Developer Operations

AWS Services

Compute
EC2
Elastic Container Service
Lightsail
Elastic Beanstalk
Lambda
Batch

Storage
S3
EFS
Glacier
Storage Gateway

Database
RDS
DynamoDB
ElastiCache
Amazon Redshift

Networking & Content Delivery
VPC
CloudFront
Direct Connect
Route 53

Migration
AWS Migration Hub
Application Discovery Service
Database Migration Service
Server Migration Service
Snowball

Developer Tools
CodeStar
CodeCommit
CodeBuild
CodeDeploy
CodePipeline
X-Ray

Management Tools
CloudWatch
CloudFormation
CloudTrail
Config
OpsWorks
Service Catalog
Trusted Advisor
Managed Services

Security, Identity & Compliance
IAM
Inspector
Certificate Manager
Directory Service
WAF & Shield
Artifact
Amazon Macie
CloudHSM

Analytics
Athena
EMR
CloudSearch
Elasticsearch Service
Kinesis
Data Pipeline
QuickSight
AWS Glue

Artificial Intelligence
Lex
Amazon Polly
Rekognition
Machine Learning

Internet Of Things
AWS IoT
AWS Greengrass

Contact Center
Amazon Connect

Game Development
Amazon GameLift

Mobile Services
Mobile Hub
Cognito
Device Farm
Mobile Analytics
Pinpoint

Application Services
Step Functions
SWF
API Gateway
Elastic Transcoder

Messaging
Simple Queue Service
Simple Notification Service
Simple Email Service

Business Productivity
WorkDocs
WorkMail
Amazon Chime

Desktop & App Streaming
WorkSpaces
AppStream 2.0

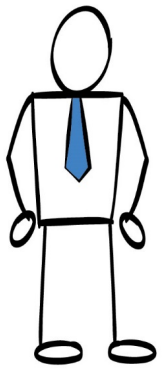
Mar. 2019 – 100+ Services

AWS Shared Responsibility Model

Responsible for security ***'in'*** the Cloud	AWS services ***you*** utilize to run your workloads (EC2 instances, S3 buckets, etc.)
Responsible for security ***'of'*** the Cloud	Policies and mechanisms ***AWS*** uses to protect the cloud itself

AWS Shared Responsibility Model

AWS Shared Responsibility Model



Customer

Responsible for security *'in'* the Cloud

AWS services ***you*** utilize to run your workloads (EC2 instances, S3 buckets, etc.)

AWS

Responsible for security *'of'* the Cloud

Compute | Storage | Database | Networking

AWS Global Infrastructure

Regions

Availability Zones

Edge Locations

AWS Shared Responsibility Model

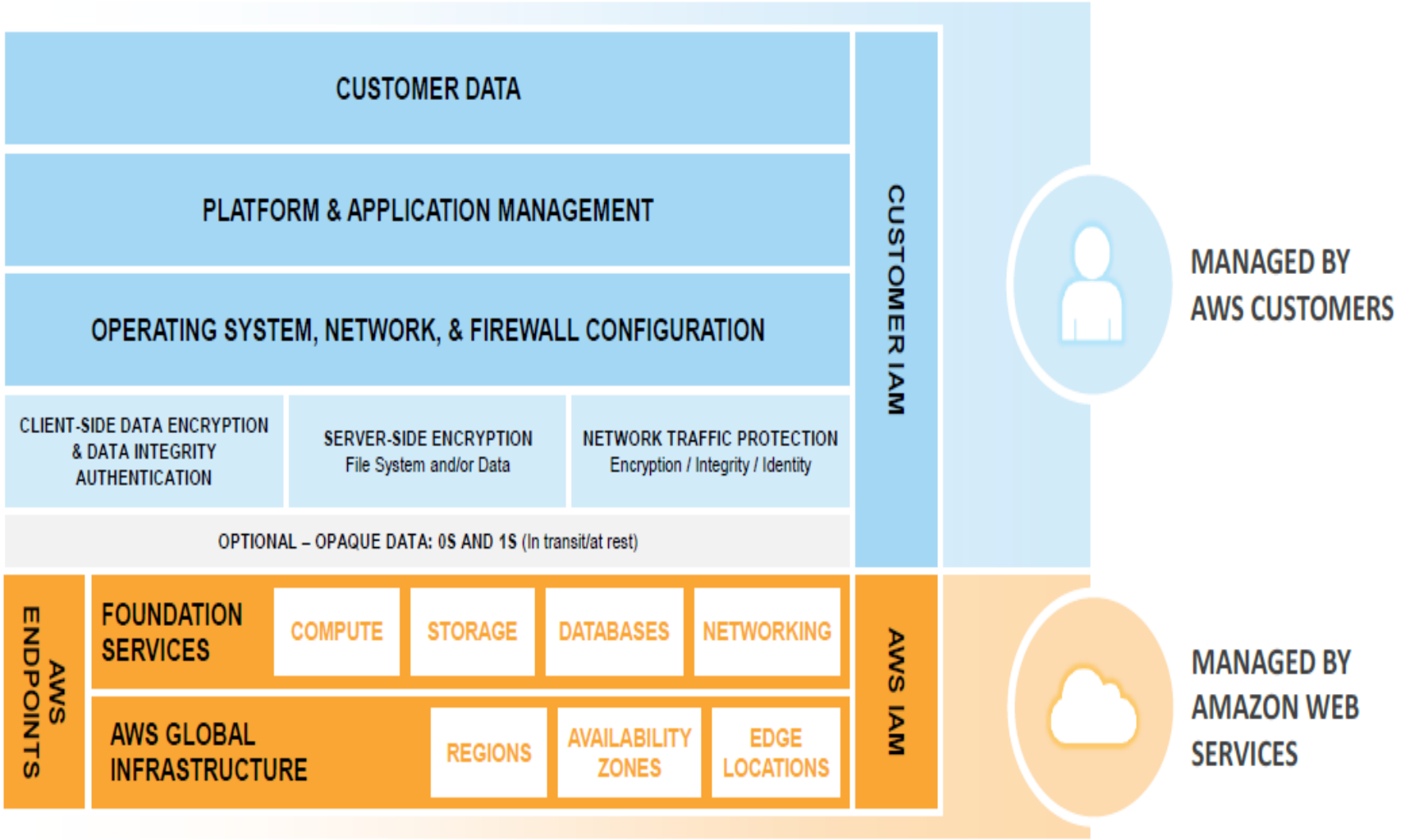


Figure 1: Shared Responsibility Model for Infrastructure Services

Manual Configuration Challenges

Creating and configuring AWS services and resources through a management console is a **manual** process

What are the challenges and concerns for a manual process?

