



Training and
Certification

Systems Operations on AWS
Student Guide
Version 1.1 Update 1

100-SYS-11-EN-SG



Copyright © 2013, 2014 Amazon Web Services, Inc. and its affiliates.
All rights reserved.

This work may not be reproduced or redistributed, in whole or in part,
without prior written permission from Amazon Web Services, Inc.

Commercial copying, lending, or selling is prohibited.

Corrections or feedback on the course, please email us at:
aws-course-feedback@amazon.com.

For all other questions, please email us at aws-training-info@amazon.com.



Table of Contents

Course Overview	1
AWS Platform.....	11
Amazon Virtual Private Cloud (VPC)	40
AWS Identity and Access Management (IAM)	78
Amazon Elastic Compute Cloud (EC2).....	138
Elastic Block Store (EBS)	170
Tagging.....	190
Monitoring	208
Backup and Archive	233
Operations Security	269
Logging.....	306
Elastic Infrastructure.....	311
Cost Control	404
Course Summary	426



Systems Operations on AWS

Course Overview

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Course Overview | Introduction

What We Will Cover

The Systems Operations on AWS course is designed to:

- Enable you to operate highly available and scalable infrastructure on the AWS platform.
- Demonstrate how to effectively manage and support AWS resources.
- Cover concepts such as provisioning infrastructure, deploying applications, tracking costs, monitoring utilization, and creating backups.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The Systems Operations on AWS course is designed to help individuals operate highly available and scalable infrastructure on the AWS platform. In this course, we demonstrate how to effectively manage and support AWS resources. We will cover concepts such as provisioning infrastructure, deploying applications, tracking costs, monitoring utilization, and creating backups.

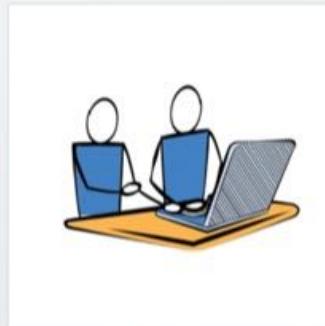
Systems Operations on AWS

Course Overview | Introduction

Course Components

- Presentations
- Hands-on practice labs
- Group discussion

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Systems Operations on AWS has a focus on:

- Presentations
- Hands-on practice labs utilizing command line interface and AWS resources and services.
- Group discussion



The screenshot shows the AWS Systems Operations on AWS Practice Labs interface. At the top, there's a navigation bar with the AWS logo, a search bar, and links for 'Welcome', 'Create', and 'Sign Out'. Below the navigation is a header for 'Systems Operations on AWS - Lab 09 - Elastic Infrastructure - Linux'. The main content area displays a preview of the lab, showing its title, creation date ('Created: Fri, Jun 28, 2013 09:30'), and type ('Lab Type: student'). There are buttons for 'Home', 'All Labs', 'Create', 'Edit', 'Publish', and 'Delete'. At the bottom, there's a footer with the text '© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.'

Systems Operations on AWS

Course Overview | Introduction

Practice Labs

- Linux or Windows
- Practice working with AWS in a variety of pre-designed scenarios.

AWS labs allow you to practice working with AWS at your own pace in a variety of pre-designed scenarios.

Systems Operations on AWS

Course Overview | Introduction

Foundational Knowledge

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



The slide is titled "Foundational Knowledge" and contains three bullet points:

- Working knowledge of systems administration
- Familiarity with cloud computing concepts
- Linux or Windows command-line experience

We recommend that you have foundational knowledge in the following areas.

- Working knowledge of systems administration
- Familiarity with cloud computing concepts
- Linux or Windows command-line experience

Systems Operations on AWS

Course Overview | Introduction

Course Objectives

By the end of this course, you will be able to:

- Use Amazon EC2 features to provision, monitor, scale and distribute compute infrastructure.
- Create Amazon Virtual Private Cloud (VPC) resources such as subnets, network access control lists, and security groups.
- Backup AWS and on-premise resources using AWS services.
- Use Amazon CloudWatch metrics to monitor the health and utilization of AWS resources.
- Leverage resource tagging to allocate costs and optimize resource planning.
- Create a gold image and employ auto scaling into the VPC.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Learning Objectives

At the end of this course, you will be able to:

- Use Amazon EC2 features to provision, monitor, scale and distribute compute infrastructure
- Create Amazon Virtual Private Cloud (VPC) resources such as subnets, network access control lists, and security groups
- Backup AWS and on-premise resources using AWS services
- Use Amazon CloudWatch metrics to monitor the health and utilization of AWS resources
- Leverage resource tagging to allocate costs and optimize resource planning
- Create a gold image and employ auto scaling into the VPC.

