AWS

SUMMIT

Hybrid Architecture Design and Well-Architected Framework

Jhen-Wei Huang (黃振維), Solutions Architect

June 7, 2017

amazon web services

What to expect from the session

What is Hybrid Architecture Design?

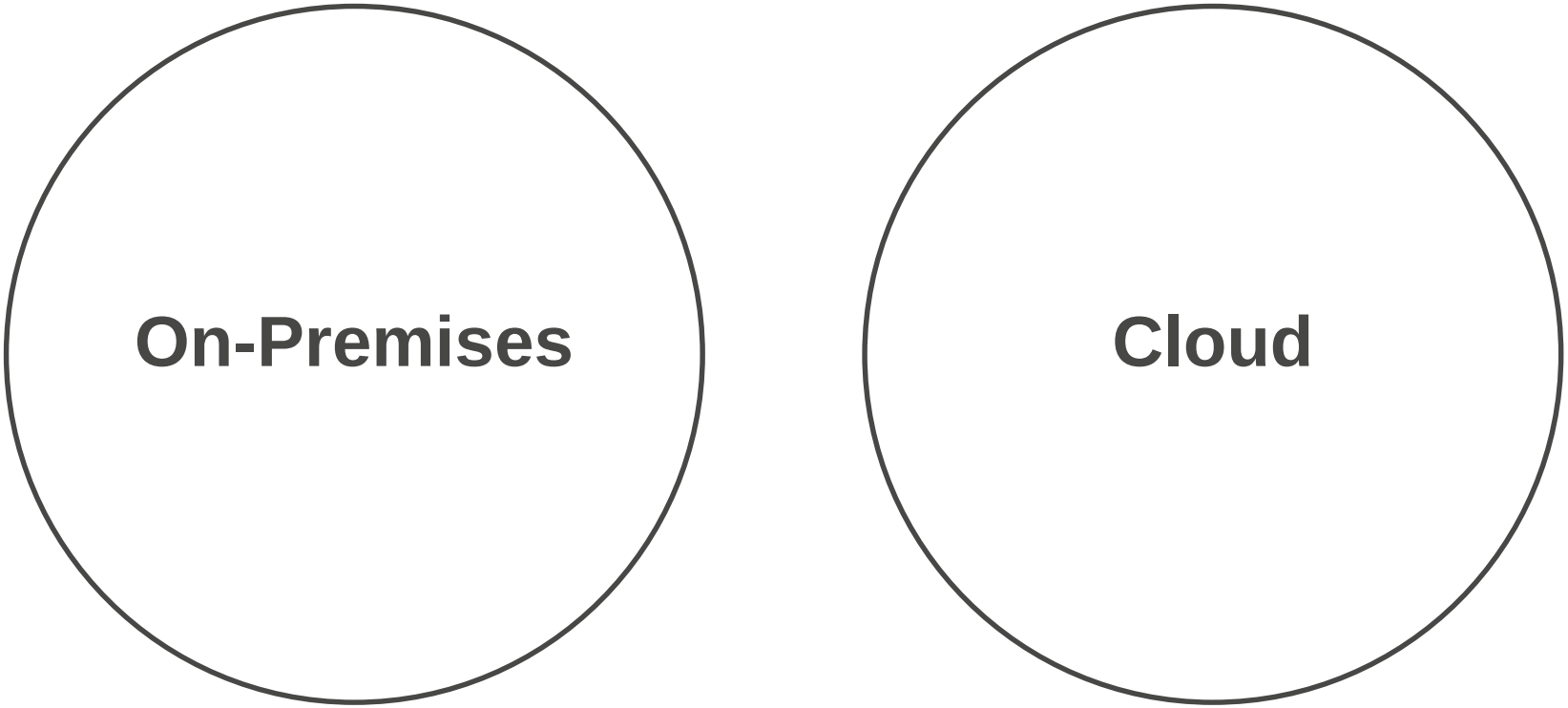
What is the AWS well-architected framework?

What are core tenets to being well architected?

Customer use case

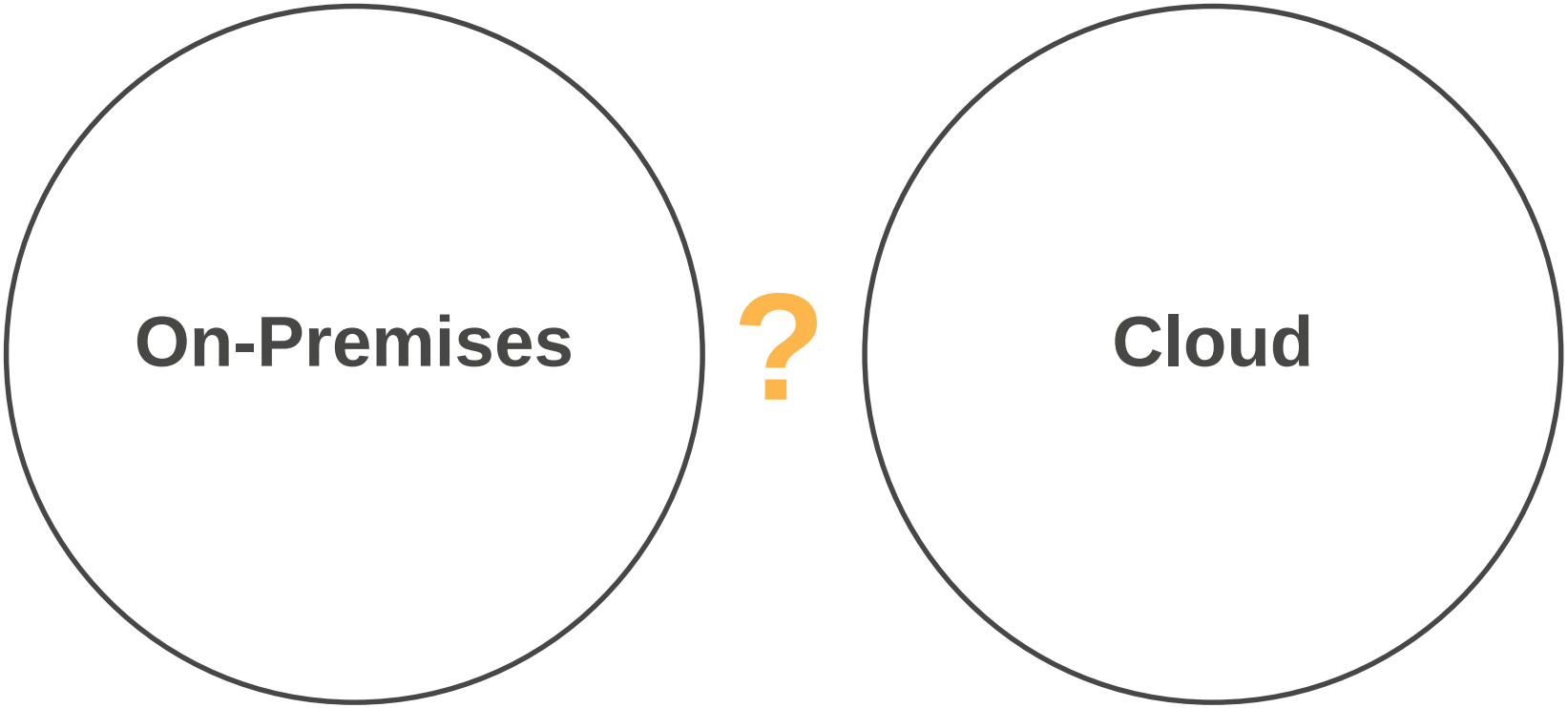
Should I migrate everything to AWS?

No, this is more than a binary choice.



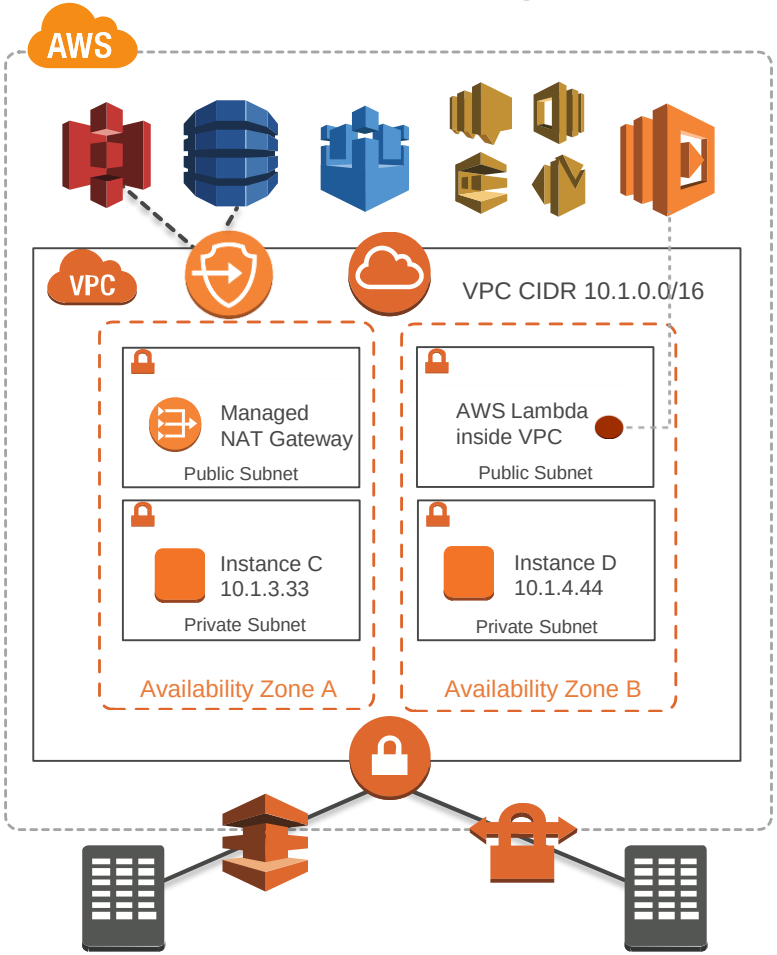
Should I migrate everything to AWS?