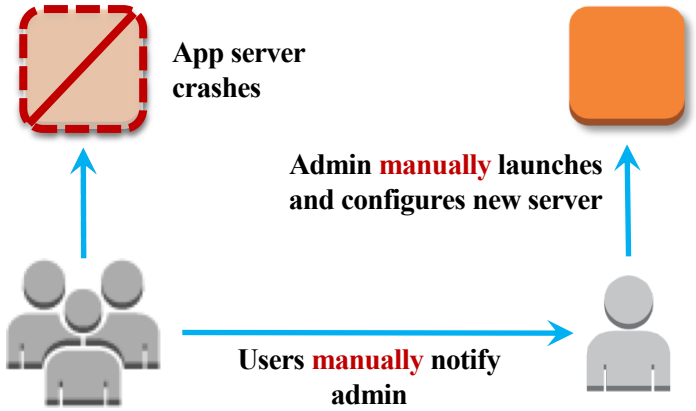
- Reliability
- Reproducibility
  - DEV
  - TEST
  - PROD
- Documentation

Best Practice: Automate Your Environment

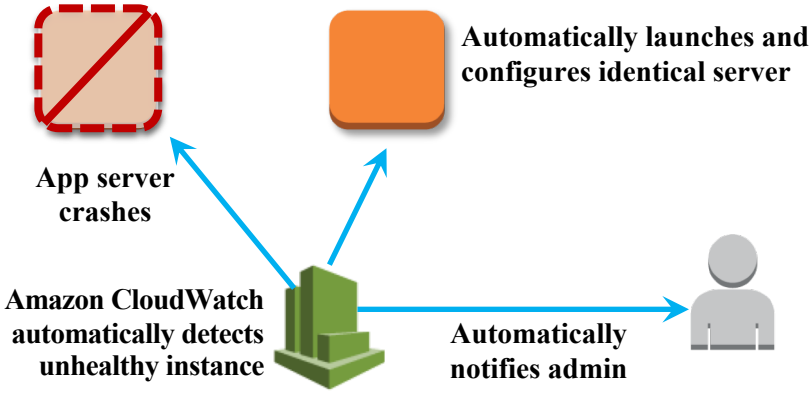
Where possible, automate the provisioning, termination, and configuration of resources.

Improve your system's stability and consistency, as well as the efficiency of your organization, by removing manual processes.

Anti-pattern



Best practice



Best Practice: Use Disposable Resources

Take advantage of the dynamically provisioned nature of cloud computing.

Think of servers and other components as temporary resources.

Anti-pattern

- Over time, different servers end up in different configurations
- Resources run when not needed
- Hardcoded IP addresses prevent flexibility
- Difficult/inconvenient to test new updates on hardware that's in use

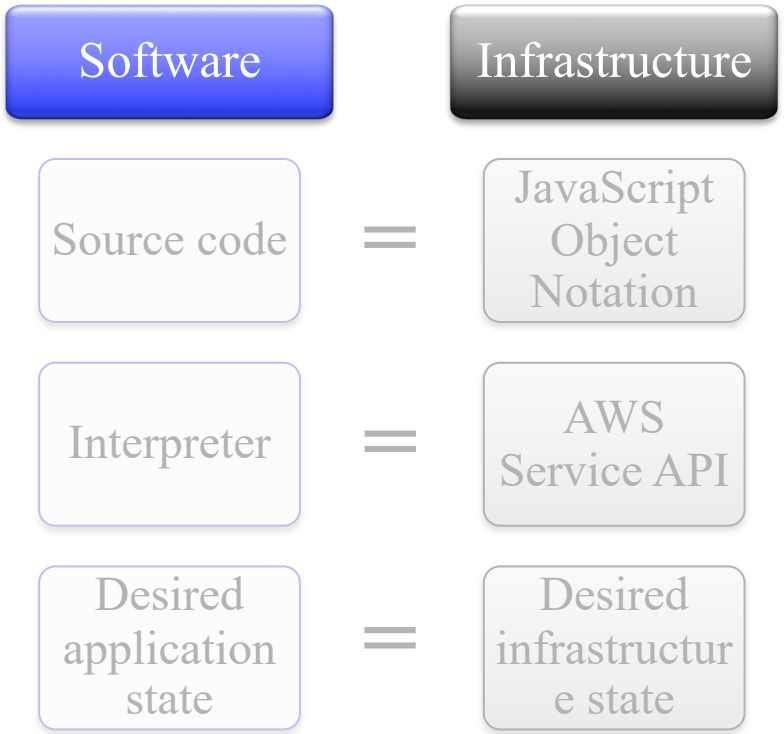
Best practice

- Automate deployment of new resources with identical configurations
- Terminate resources not in use
- Switch to new IP addresses automatically
- Test updates on new resources, and then replace old resources with updated ones

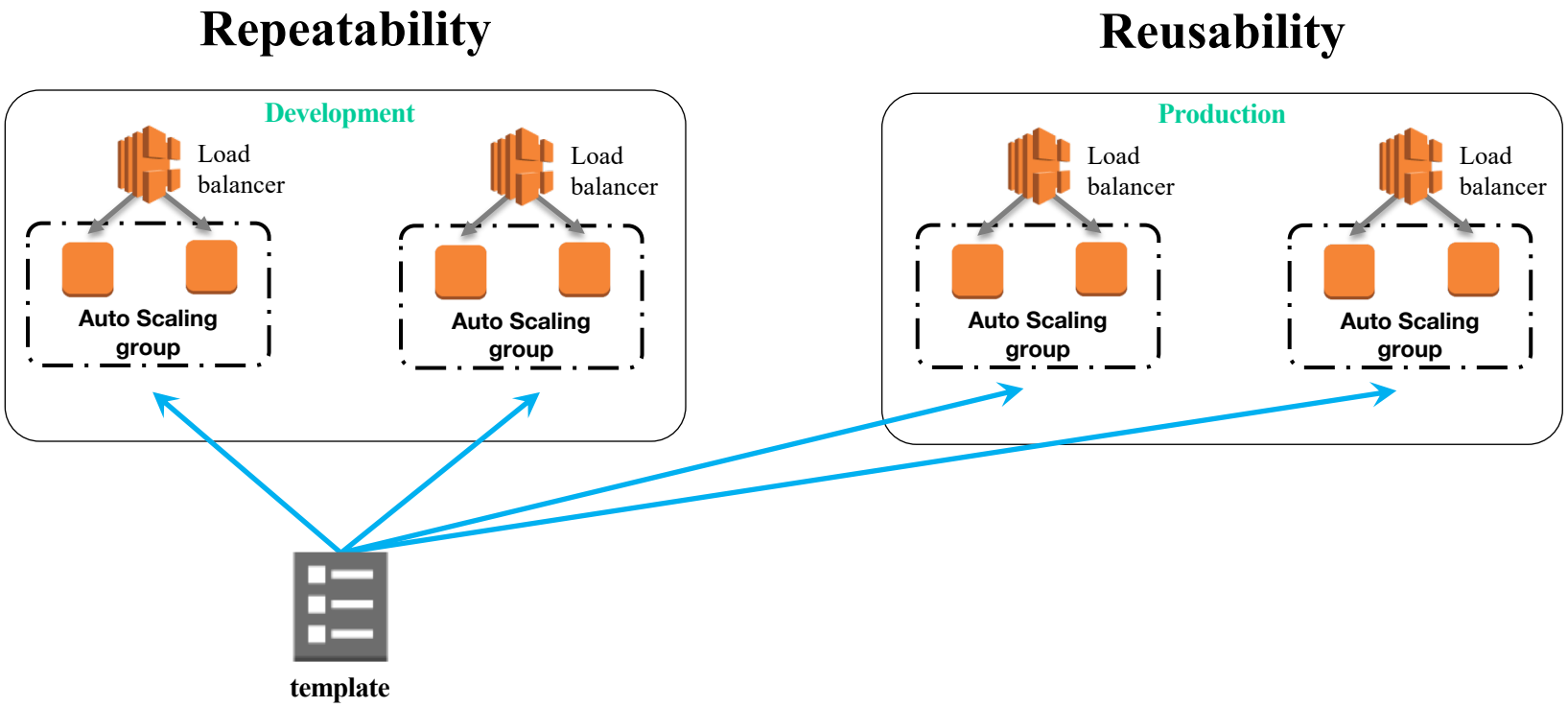
What Does Infrastructure As Code Mean?

