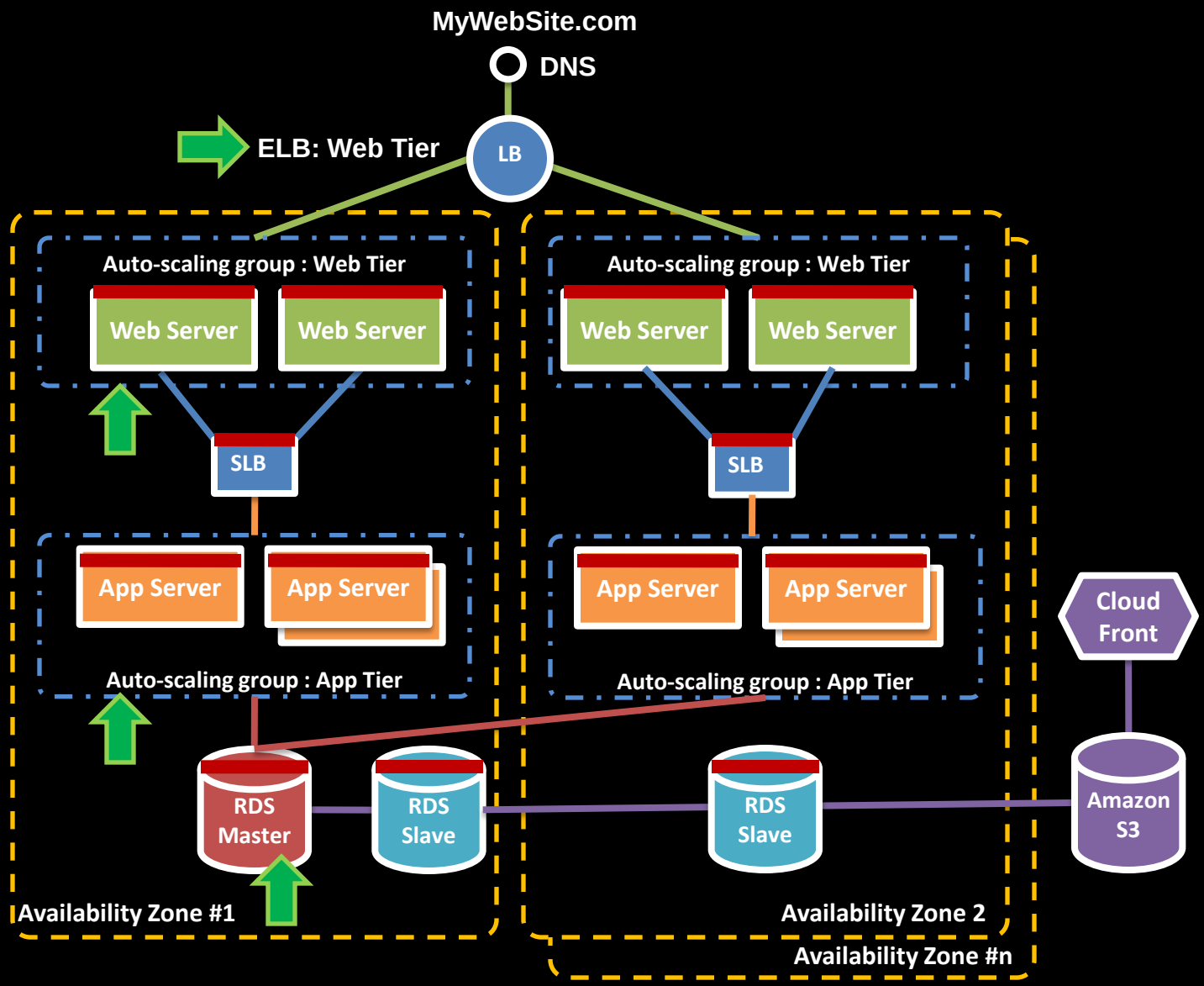
A Scalable Web Architecture on AWS

Loose coupling sets you free

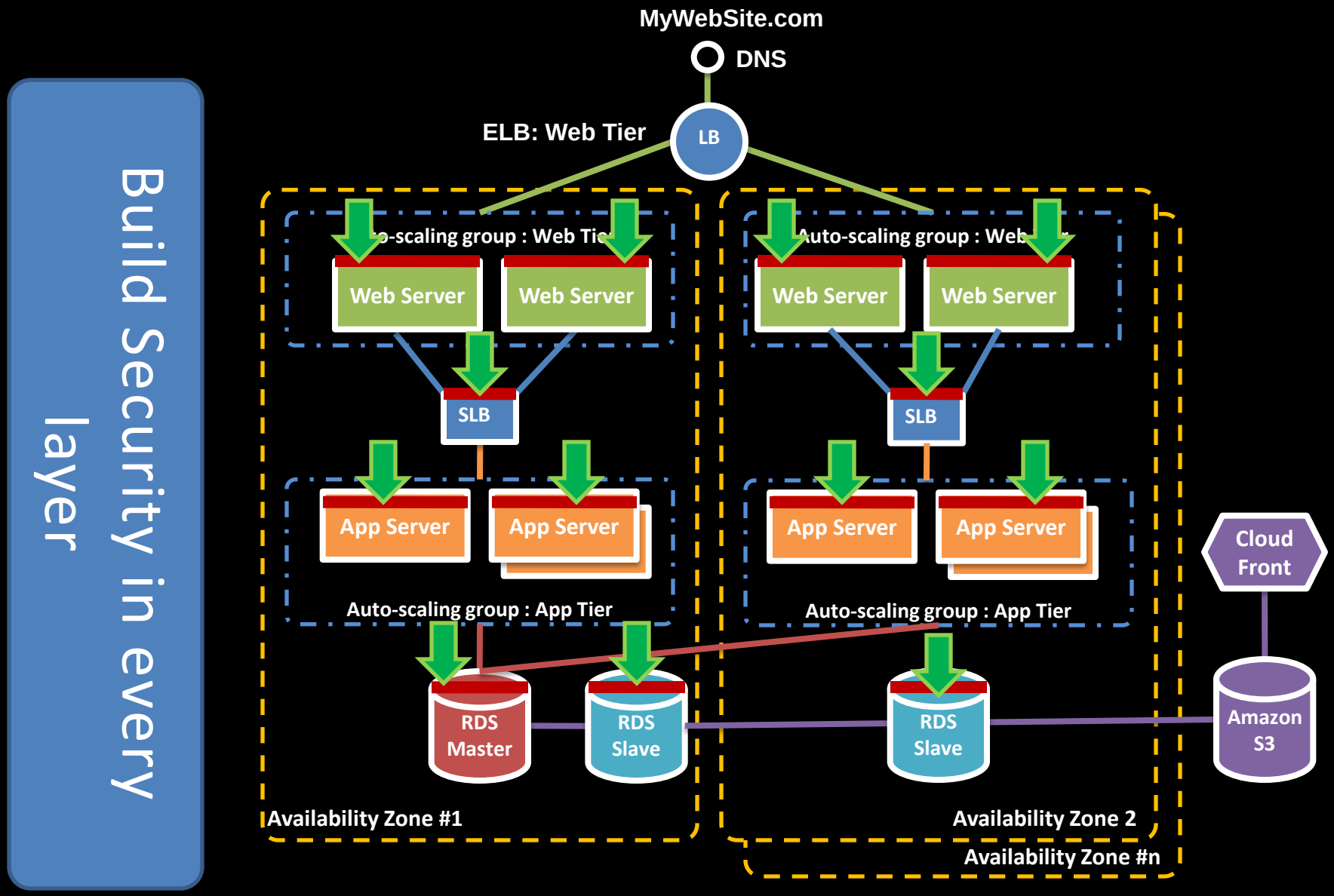