We *just* need to figure out the connectivity…



Hybrid networking

AWS networking



Lets get distracted by new things:

Virtual Private Endpoints for S3

Gives you the ability to connect privately to S3

AWS Lambda inside a VPC

Access Lambda without having to go through a VGW

NAT Gateway

Use NAT gateway within a VPC for manage NAT to the Internet

Connecting to AWS

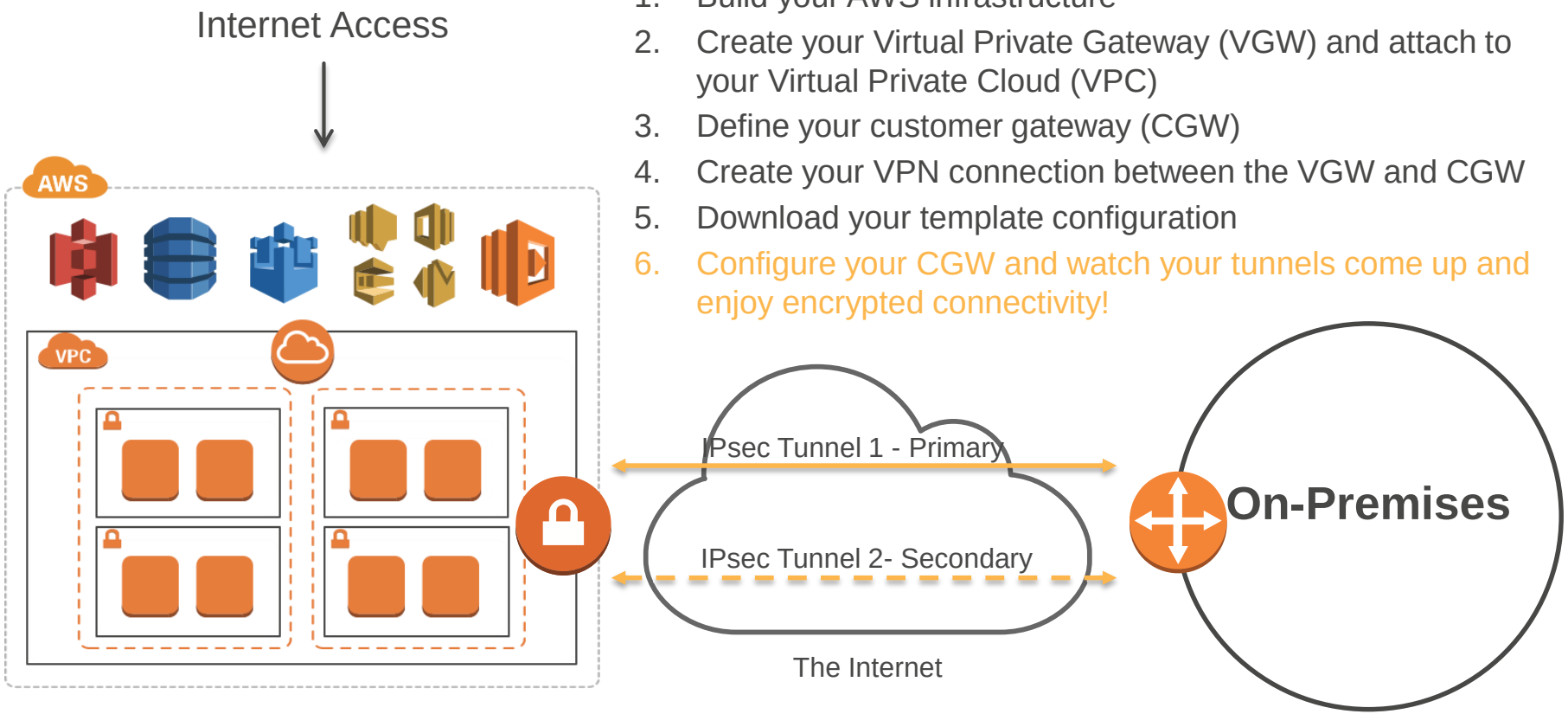
IGWs, VGWs, VPNs, and AWS Direct Connect

amazon web services

VPN connectivity

Provisioning VPN connections

1. Build your AWS infrastructure
2. Create your Virtual Private Gateway (VGW) and attach to your Virtual Private Cloud (VPC)
3. Define your customer gateway (CGW)
4. Create your VPN connection between the VGW and CGW
5. Download your template configuration
6. Configure your CGW and watch your tunnels come up and enjoy encrypted connectivity!

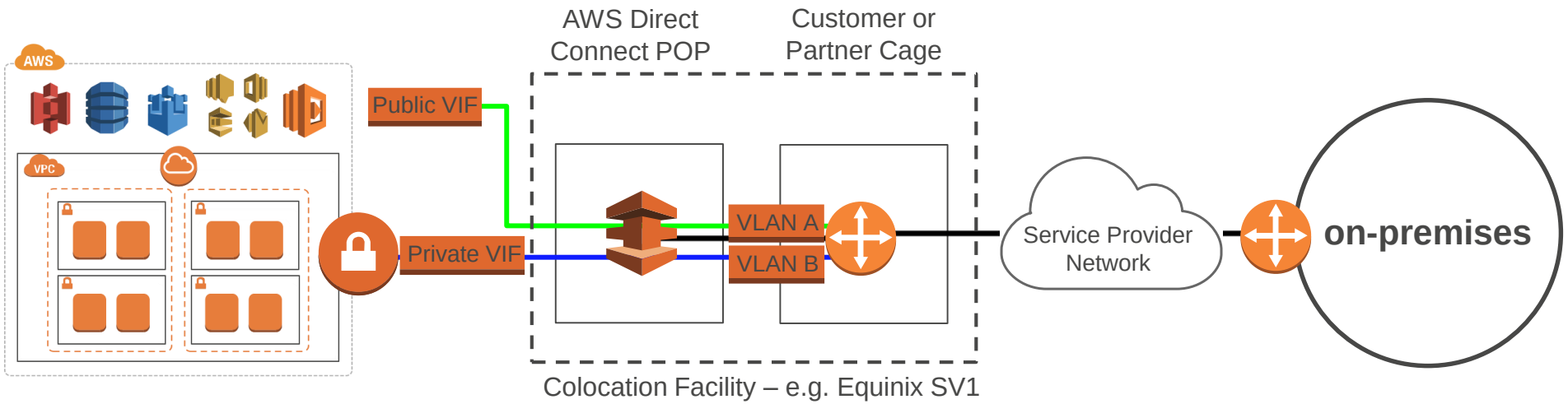


AWS Direct Connect – Provisioning

1. Build your AWS infrastructure
2. Create your Virtual Private Gateway (VGW) and attach to your Virtual Private Cloud (VPC)
3. Order an AWS Direct Connect from the console or through a Direct Connect Partner

at&t Level(3) verizon + More

4. Have your cross connect provisioned from the AWS router to your device or your partners device (or use a partners NNI)
5. Build connectivity if not already available through partner back to on-premises
6. Provision your Virtual interfaces (private or public) and start using your AWS Direct Connect.



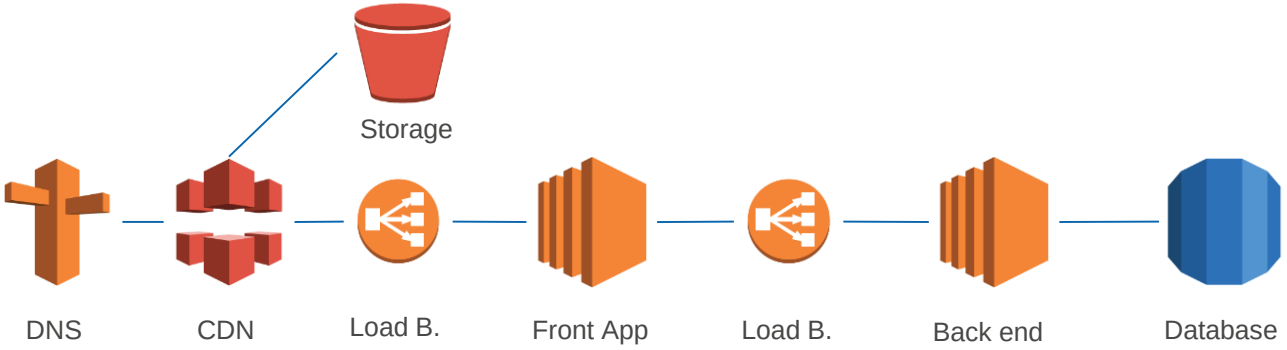
Common hybrid use cases

What kind of hybrid architectures can we build?

amazon web services

Customer-facing applications

External apps on AWS



Scalability and Elasticity

Auto Scaling infrastructure to required capacity and **match spending to actual utilization**

High Availability

Application deployments that span across **multiple facilities** with adequate load balancing