Techniques, practices, and tools from software development applied to creating reusable, maintainable, extensible and testable infrastructure.

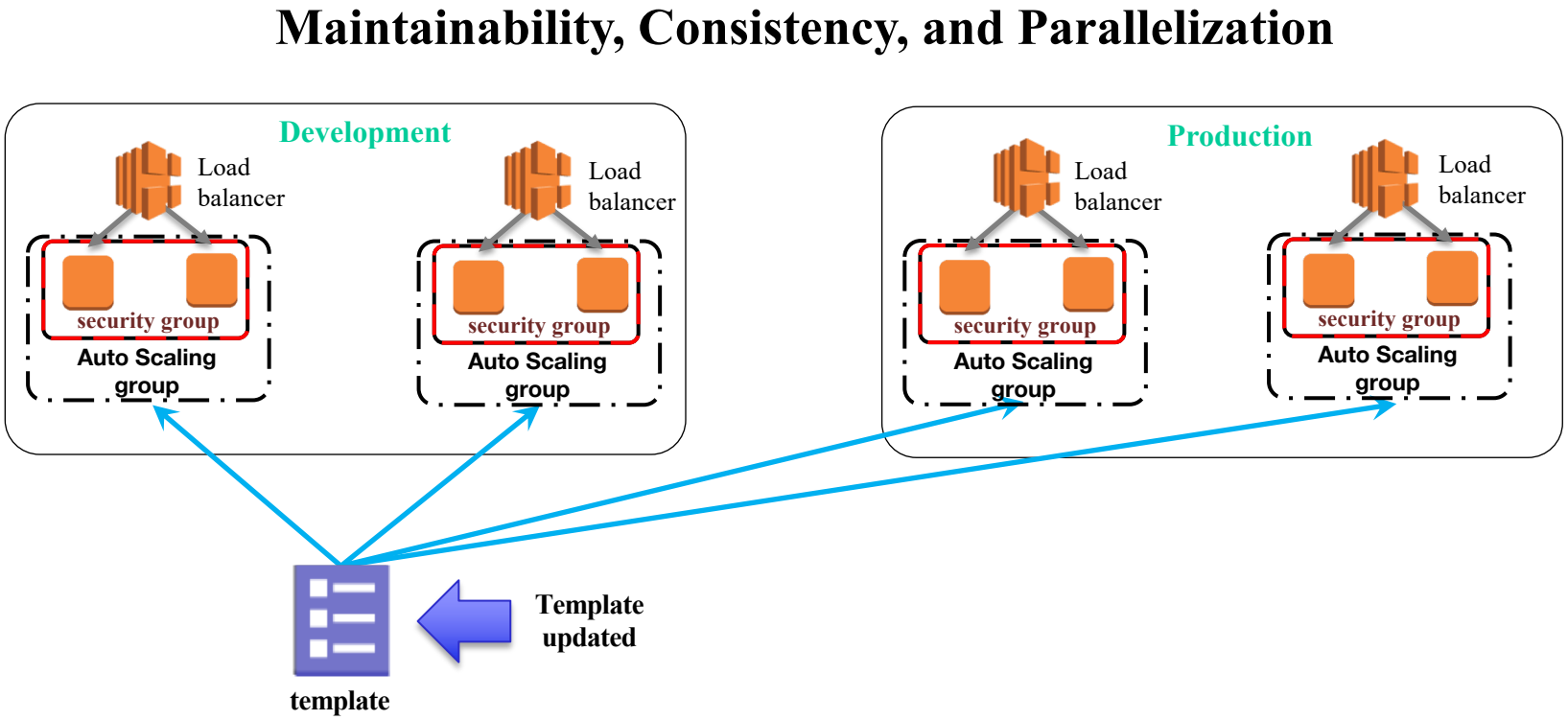
Build And Operate Your Infrastructure Like Software



Benefits Of Treating Infrastructure As Code



Benefits Of Treating Infrastructure As Code



Infrastructure as Code : Idempotent

- Many popular tools used for Infrastructure as Code :
  - Ansible
  - Chef
  - Terraform
  - Puppet
- Ansible documentation – Idempotency : An operation is idempotent if the result of performing it once is exactly the same as the result of performing it repeatedly without any intervening actions.
- Compares desired state with current state to assess if changes required
- Is CloudFormation Idempotent ? Initially was not but now maybe. Partially ?

CloudFormation: Infrastructure As Code

AWS CloudFormation allows you to launch, configure, and connect AWS resources with JavaScript Object Notation (JSON) and YAML-formatted templates.

Template

- JSON-formatted file describing the resources to be created
- Treat it as source code: put it in your repository

AWS CloudFormation Engine

- AWS service component
- Interprets AWS CloudFormation template into stacks of AWS resources

Stack

- A collection of resources created by AWS CloudFormation
- Tracked and reviewable in the AWS Management Console

Cloudformer

- CloudFormer is a template creation beta tool that creates an AWS CloudFormation template from existing AWS resources in your account. You select any supported AWS resources that are running in your account, and CloudFormer creates a template in an Amazon S3 bucket.