A Scalable Web Architecture on AWS

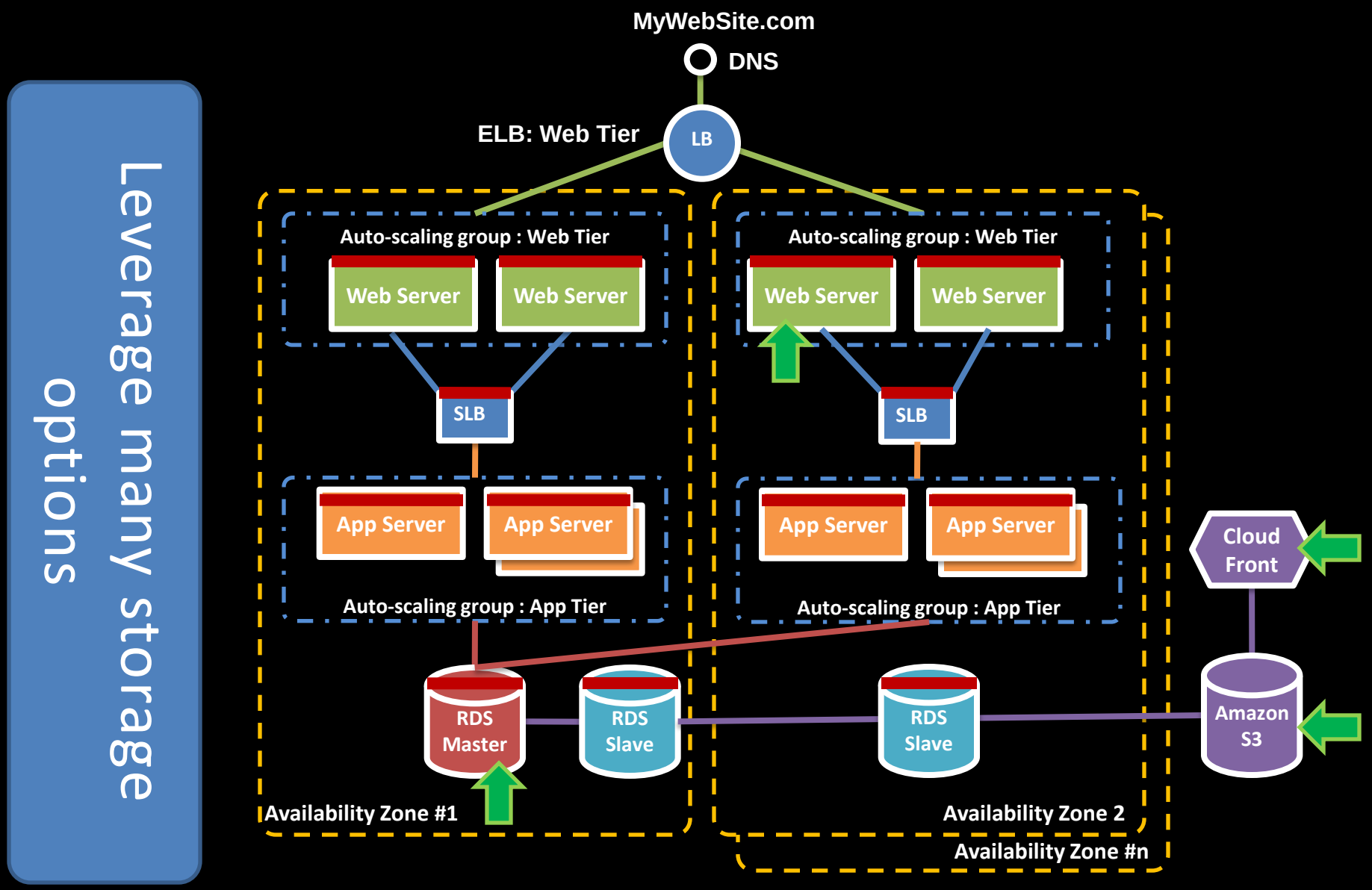
Implement elasticity



A Scalable Web Architecture on AWS



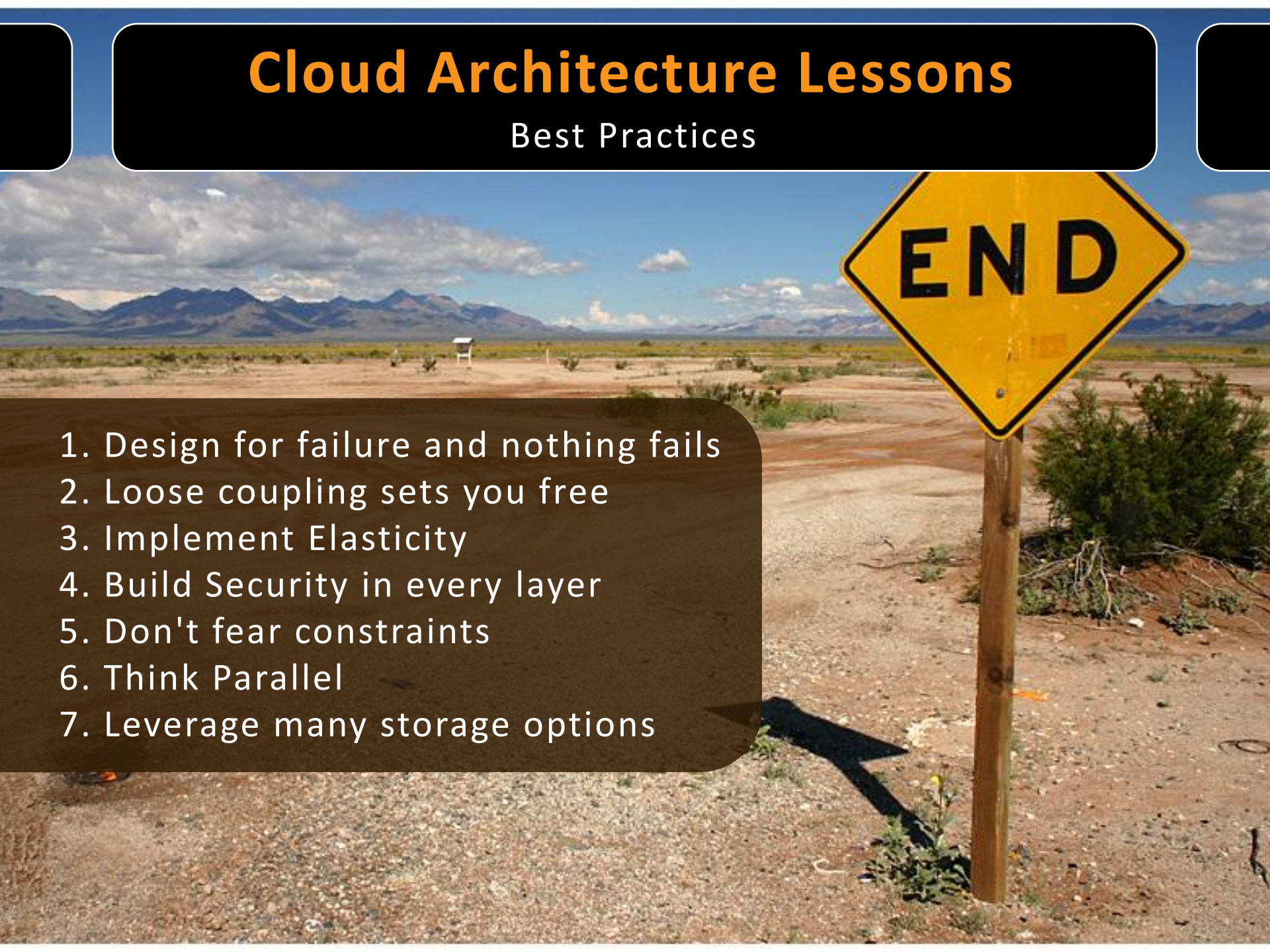
A Scalable Web Architecture on AWS



A Scalable Web Architecture on AWS

Cloud Architecture Lessons

Best Practices



1. Design for failure and nothing fails
2. Loose coupling sets you free
3. Implement Elasticity
4. Build Security in every layer
5. Don't fear constraints
6. Think Parallel
7. Leverage many storage options