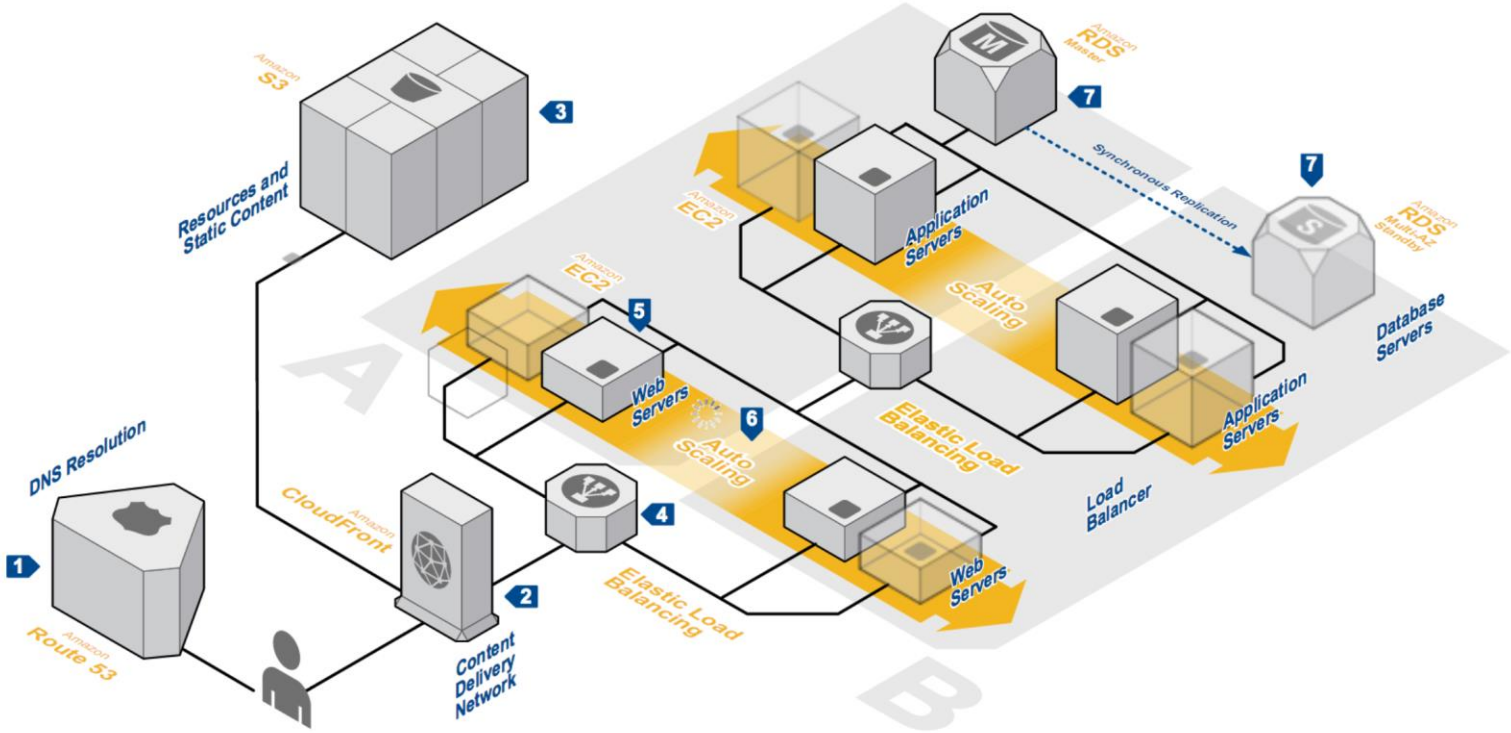
Global Reach

Highly available **global services** on edge locations across the world

Maintainability

Fully managed service portfolio for most common application components

The famous three-tiered web application

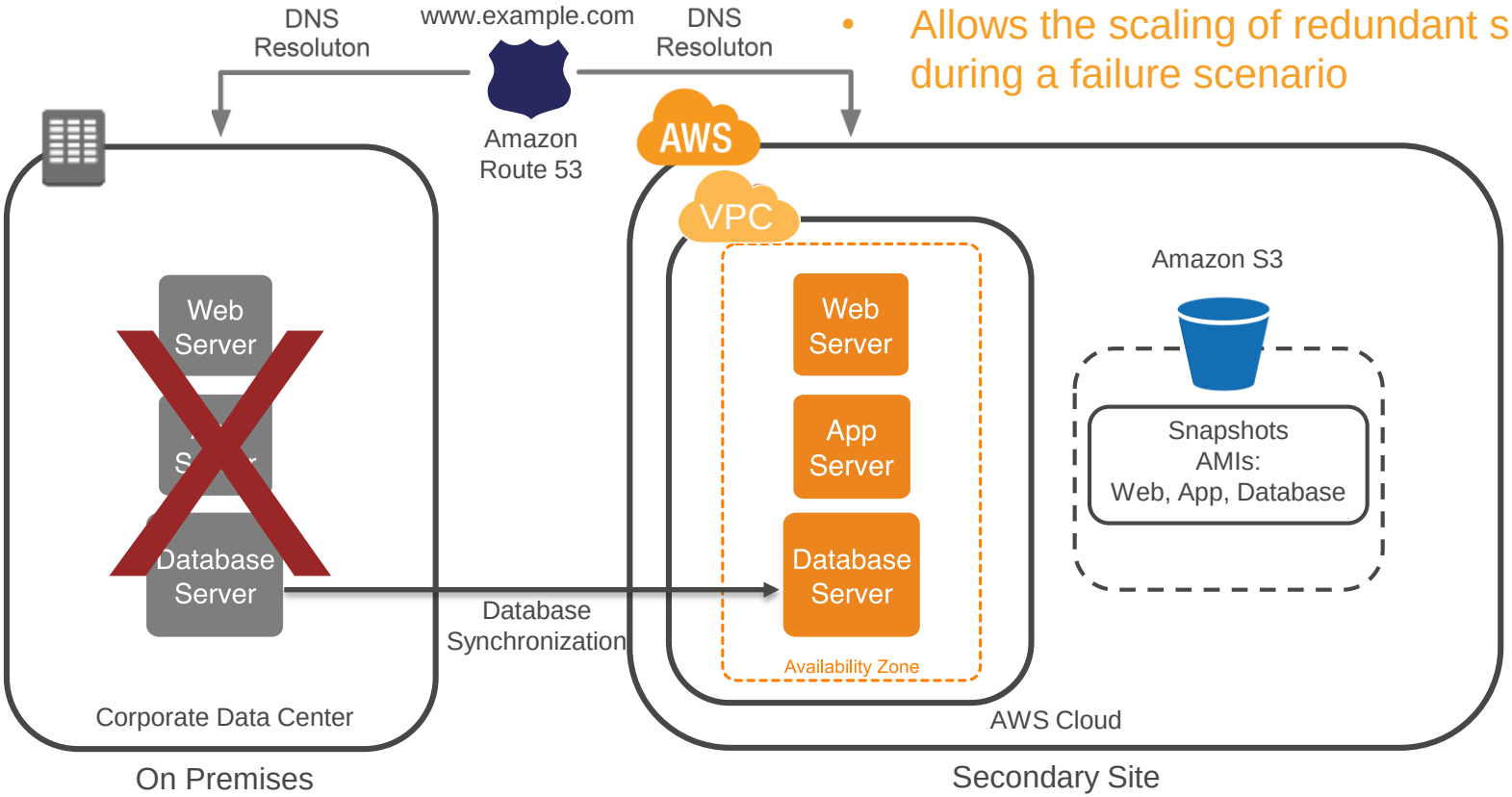


Reference: https://aws.amazon.com/architecture/

Building multi-site deployments with AWS

Pilot light architecture

- Allows the scaling of redundant sites during a failure scenario



Defining communications

The communications matrix

Allows for the description of interconnectivity between applications.

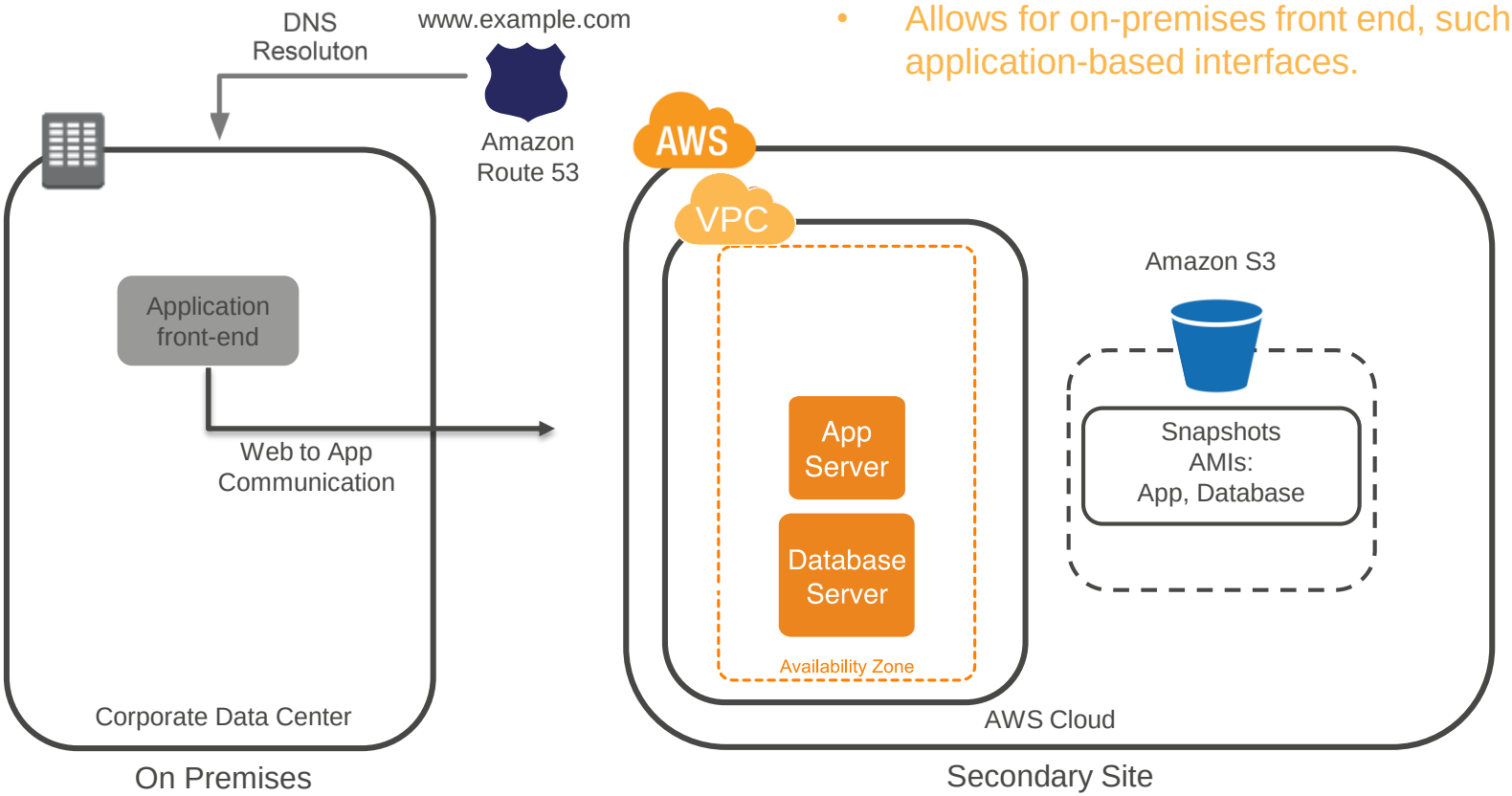
By defining communications you can determine where applications may be placed based on the network properties of any points of interconnection.

#	Source Application	Destination Application	Port	Bandwidth	Latency
#1	Web Tier	Application Tier	443	10Mbps	10ms
#2	Application Tier	Database Tier 1	1433	50Mbps	2ms
#3	Database Tier 1	Database Tier 2	1521	50Mbps	50ms

Placing your application where it makes sense

On-premises based front end

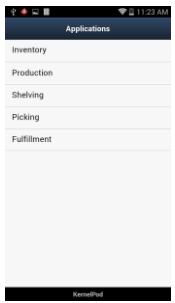
- Allows for on-premises front end, such as application-based interfaces.

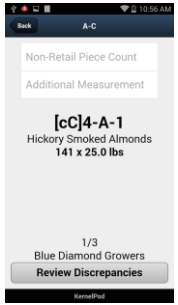


Customer case study: Nuts.com

Nuts.com required the front end for their web application to reside inside their distribution centers in the form of an application running on portable Motorola Simbol TC70 hardened barcode scanners.