Systems Operations on AWS

Course Overview | Introduction

Areas of Responsibility

- Provisioning Infrastructure
- Monitoring Utilization
- Tracking Costs
- Creating Backups



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

This course is designed around 4 key areas of responsibility for IT Systems Operations:

Provisioning Infrastructure: Facilitate rapid deployment of, and update components of infrastructure or application.

Monitoring Utilization : Identify when and where to add capacity as well as understand the performance parameters of their applications, and their related AWS resources and services.

Track Costs: Collect and analyze metrics and resource tagging to allocate costs and analyze resource planning.

Creating Backups: Making copies of data which may be used to restore the original after a data loss event. The primary purpose is to recover data after its loss, be it by data deletion or corruption. The secondary purpose of backups is to recover data from an earlier time.

This course is intended for:

Systems Administrators, Operations Managers, and individuals responsible for supporting operations on the AWS platform

Systems Administrator responsibilities

- Create user roles for access and identity management
- Create a virtual private cloud environment
- Launch an instance
- Add tags to instances for monitoring and tracking
- Create backups to data
- Implement security measures
- Log data
- Analyze billing and costs
- Employ Auto Scaling, monitoring and messaging

Systems Operations on AWS

[Course Overview](#) | [Course Schedule](#)



Day One	
<ul style="list-style-type: none">• AWS Platform• Virtual Private Cloud (VPC)• AWS Identity & Access Management (IAM)• Amazon Elastic Compute Cloud (EC2)• Amazon Elastic Block Store (EBS)	<p>Demo: AWS Management Console Lab: VPC Demo: IAM Lab: EC2 Lab: EBS</p>
Day Two	
<ul style="list-style-type: none">• Tagging• Monitoring• Backup• Operations Security	<p>Lab: Tagging Lab: Monitoring Lab: Backup Lab: Event Collection</p>
Day Three	
<ul style="list-style-type: none">• Logging• Elastic Infrastructure• Cost Control and Optimization• Bonus Activity	<p>Lab: Logging Lab: Creating Elastic Infrastructure Activity: Cost Billing Report Analysis Discussion</p>

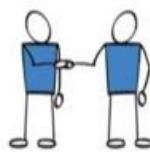
© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Course schedule

Systems Operations on AWS

Course Overview | Introduction

Classroom Introductions



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Classroom Introductions



Module 1: The AWS Platform

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Overview of AWS Platform| Module Objectives

What We Will Cover

- AWS Platform
- AWS Global Infrastructure
- AWS Foundational Services
- Interacting with the AWS Platform



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

This course module introduces the AWS platform and global infrastructure, provides a basic overview of the AWS core products and foundational services, and discusses the ways of accessing AWS services.

Module Objectives

- AWS Platform
- AWS Global Infrastructure
- AWS foundational services with core AWS products
- AWS Management and Administration features of the AWS Platform
- Identify various approaches to interfacing with the AWS Platform

Systems Operations on AWS

Overview of AWS Platform | The AWS Global Infrastructure

AWS keeps business applications running by supporting a secure, redundant, and global infrastructure, which is divided into Regions, Availability Zones, and Edge Locations.

Regions and Edge Locations are current as of March 2013.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Global Infrastructure:

Our data center footprint is global, spanning 5 continents with highly redundant clusters of data centers in each region. Our footprint is expanding continuously as we increase capacity, redundancy and add locations to meet the needs of our customers around the world.

AWS ensures availability of IT infrastructure required to keep business applications running while liberating businesses from maintaining on-premise IT operations. AWS achieves this by supporting a secure, redundant, and global infrastructure, which is divided into Regions, Availability Zones, and Edge Locations to give organizations the option to store data in multiple geographic locations and multiple facilities within a given location. This global infrastructure forms the basis of all other layers of the AWS cloud computing platform.

A **region** is a collection of two or more availability zones in a specific geographic area.

An **availability zone** is an isolated collection of AWS resources. Availability Zones within a region are connected through low-latency links. Remember that, although Availability Zones are relatively close to each other geographically, they are still kept isolated from each other.

Unlike availability zones, **regions** are completely independent from each other.