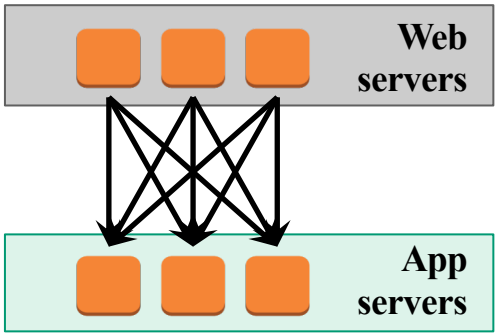
- Use CloudFormer to produce templates that you can use as a starting point. Not all AWS resources or resource properties are supported.

Loosely Couple Your Components

Design architectures with independent components.

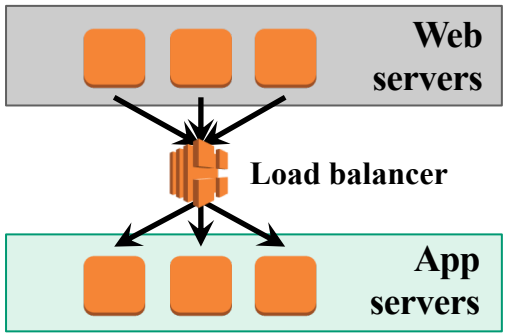
Reduce interdependencies so that the change or failure of one component does not affect other components.

Anti-pattern



Web servers **tightly coupled** to app servers

Best practice



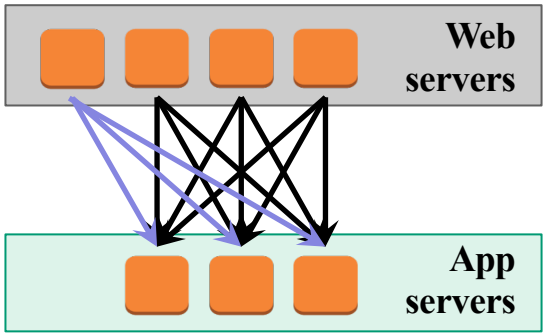
Decoupled with a load balancer

Decoupling

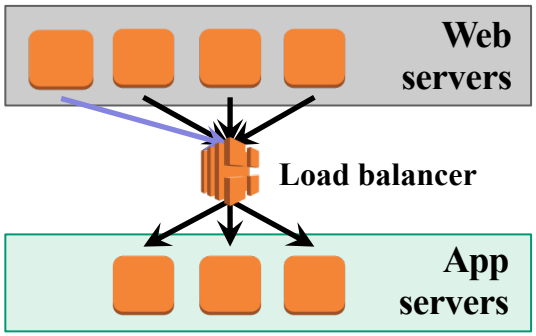
The more loosely your system is coupled…

the more easily it scales.

Tightly coupled



Loosely coupled



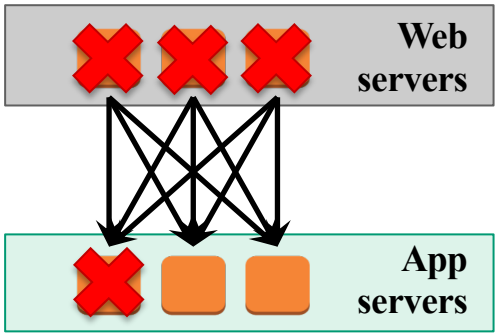
Decoupling

The more loosely your system is coupled…

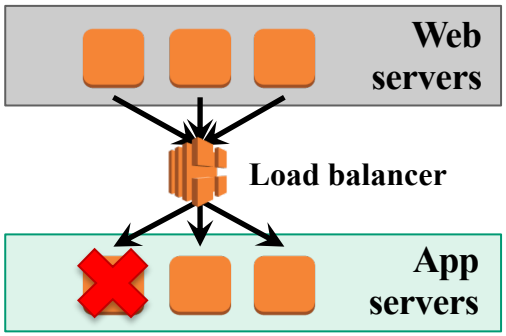
the more easily it scales.

the more fault-tolerant it can be.

Tightly coupled

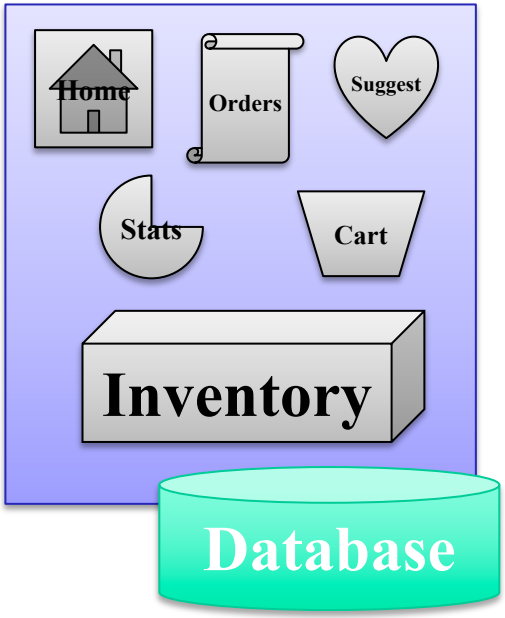


Loosely coupled

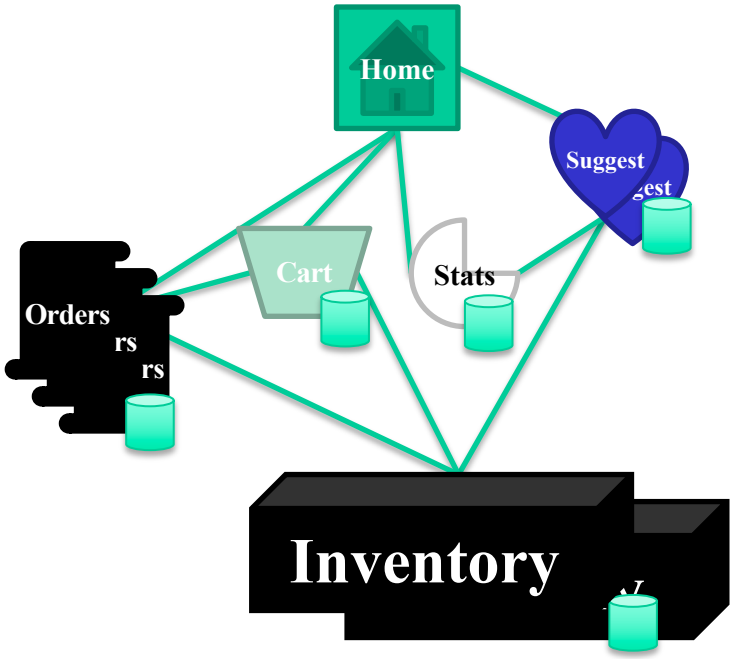


Comparing Architectural Styles

Traditional app architectures are monolithic:

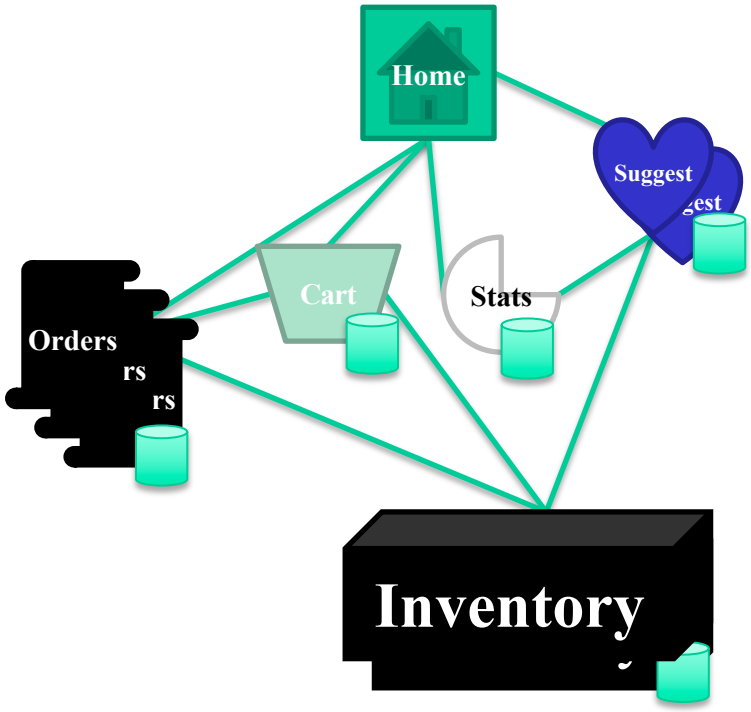


Microservice-based architectures are loosely coupled:

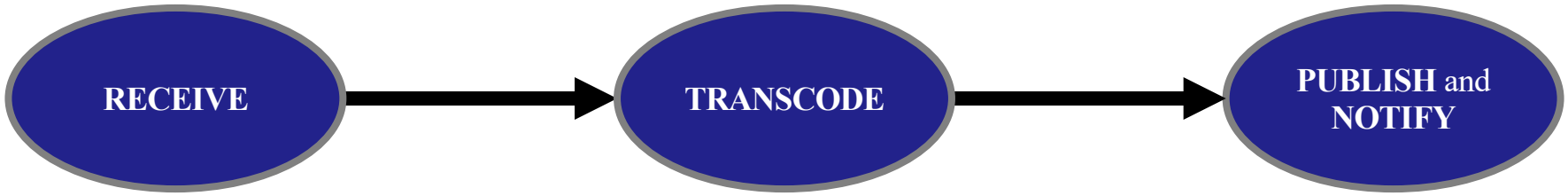


Microservices

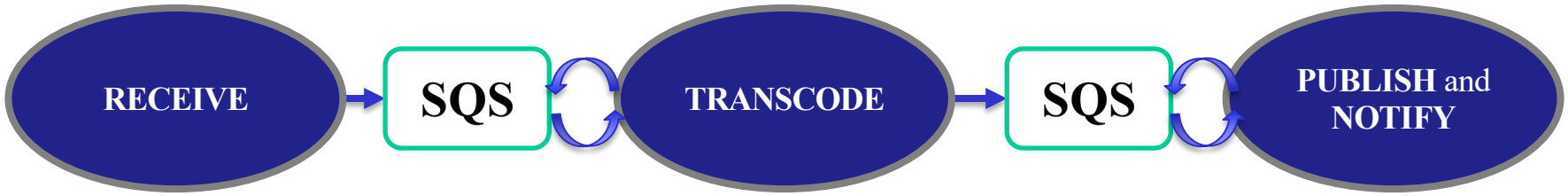
- Split features into individual components
- Have smaller parts to iterate on
- Have a reduced test surface area
- Benefit from a lower risk of change
- Use individual horizontally scalable parts



Tightly Coupled Systems



Loosely Coupled Systems



Amazon Simple Queue Service (SQS)

Amazon SQS is a fully managed message queueing service. Transmit any volume of messages at any level of throughput without losing messages or requiring other services to be always available.

Messages

- Generated by one component to be consumed by another.
- Can contain 256 KB of text in any format.

Amazon SQS

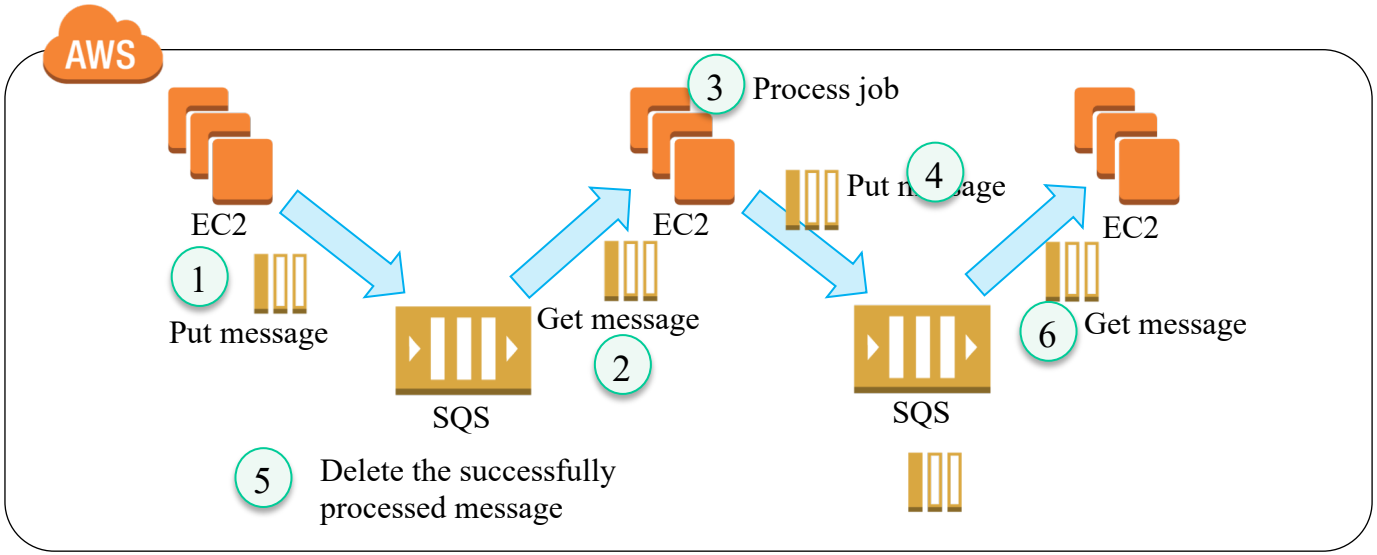
- Ensures delivery of each message at least once.
- Supports multiple readers and writers on the same queue.
- Does not guarantee first in, first out.

Queues

- Repository for messages awaiting processing.
- Acts as a buffer between the components which produce and receive data.