With users constantly communicating with the AWS-built application continuously, low latency seamless connectivity was a hard requirement of the project.

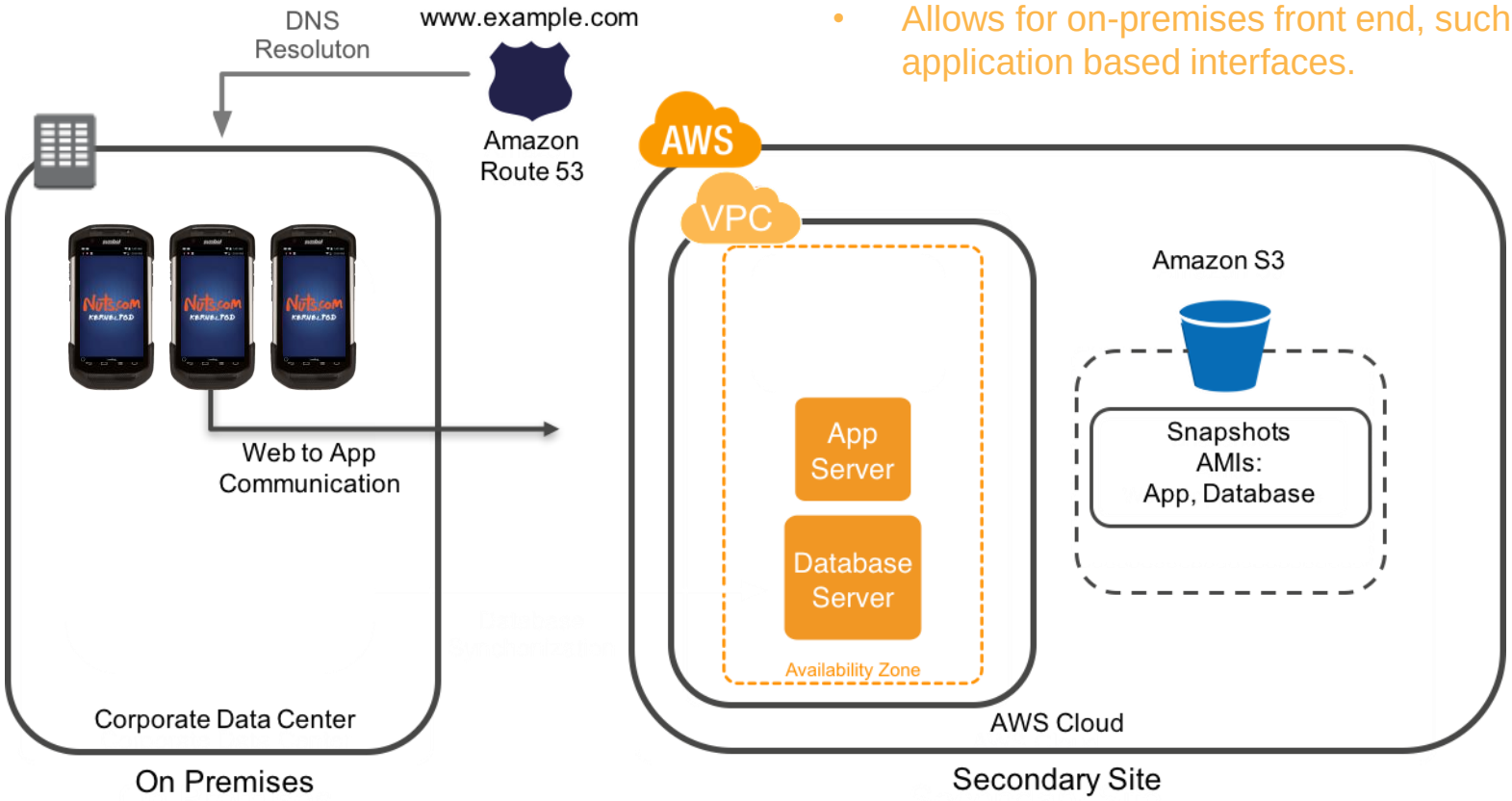




Customer case study: Nuts.com

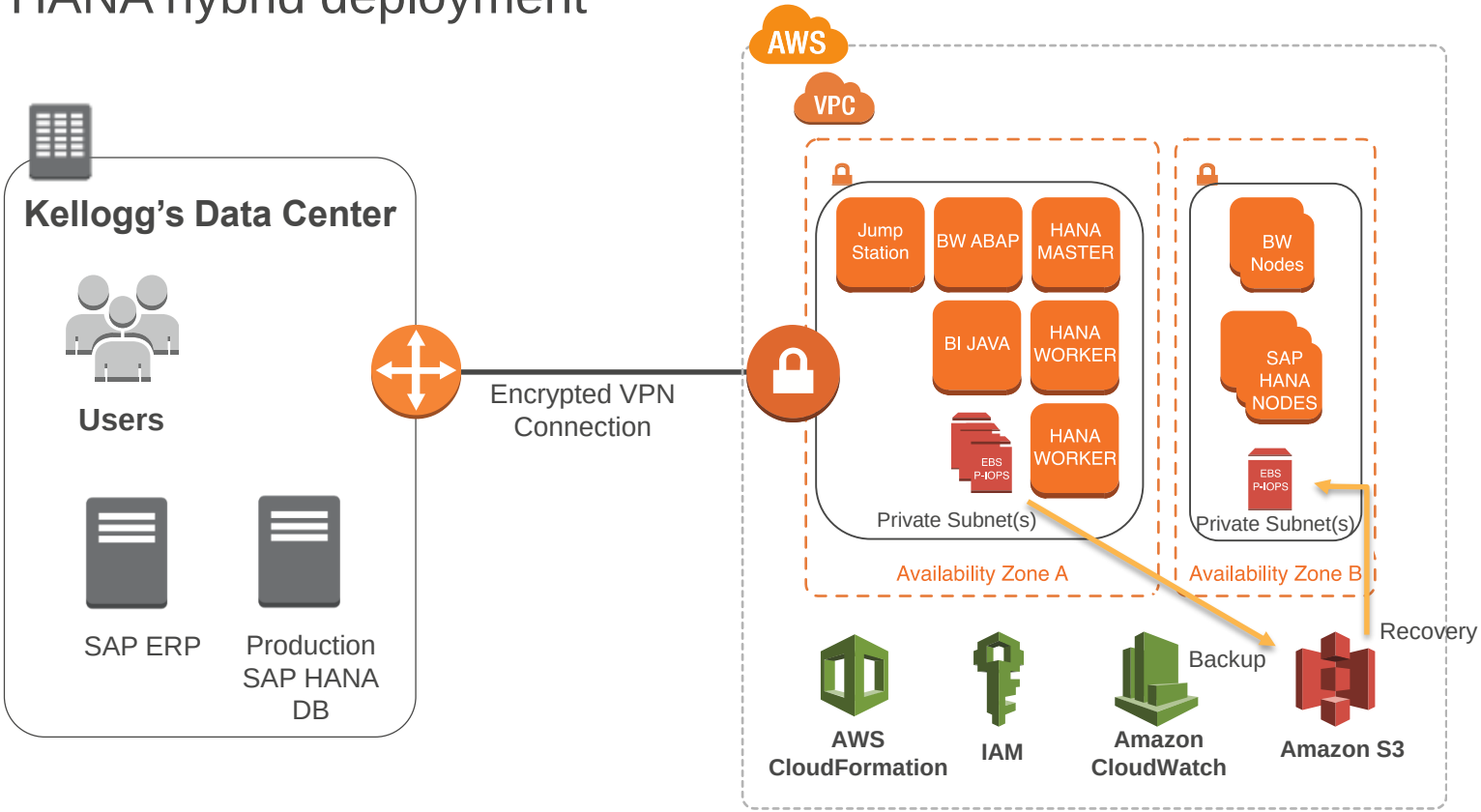
On-premises based front end

- Allows for on-premises front end, such as application based interfaces.