Systems Operations on AWS

Overview of AWS Platform | The AWS Global Infrastructure

Edge Locations

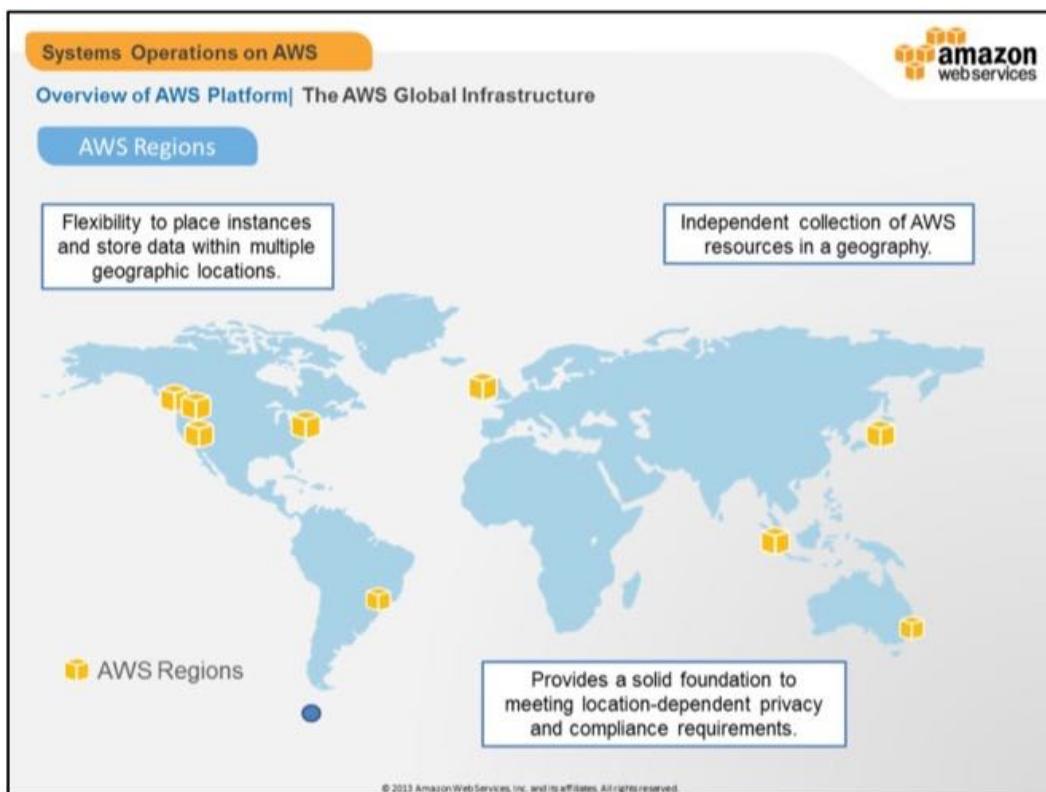
- Host content delivery network
- Used to deliver entire web sites, and dynamic, static, and streaming content
- Requests are automatically routed to the nearest edge location
- Content delivery with the best possible performance



• AWS Edge Locations

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

An AWS Edge Location hosts a content delivery network that can be used to deliver entire web sites, and dynamic, static, and streaming content. Requests for content are automatically routed to the nearest edge location, so content is delivered with the best possible performance.



An AWS Region is an independent collection of AWS resources in a defined geography. It provides customers the flexibility to place instances and store data within multiple geographic locations. With nine Regions all over the world, the selection of a Region provides a solid foundation to meeting location-dependent privacy and compliance requirements.

- Regions – choose to be close to customers and data
- AZs for Availability and Reliability

Systems Operations on AWS

Overview of AWS Platform| Regions and Availability

Availability Zones

- Place instances and store data across multiple Availability Zones within a Region
- Automatically store data on multiple devices across multiple facilities within a Region

Region

Availability Zone

Availability Zone

Availability Zone

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Within a given Region, AWS allows customers to place instances and store data across multiple Availability Zones. These services automatically store data on multiple devices across multiple facilities within a Region. AWS highly recommends provisioning your resources across multiple availability zones. If you have more than one server, it costs nothing extra to run them across more than one AZ and doing so will get you added redundancy.* Should a single AZ have a problem, all assets in your second AZ will be unaffected.

It's not important that you can list each AWS region. It's more valuable to know what regions are available so you can deploy your systems in locations that provide the greatest benefit to your customers.

We list GovCloud here for completeness. Most of you likely will not use GovCloud. If you think GovCloud is the right region for you, fill out the form on this site:
<https://aws.amazon.com/govcloud-us/contact/>.

* Unless the EC2 instances are communicating, there can be bandwidth fees associated with inter-AZ bandwidth.

Systems Operations on AWS

Overview of AWS Platform| Regions and Availability

Most AWS services are deployed on a per region basis except:

Service	Region								
	N. Virginia	Oregon	N. California	Ireland	Singapore	Tokyo	Sydney	São Paulo	GovCloud
VM Import/Export	✓	✓	✓	✓	✓	✓	✓	✓	✓
AWS CloudFormation	✓	✓	✓	✓	✓	✓	✓	✓	✓
AWS Elastic Beanstalk	✓	✓	✓	✓	✓	✓	✓	✓	✓
AWS Storage Gateway	✓	✓	✓	✓	✓	✓	✓	✓	✓
Amazon Simple Notification Services	✓	✓	✓	✓	✓	✓	✓	✓	✓
Amazon DynamoDB	✓	✓	✓	✓	✓	✓	✓	✓	✓
Amazon ElastiCache	✓	✓	✓	✓	✓	✓	✓	✓	✓
AWS Direct Connect	✓		✓	✓	✓	✓	✓	✓	✓
Amazon Elastic Transcoder	✓	✓	✓	✓	✓	✓	✓		
Amazon CloudSearch	✓	✓	✓	✓	✓				
AWS Import/Export	✓	✓	✓	✓	✓				
Amazon Glacier	✓	✓	✓	✓		✓			
High Performance Computing	✓	✓		✓					✓
Amazon Redshift	✓	✓		✓					
AWS CloudHSM									
Amazon Simple Email Service	✓								
AWS Data Pipeline	✓								