Loose Coupling With Amazon SQS

The queuing chain pattern enables asynchronous processing:



Amazon Simple Notification Service (SNS)

Amazon SNS enables you to set up, operate, and send notifications to subscribing services other applications.

- Messages published to topic
- Topic subscribers receive message

Subscriber types:

- Email (plain or JSON)
- HTTP/HTTPS
- Short Message Service (SMS) clients (USA only)
- Amazon SQS queues
- Mobile push messaging
- AWS Lambda Function

Characteristics Of Amazon SNS

- Single published message
- Order is not guaranteed
- No recall
- HTTP/HTTPS retry
- 256 KB max per message

How Is Amazon SNS Different From Amazon SQS?

Amazon SQS and Amazon SNS are both messaging services within AWS.

	Amazon SNS	Amazon SQS
Message persistence	No	Yes
Delivery mechanism	Push (Passive)	Poll (Active)
Producer/consumer	Publish/subscribe	Send/receive

Serverless Computing

- **Serverless computing** is a cloud-computing execution model in which the cloud provider acts as the server, dynamically managing the allocation of machine resources. Pricing is based on the actual amount of resources consumed by an application, rather than on pre-purchased units of capacity. It is a form of utility computing.
- The name "serverless computing" is used because the server management and capacity planning decisions are completely hidden from the developer or operator.

AWS Lambda

- AWS Lambda is a compute service that lets you run code without provisioning or managing servers. Lambda executes your code only when needed and scales automatically, from a few requests per day to thousands per second. You pay only for the compute time you consume - there is no charge when your code is not running.
- AWS Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, code monitoring and logging. All you need to do is supply your code in one of the languages that AWS Lambda supports (currently Node.js, Java, C# and Python).
- You can use AWS Lambda to run your code in response to events, such as changes to data in an Amazon S3 bucket or an Amazon DynamoDB table; to run your code in response to HTTP requests using Amazon API Gateway; or invoke your code using API calls made using AWS SDKs. With these capabilities, you can use Lambda to easily build data processing triggers for AWS services like Amazon S3 and Amazon DynamoDB

Use AWS Lambda To Decouple Your Infrastructure

AWS Lambda is a great solution for processing data with high availability and a limited cost footprint.

AWS Lambda allows you to further decouple your infrastructure by replacing traditional servers with simple microprocesses.

Serverless Computing With AWS Lambda

AWS Lambda starts code within milliseconds of an event such as:

- An image upload
- In-app activity
- A website click
- Output from a connected device

Consider AWS Lambda if:

- You're using entire instances to run simple functions or processing applications
- You don’t want to worry about HA, scaling, deployment, or management