Customer case study: Kellogg's

SAP HANA hybrid deployment

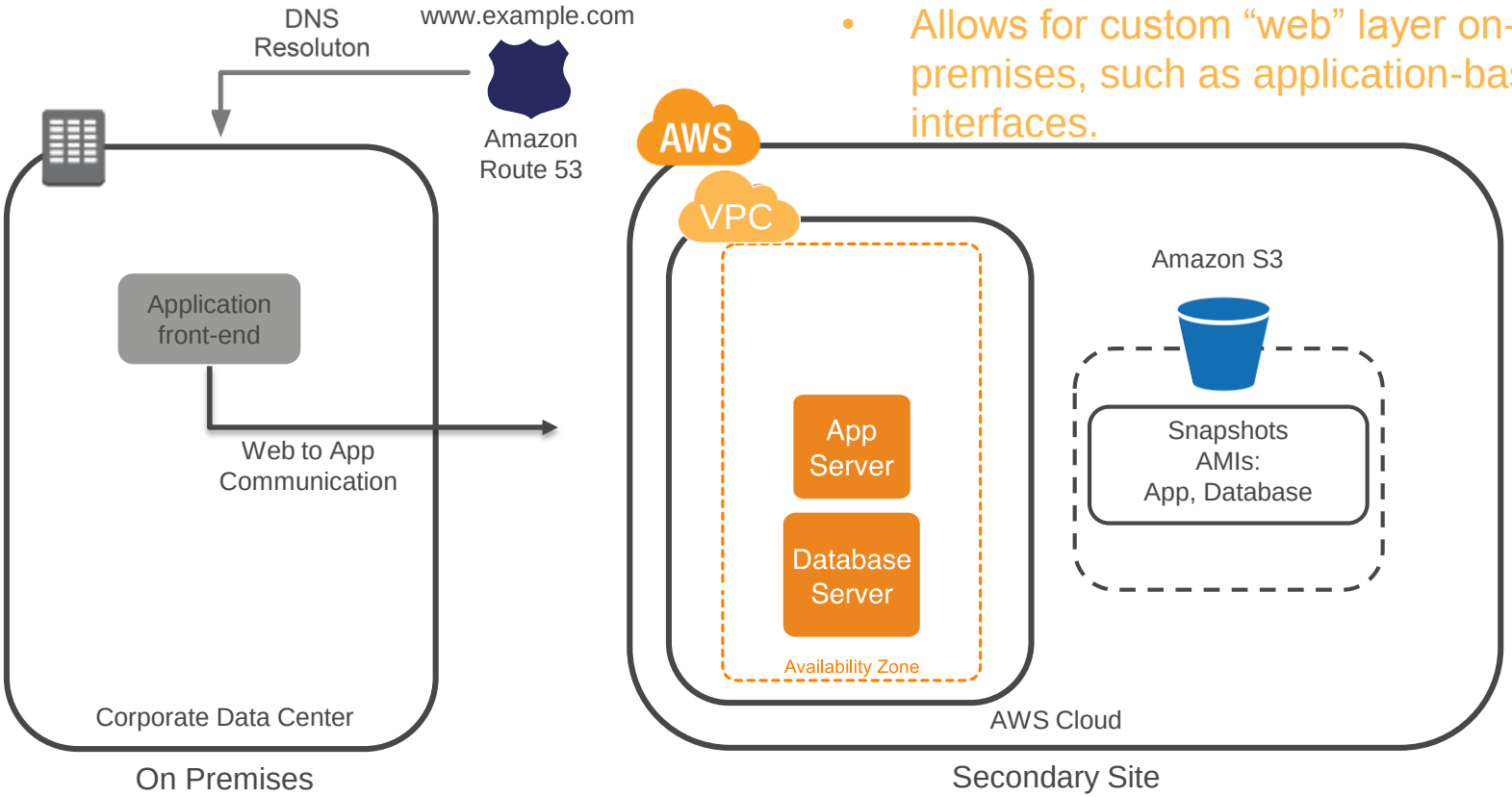


Public reference: https://aws.amazon.com/solutions/case-studies/kellogg-company/

Placing your application where it makes sense

Split-tier architecture

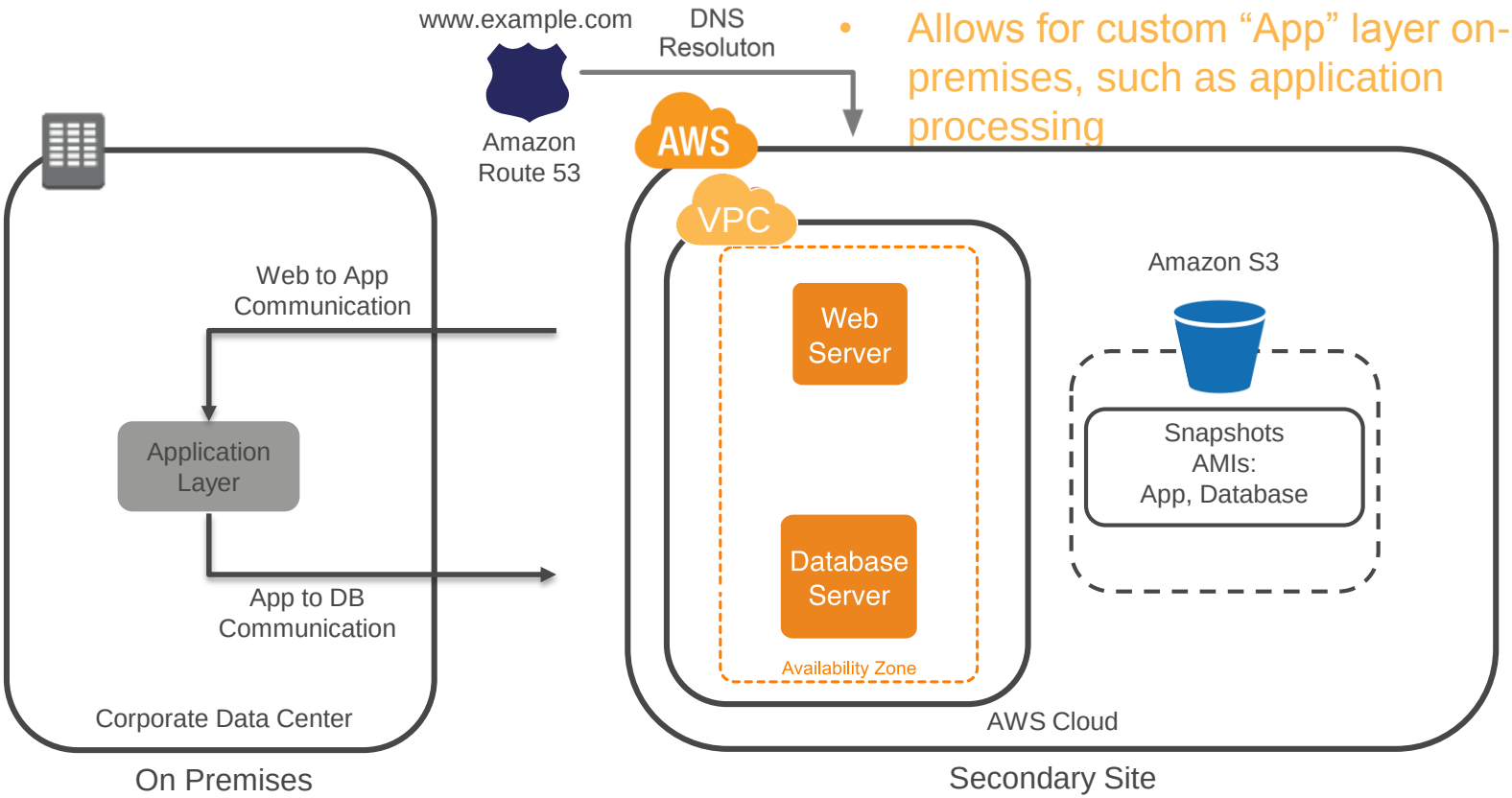
- Allows for custom “web” layer on-premises, such as application-based interfaces.



Placing your application where it makes sense

Split-tier architecture

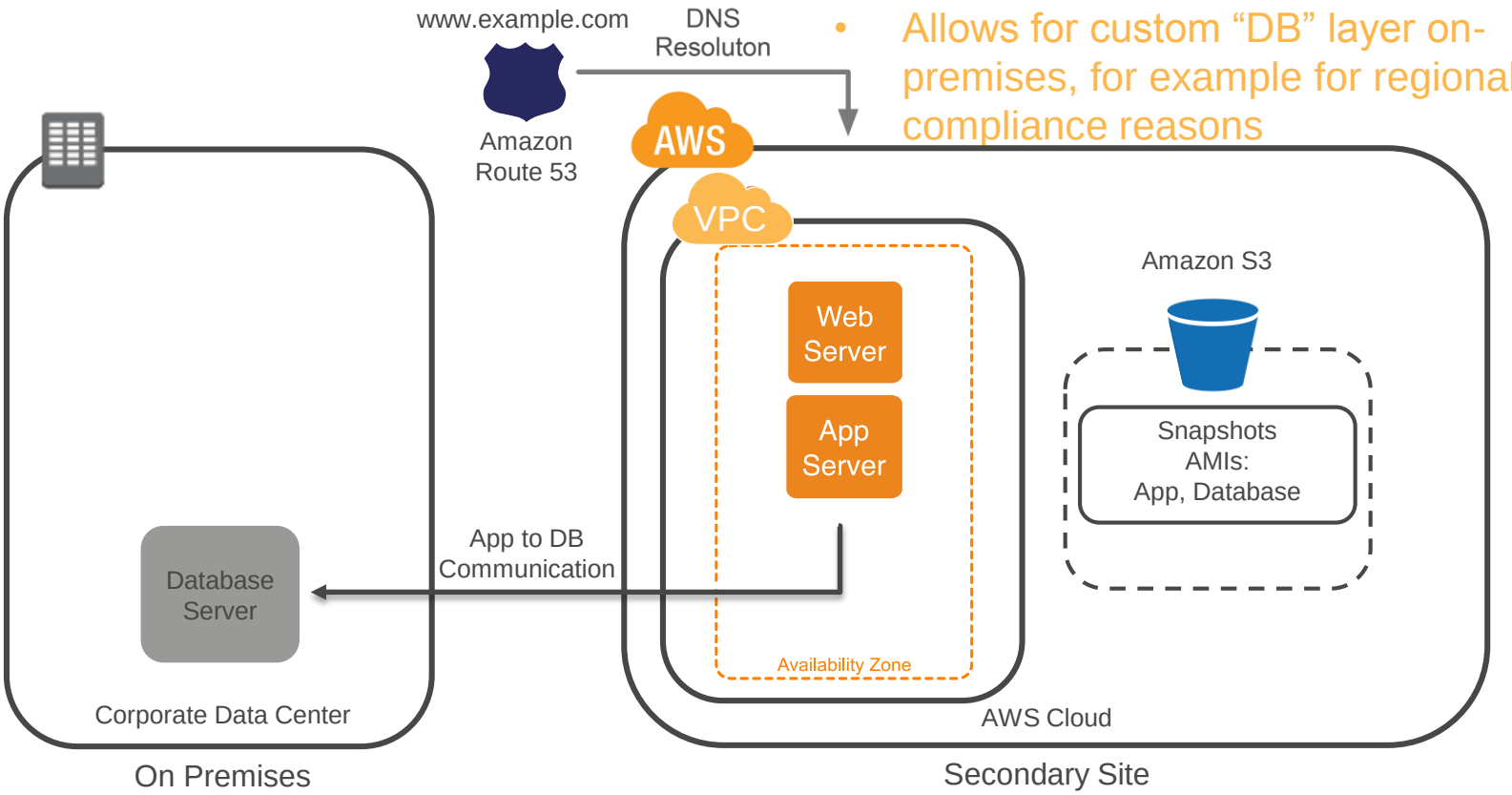
- Allows for custom “App” layer on-premises, such as application processing



Placing your application where it makes sense

Split-tier architecture

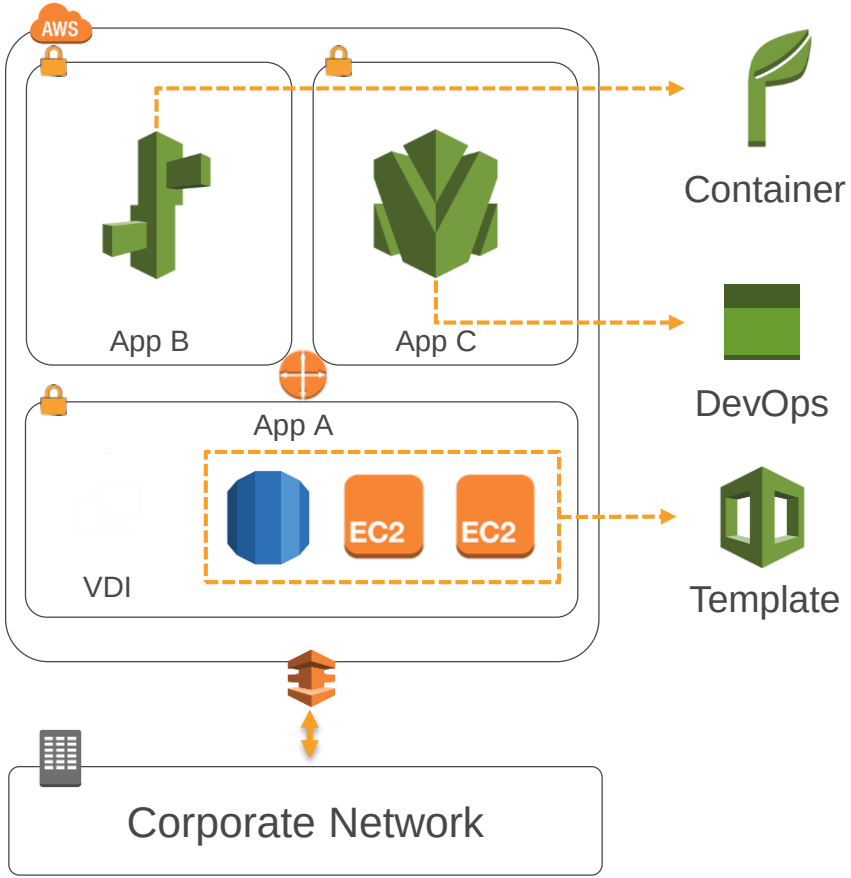
- Allows for custom “DB” layer on-premises, for example for regional or compliance reasons



Other hybrid use cases

What else can we build?