MANAGEMENT AND OPERATIONS ON AWS

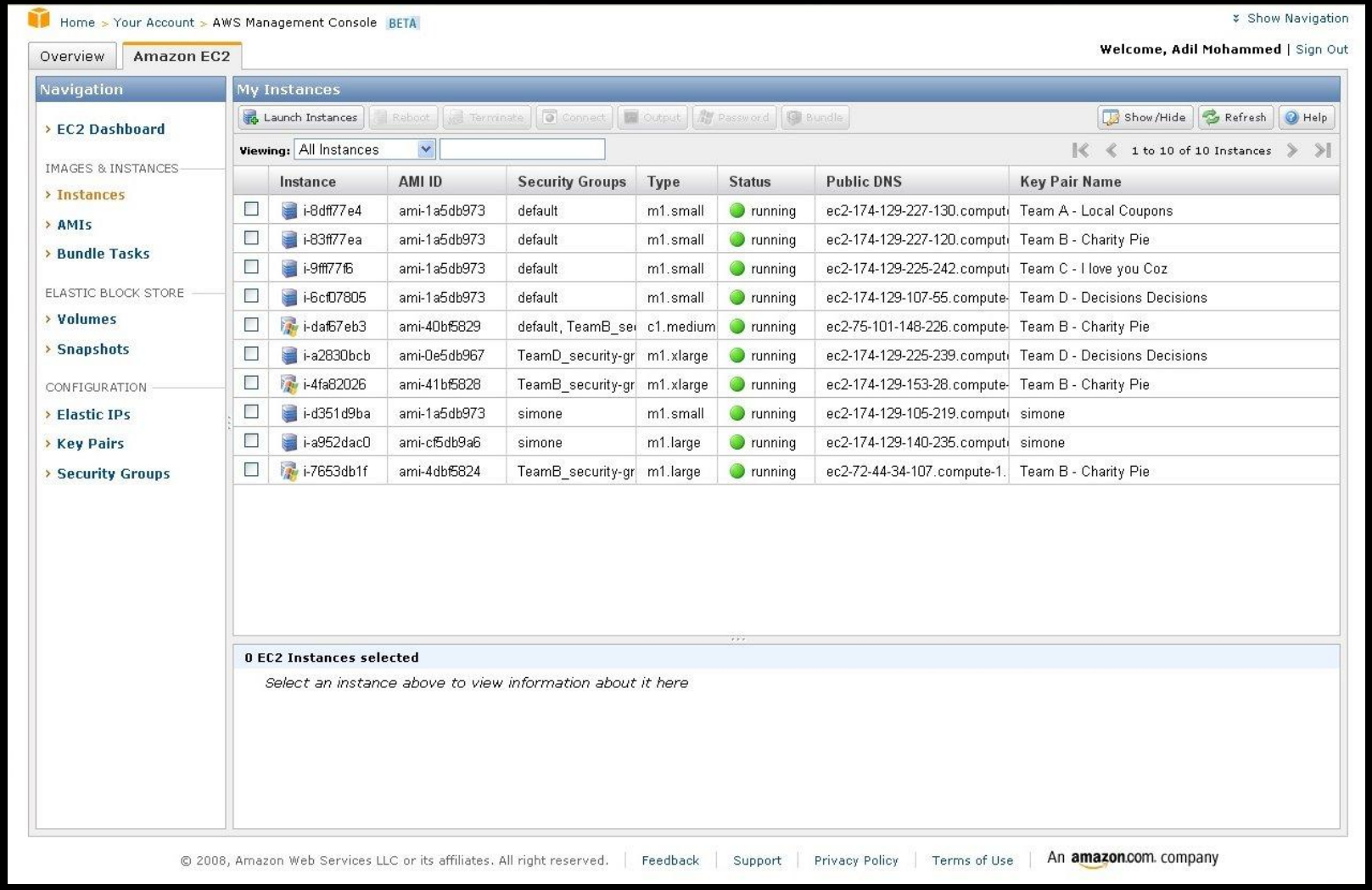
Management and Operations

- AWS affords numerous management options
- Utilize existing IT management systems
  - Amazon VPC enables existing management and operations systems, security policies, etc. to extend to cloud resources
  - AWS partners with numerous management platform providers
- Utilize cloud-oriented third-party providers
  - RightScale, Elastra and many others
- Leverage AWS APIs to build custom solutions
  - API-based control enables existing workflow applications to manage AWS resources

AWS Features for Management and Operations

- AWS provides management and operations primitives to enable plugging AWS in anywhere
  - CloudWatch provides real-time API-based monitoring data
  - AutoScaling provides API-based touch-less scaling out and in for EC2 resources
  - ElasticLoadBalancing provides APIs for touch-less balancing of load across a dynamic fleet of EC2 instances across AZs
- AWS Management Console offers GUI for management and visibility
- All AWS resources can be allocated, managed and de-allocated via a flexible API set

AWS Management Console



But can I use Package X in the AWS Cloud?


zend
IBM
Microsoft
SYBASE
sonian
STATÊRA
TIBCO
The Power of Now
RIGHT SCALE
SUGARCRM
THE CLOUD IS OPEN
acquia
Alfresco
Appian
salesforce.com
Success. Not Software.
appzero
aiCACHE
GIGASPACES
BOOMI
CITRIX
Compiere
ORACLE
model metrics
SIGNIANT
Quantivo
enStratus
ca
VERTICA
ELASTRA
PERVASIVE
New Relic
CAST IRON SYSTEMS
Pega
freedom oss
zmanda
phurnace SOFTWARE
rPath
Engine Yard
InstallFree
Designed for Users. Built for IT.
smartscale
THE POWER OF SCALE
INFORMATICA
VMLogix
redhat
kaavo
moonwalk
CloudTest by SOASTA
accenture


amazon
web services™
http://aws.amazon.com