Triggers For AWS Lambda Functions

Amazon DynamoDB

Amazon S3

Amazon SNS

Amazon Kinesis

Amazon SES

AWS Config

Scheduled Events

Amazon Cognito

AWS SDKs via Amazon API Gateway

Amazon CloudWatch

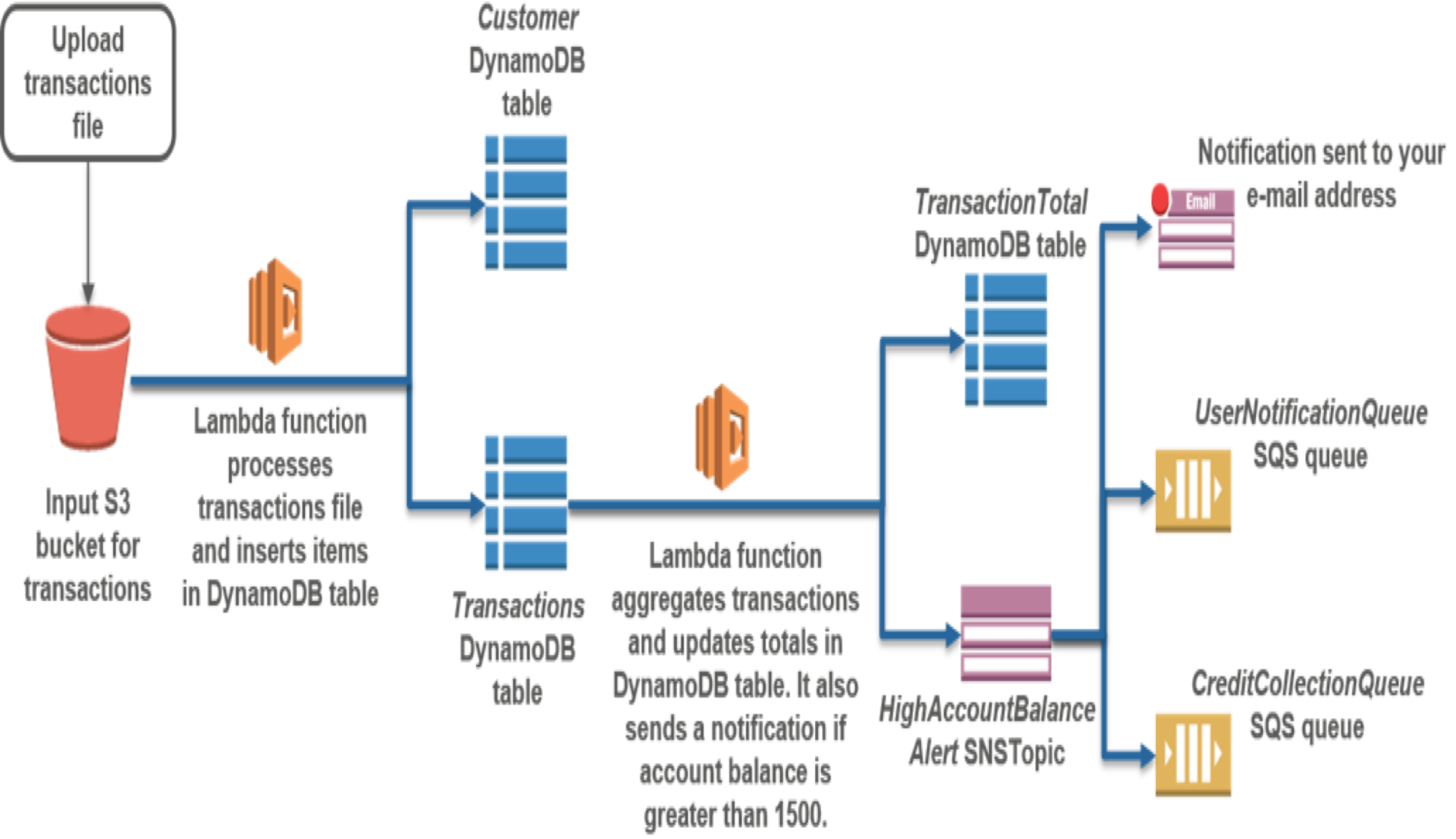
AWS CodeCommit

AWS CloudFormation

Amazon Echo: Alexa Skills

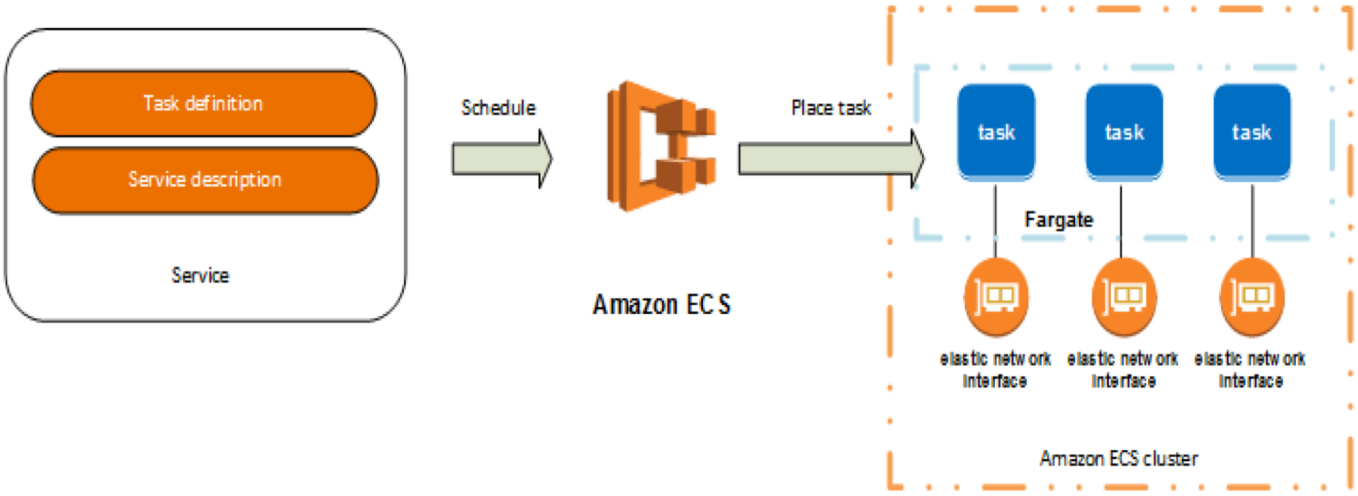
HTTPS via API Gateway

AWS Lambda


Upload transactions file
Input S3 bucket for transactions
Lambda function processes transactions file and inserts items in DynamoDB table
Customer DynamoDB table
Transactions DynamoDB table
Lambda function aggregates transactions and updates totals in DynamoDB table. It also sends a notification if account balance is greater than 1500.
TransactionTotal DynamoDB table
HighAccountBalance Alert SNSTopic
Email
Notification sent to your e-mail address
UserNotificationQueue SQS queue
CreditCollectionQueue SQS queue

Amazon ECS

- Amazon Elastic Container Service (Amazon ECS) is a highly scalable, fast, container management service that makes it easy to run, stop, and manage Docker containers on a cluster. You can host your cluster on a serverless infrastructure that is managed by Amazon ECS by launching your services or tasks using the Fargate launch type. For more control you can host your tasks on a cluster of Amazon Elastic Compute Cloud (Amazon EC2) instances that you manage by using the EC2 launch type.