Development and test



Innovation & agility

Automated builds and deployment of code

Consistent regression testing

Numerous **disposable environments** that can be (re)built within a click allowing regression tests in identical setups

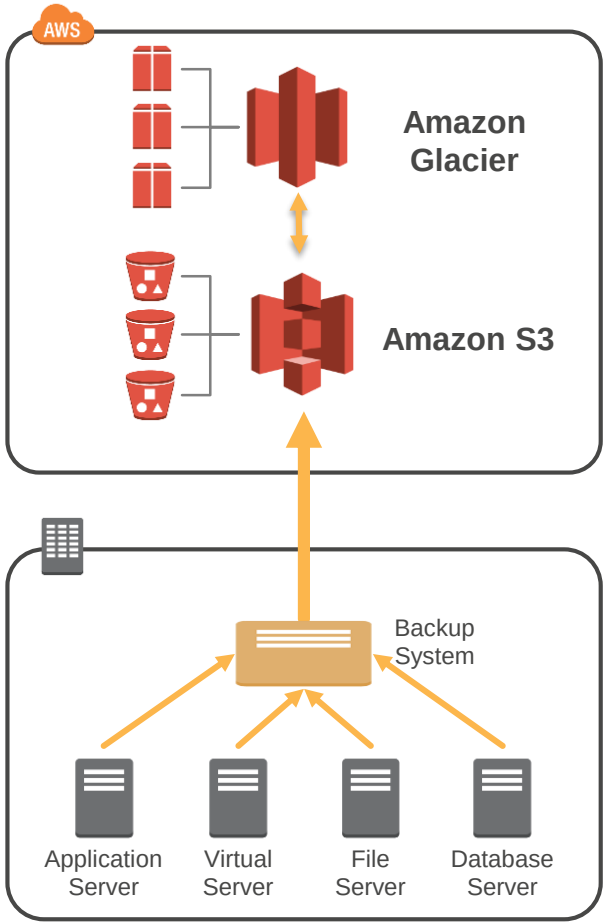
Cost-effective

Environments can be disposed or **stopped when unused**

Scalability

Conduct performance and **stress tests** with potentially thousands of simulation nodes

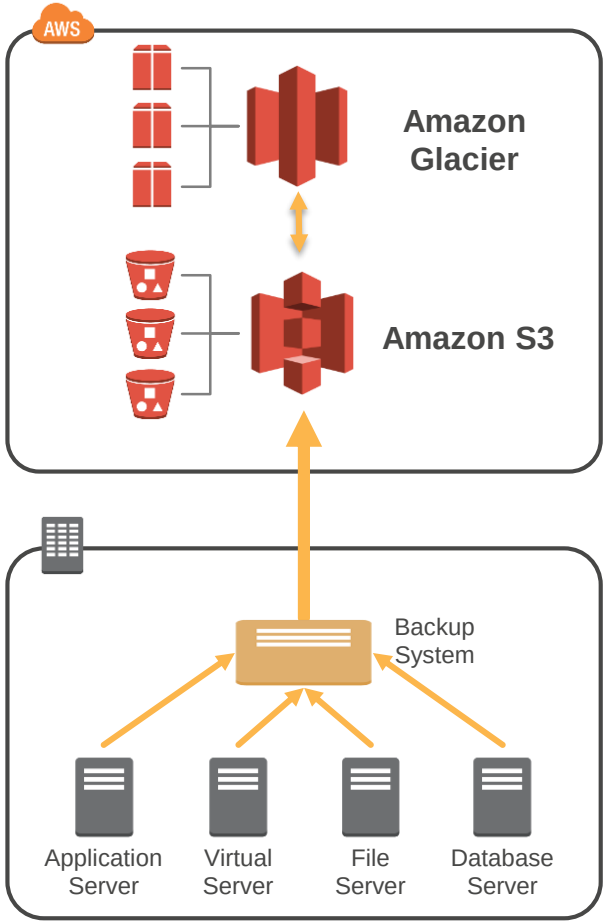
Backup and archive



Backup to cloud storage

- Eliminate tape, hardware, off-site storage
- Reduce capital expense for backup infrastructure
- Never worry about backup durability
- Never run out of backup capacity
- Data stored off-site, with high durability, in multiple locations

Backup and archive



Symantec NetBackup

Oracle RMAN and Secure Backup Module

CommVault Simpana

Veeam Backup & Replication

AltaVault (SteelStore)

Thoughts on Hybrid Architecture

- Hybrid infrastructure is key. AWS allows for full network integration and hybrid cloud architectures across on-premises and AWS.

- Reduce the heavy-lifting: Using cloud services can allow you to focus on your business and alleviate pain points in new deployments.

- Adoption is not tech but business-driven. Increased agility provides necessary reduced time-to-market.

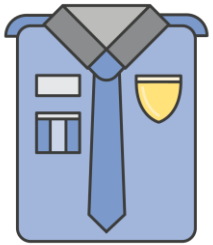
- On-premises infrastructure is not throwaway. After you move to the cloud, it’s not a cloud or no-cloud decision. You can and probably will use both.

AWS Well-Architected Framework

amazon web services

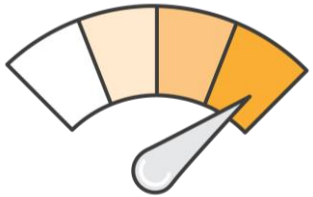
AWS well-architected framework

Set of questions you can use to evaluate how well an architecture is aligned to AWS best practices



Security

Reliability



Performance efficiency

Cost optimization

Operational excellence

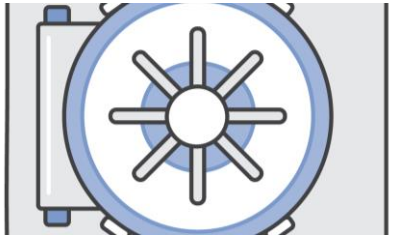
Security pillar

Protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies

Security at all layers

Enable traceability

Implement a principle of least privilege

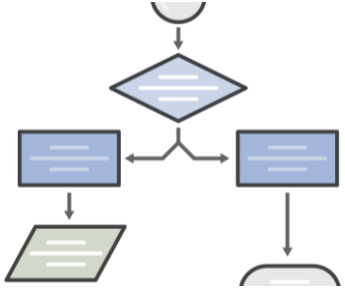


Focus on securing system

Automate security best practices

Reliability pillar

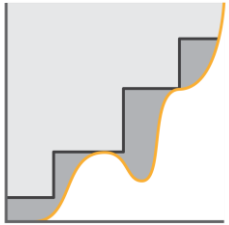
Ability of a system to recover from infrastructure or service disruptions, dynamically acquire computing resources to meet demand, and mitigate disruptions such as misconfigurations or transient network issues



Test recovery procedures

Automatically recover from failure

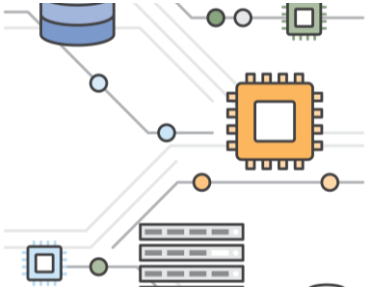
Scale horizontally to increase availability



Stop guessing capacity

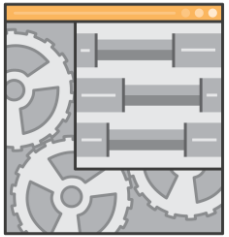
Performance efficiency pillar

Efficiently use of computing resources to meet requirements, and maintaining that efficiency as demand changes and technologies evolve



Democratize advanced technologies

Go global in minutes



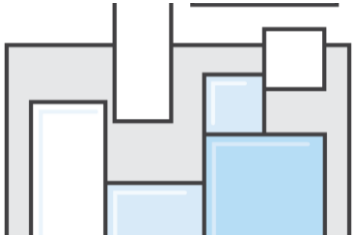
Use server-less architectures

Experiment more often

Cost optimization pillar

Assess your ability to avoid or eliminate unneeded costs or suboptimal resources, and use those savings on differentiated benefits for your business

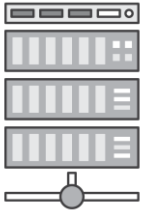
Analyze and attribute expenditure



Managed services to reduce TCO

Adopt a consumption model

Benefits from economies of scale



Stop spending money on data center operations

Operational excellence pillar

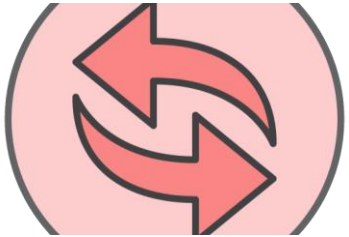
Operational practices and procedures used to manage production workloads

Perform operations with code

Align operations processes to business objectives

Make regular, small, incremental changes

Test for responses to unexpected events



Learn from operational events and failures

Keep operations procedures current

National Instruments: Achieving Agility

amazon web services

NATIONAL INSTRUMENTS™

NI equips engineers and scientists with systems that accelerate productivity, innovation, and discovery

NATIONAL INSTRUMENTS LabVIEW

40-year-old company headquartered in Austin, TX; annual sales greater than $1.25 B

"Products used from toys to supercolliders"

Cloud journey

Started developing on platform in 2008

FPGA Compile Cloud - August 2010
LabVIEW Web UI Builder - November 2010

2013 – Introduced well-architected design

2014 – Launched well-architected product

2015 – All products followed well-architected framework

National Instruments: Cloud Infrastructure 2012

amazon web services

Cloud infrastructure 2012

EC2-Classic, Elastic Load Balancing, Amazon S3, Amazon SimpleDB

MySQL on EC2

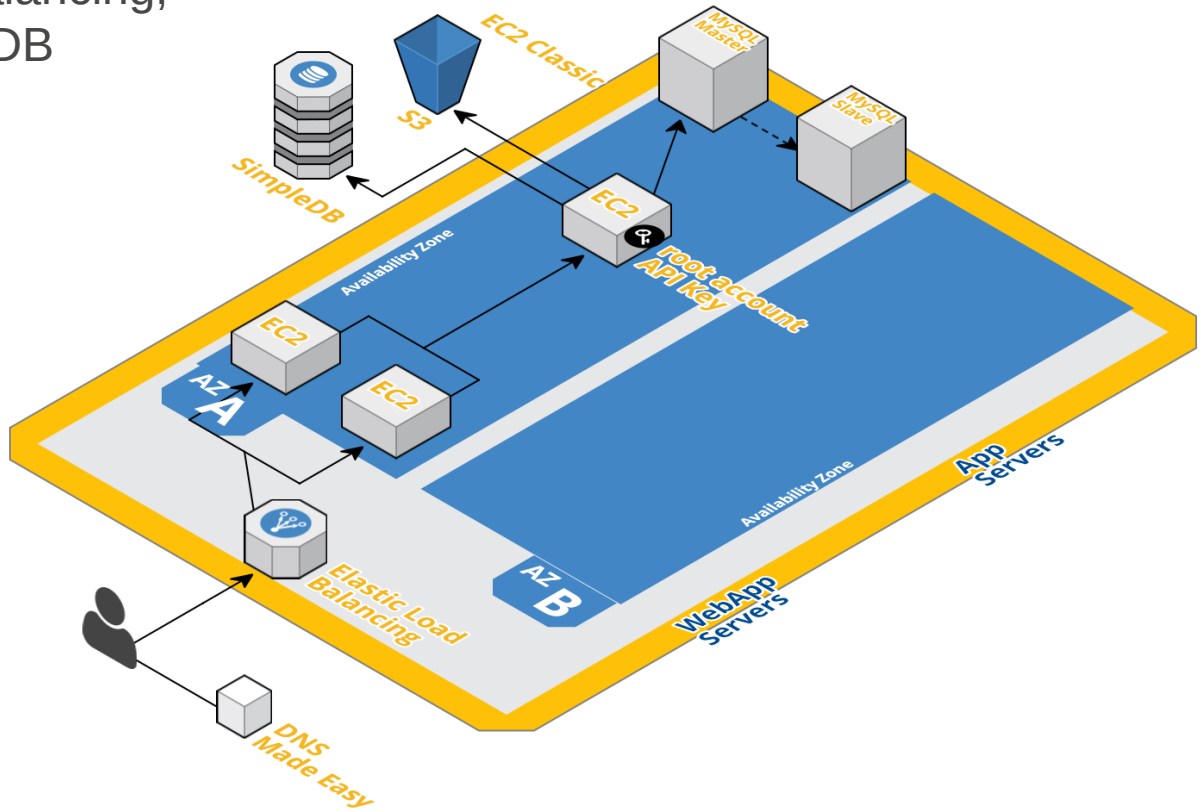
“Root” credentials

Single-AZ

Internally developed tooling

Backups sent to data center

Manual AMI creation



Cloud infrastructure 2012: Challenges

Deployment of infrastructure was manual, resource intensive, and prone to error

Software deployment took 5–30 minutes

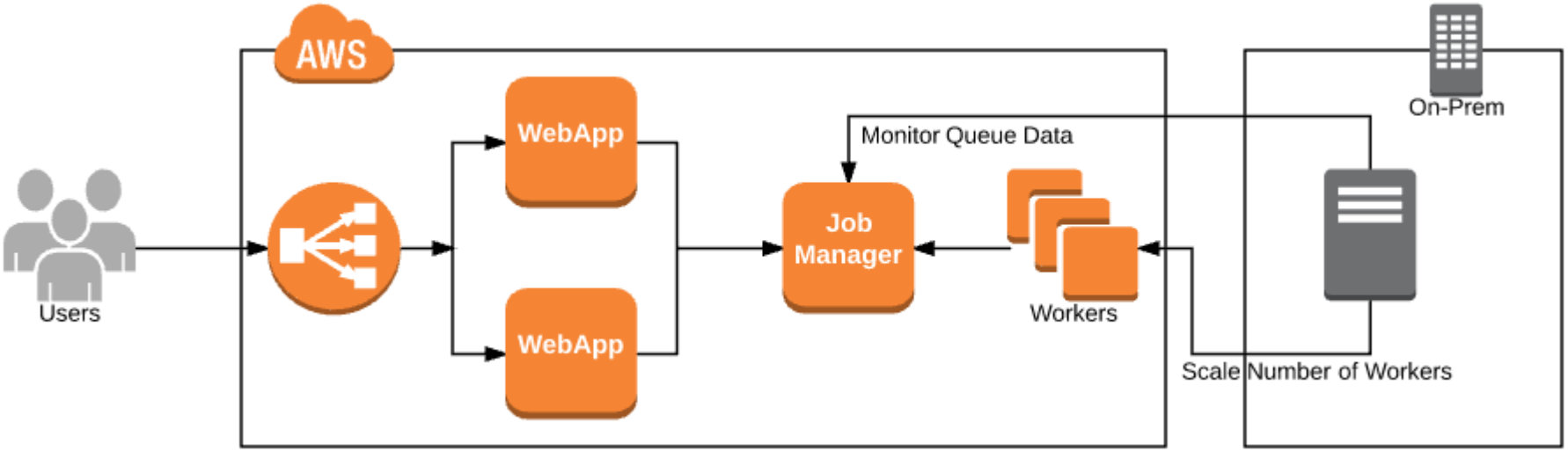
Lack of infrastructure automation

Scaling took 10–30 minutes to meet demand

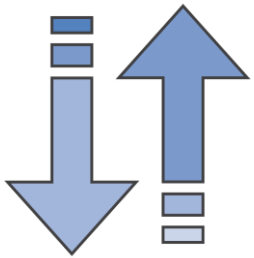
FPGA Compile Cloud: Scaling for Growth

amazon web services

FPGA Compile Cloud: Original scaling design



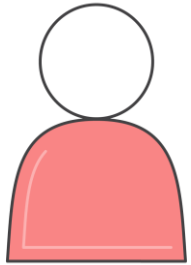
FPGA Compile Cloud: Challenges



Increased demand causing scaling issues

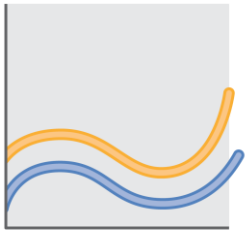
Delayed results

Alert fatigue



Manual intervention

FPGA Compile Cloud: Improvement



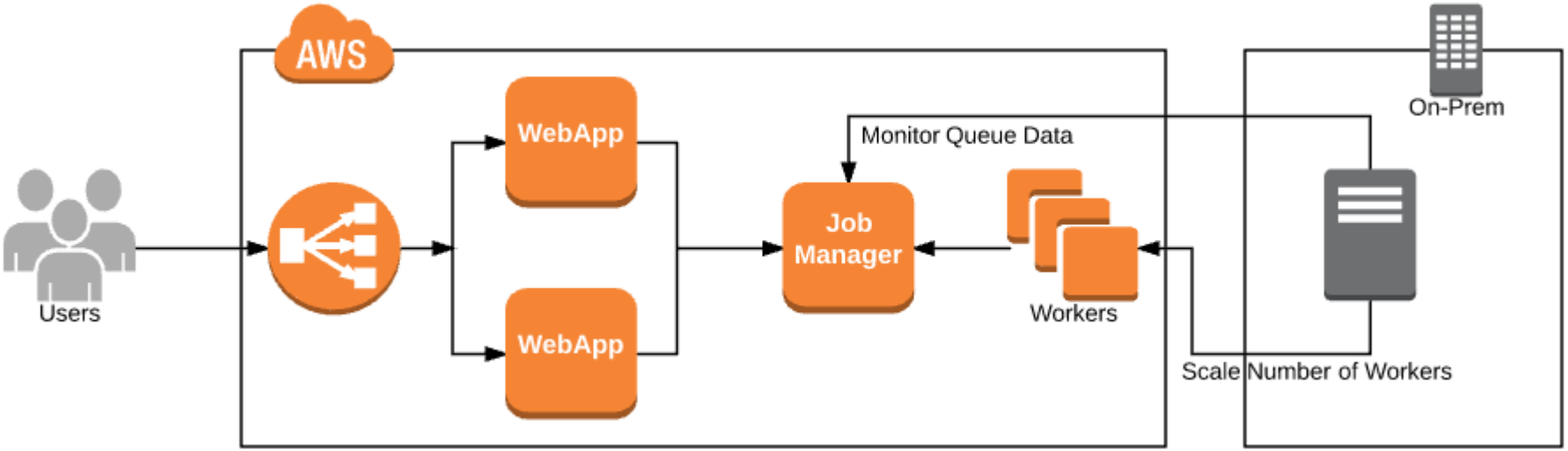
How can we better match demand?

What can be made faster?