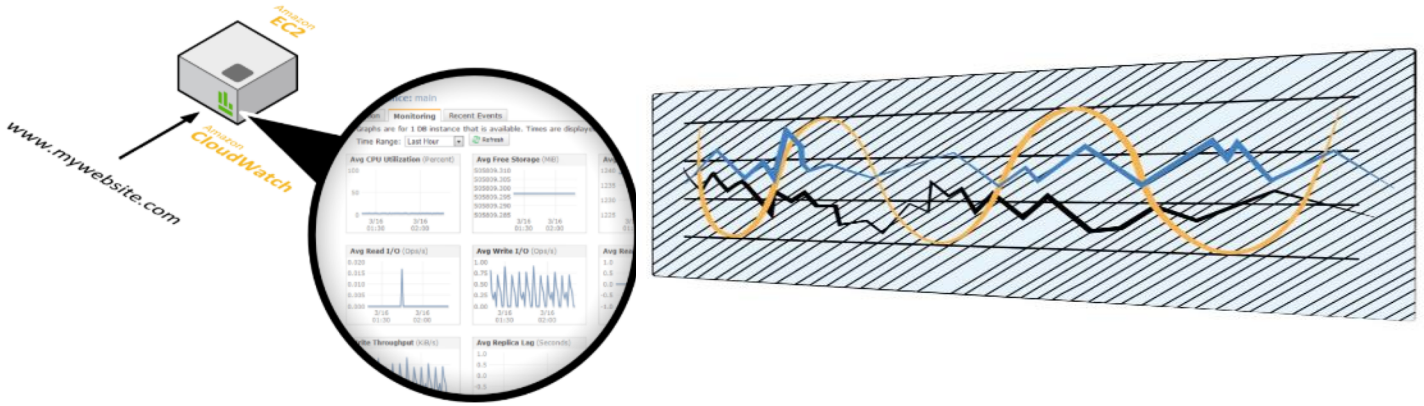
Amazon CloudWatch

Amazon CloudWatch

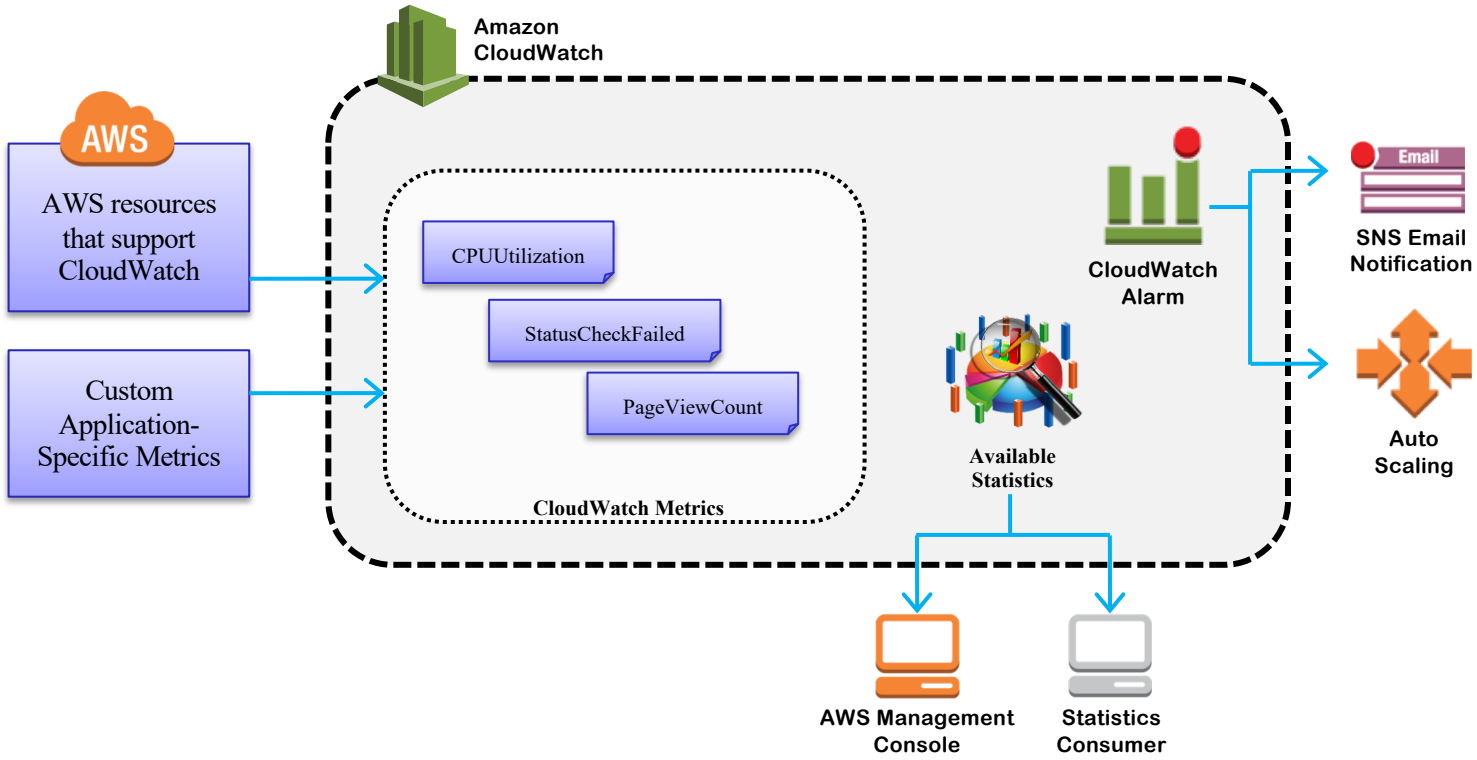
- A **monitoring service** for AWS cloud resources and the applications you run on AWS
- **Visibility into** resource utilization, operational performance, and overall demand patterns
- **Custom application-specific** metrics of your own
- **Accessible** via AWS Management Console, APIs, SDK, or CLI

Amazon CloudWatch Facts

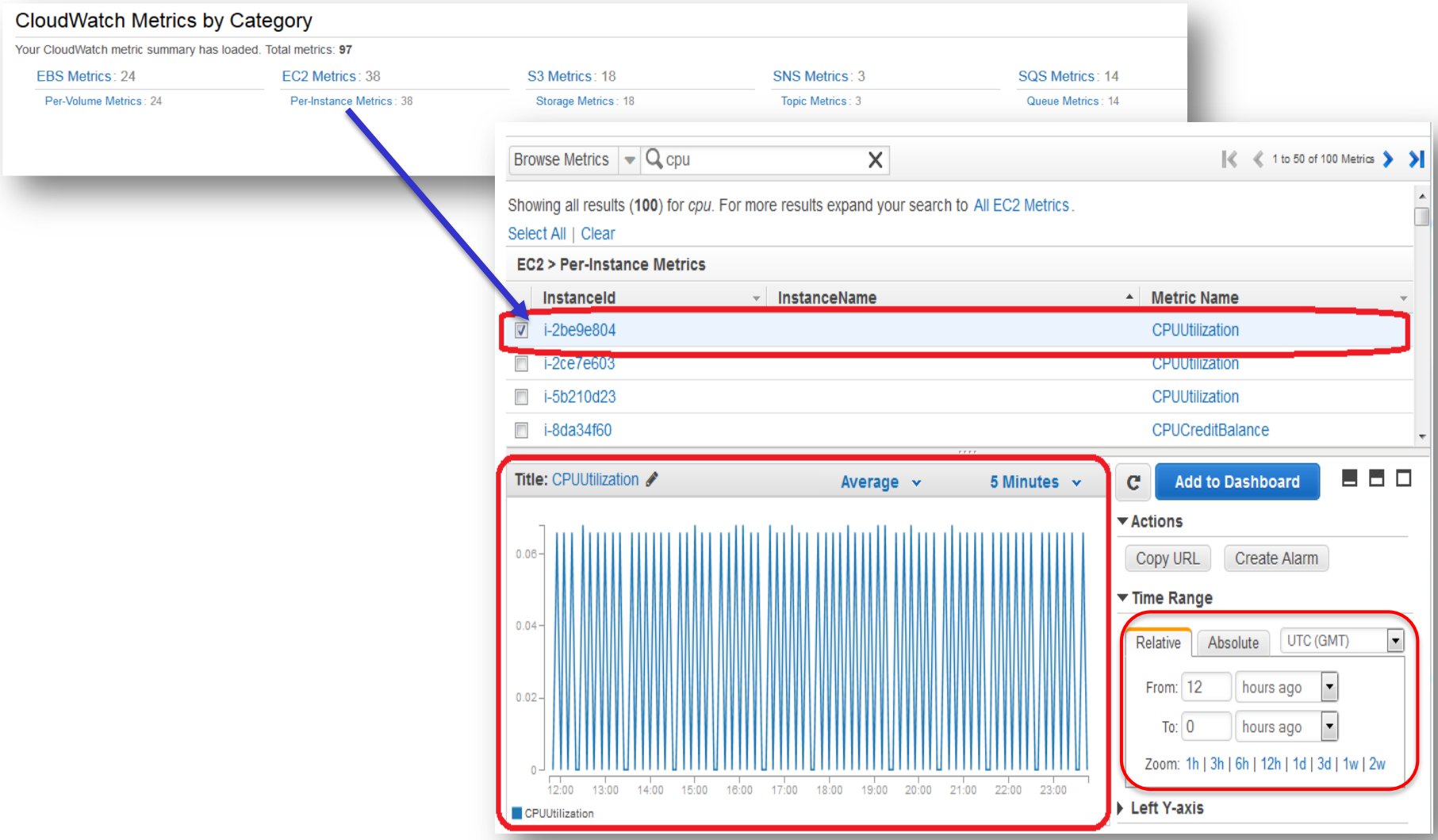
- Monitor other AWS resources
  View graphics and statistics
- Set Alarms



Amazon CloudWatch Architecture



CloudWatch Metrics Examples



CloudWatch Custom Metrics

- You can publish your own metrics to CloudWatch using the AWS CLI or an API. You can view statistical graphs of your published metrics with the AWS Management Console.
- CloudWatch stores data about a metric as a series of data points. Each data point has an associated time stamp.