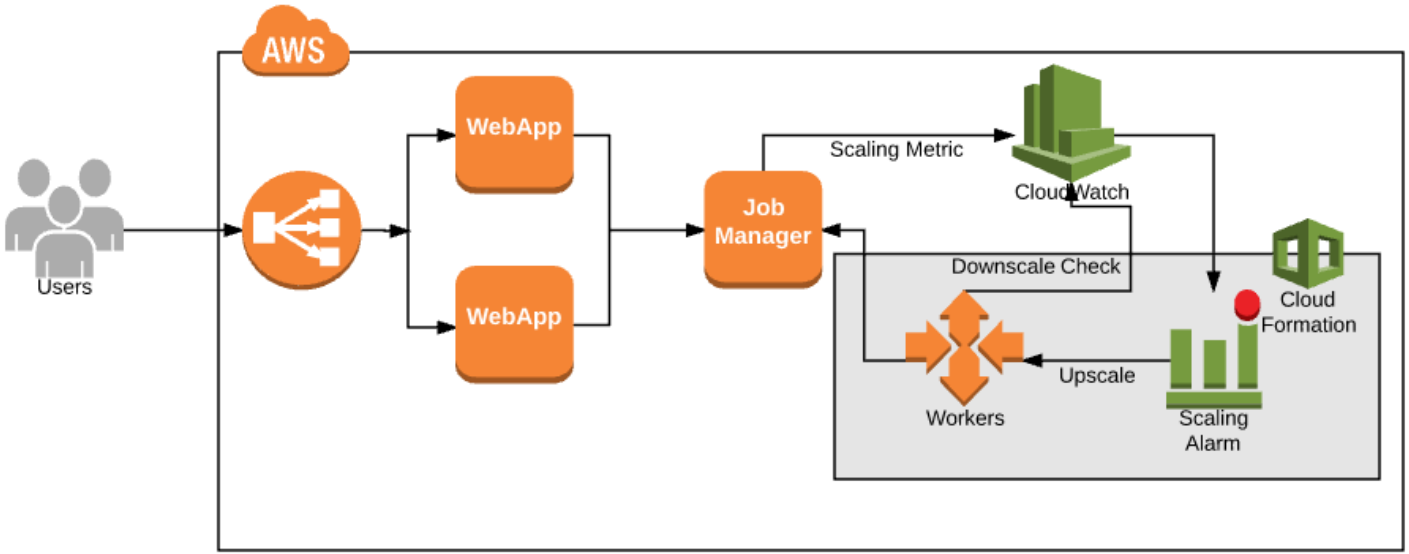
How to reduce alerts?

How to automate it?

FPGA Compile Cloud: Original scaling design



FPGA Compile Cloud: Improved scaling design



Scaling to meet demand

Autonomous instances

Intelligent monitoring

Automated deployment

Benefits from well-architected framework

Decreased scaling latency from 30 minutes to 5

Optimized cost from overprovisioning

Removed data center dependency

Increased developer efficiency

Adopting Well-Architected Framework from Product Inception

amazon web services

Cloud infrastructure 2014

Cloud native services: VPC, Auto Scaling, Amazon Route 53, Amazon CloudFront

RDS-MySQL

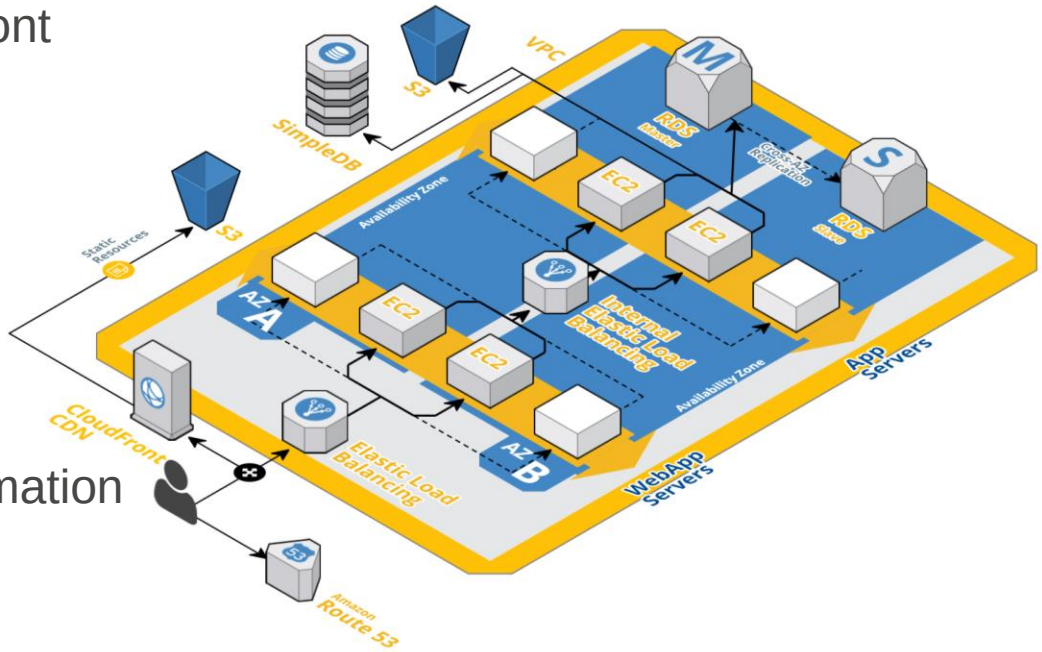
Least-privileged access: IAM

Multi-AZ

Cloud native tooling: AWS CloudFormation

Automated AMI process: Ansible

Adopted DevOps principles: Created CI/CD pipelines



Benefits from well-architected framework

Load tested above production capacity

Reduced attack surface

Cost optimized

Faster updates; decreased time to market

Migrate Existing Products to Well-Architected Framework

amazon web services

Existing products: Desired changes

Area	2012	2015 – Well architected
EC2	Classic	VPC
Relational database	MySQL on EC2	RDS-MySQL
Auto Scaling	Zero	Everything
Elastic Load Balancing	External only	Everything
CloudFormation	Zero	95%
AMI creation	Manual	Automated: Ansible
Application deployment	Manual	AWS CodeDeploy

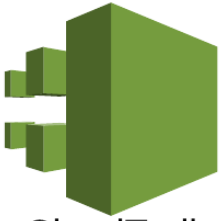
Existing Products: Measured Improvements

Area	2012	2015 – Well architected
Security	Root API key	IAM, Network ACL, Egress filtering
Single point of failure	10+	1
Time to create separate environment	1 month	< 2 hours
Longest code deployment time	2 weeks	< 4 hours
Typical code deployment time	15 minutes	< 1 minute

Continuous Improvement from Well-Architected Framework

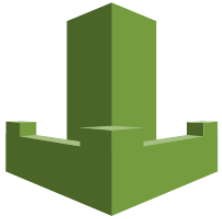
amazon web services

Road map: Additional security



CloudTrail

Amazon Inspector



AWS WAF

Road map: Multiregion disaster recovery

Region specific

```
452    "PublicSubnetA0" : {
453      "Type" : "AWS::EC2::Subnet",
454      "Properties" : {
455        "AvailabilityZone" : "us-east-1a",
456        "CidrBlock" : { "Ref" : "PublicSubnetA0Cidr" },
457        "Tags" : [
```

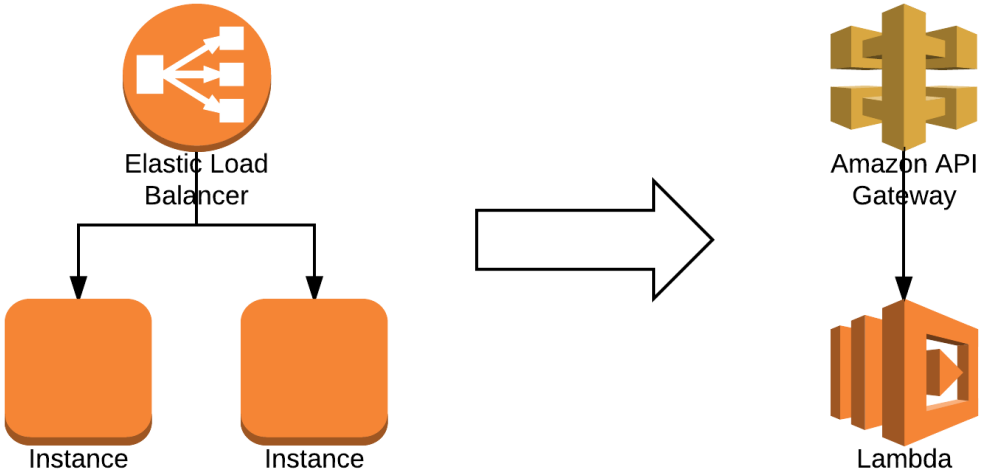
Line 452, Column 1 | Spaces: 2 | Plain Text

Region agnostic

```
756    "PublicDmz1Subnet1" : {
757      "Type" : "AWS::EC2::Subnet",
758      "Properties" : {
759        "AvailabilityZone": { "Fn::Select": [ "0", { "Fn::GetAZs": "" } ] },
760        "CidrBlock" : { "Ref" : "PublicDmz1Subnet1Cidr" },
761        "MapPublicIpOnLaunch" : true,
762        "Tags" : [
```

Line 756, Column 1 | Spaces: 2 | Plain Text

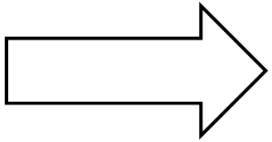
Road map: Simpler, more efficient



Road map: Simpler, more efficient

NAT
Instance

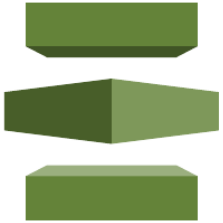
Several hundred lines
in CloudFormation



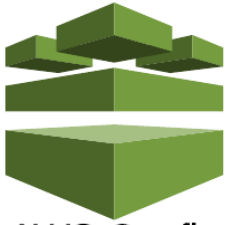
NAT
Gateway

Twelve lines
in CloudFormation

Road map: Simpler, more efficient



AWS Certificate Manager



AWS Config

Lessons Learned Utilizing Well-Architected Framework

amazon web services

Most valuable lessons learned

Be willing to make change

Know when architecture is nearing its limits

Take appropriately sized steps

Don’t reinvent the wheel

Most valuable lessons learned

Invest time to save time

Automation empowers faster change and improvement

Need qualified people to accomplish

It’s a journey, not a destination

Resources

https://aws.amazon.com/well-architected/

AWS Well-Architected

The Well-Architected framework has been developed to help cloud architects build the most secure, high-performing, resilient, and efficient infrastructure possible for their applications. This framework provides a consistent approach for customers and partners to evaluate architectures, and provides guidance to help implement designs that will scale with your application needs over time.

Build and deploy faster

Stop guessing capacity needs, test systems at scale, and use automation to make experimentation easier by building cloud-native architectures.

Lower or mitigate risks

Understand where you have risks in your architecture, and address them before your applications are put into production.

Make informed decisions

Determine how architectural decisions and/or trade-offs might impact the performance and availability of your applications and business outcomes.

Learn AWS best practices

Access training and whitepapers that provide guidance based on what we have learned through reviewing thousands of customers' architectures on AWS.

AWS

SUMMIT

Thank you!

amazon web services