Link: Complete list of services available in each region:
<http://aws.amazon.com/about-aws/globalinfrastructure/regional-product-services/>

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

AWS strives to ensure that all services are available in all regions. However, the availability of some services varies depending on the region you select.

Notice that the North Virginia region currently has all services available. Typically (but not always), AWS launches new services in this region first.

To view a complete list of what services are available in each region, find the link on the AWS website:

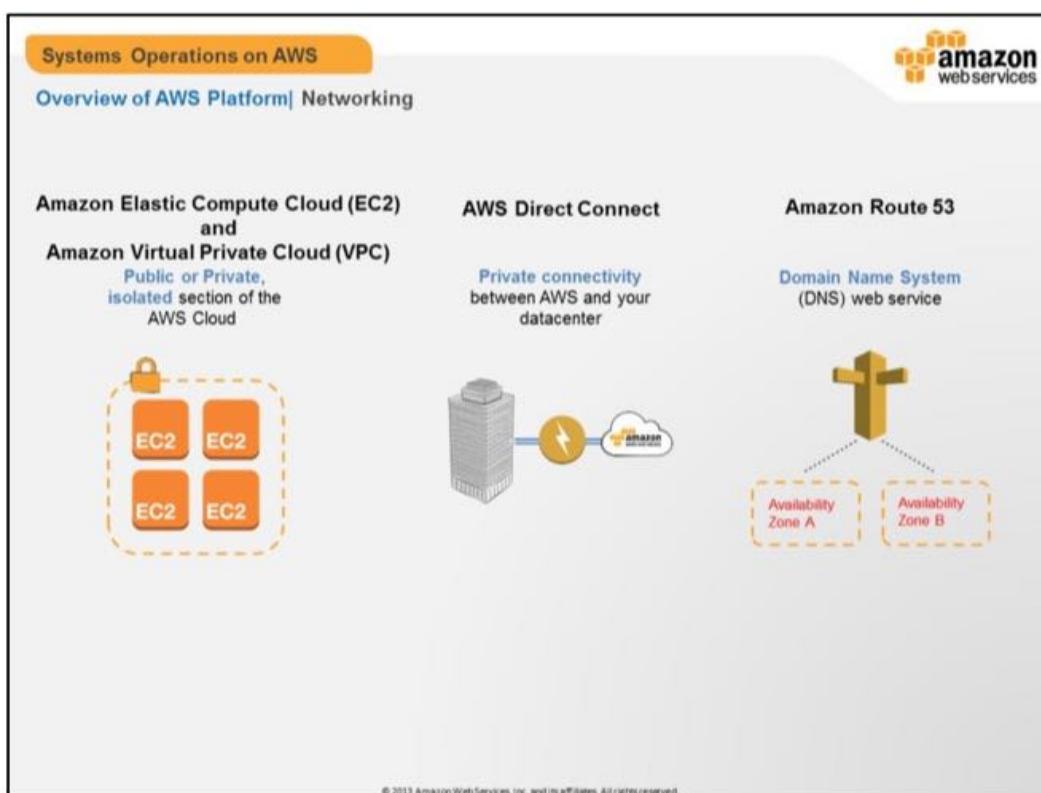
<http://aws.amazon.com/about-aws/globalinfrastructure/regional-product-services/>.



AWS provides a highly reliable, scalable, low-cost infrastructure platform in the cloud that powers hundreds of thousands of businesses all over the world.

The AWS platform is available throughout the world.

This is a simple view of the set of services that we offer. At the core is the compute, storage and data services that are the heart of our offering. We then surround these offerings with a range of supporting components like management tools, networking services and application augmentation services. All this is hosted within our global data center footprint that allows you to consume services without having to build out facilities or equipment.



Amazon Virtual Private Cloud (Amazon VPC) lets you provision a private, isolated section of the Amazon Web Services (AWS) Cloud where you can launch AWS resources in a virtual network that you define.

- **Private network:** Create an Amazon Virtual Private Cloud on AWS's scalable infrastructure, and specify its private IP address range from any range you choose.
- **Subnets:** Divide your Amazon VPC's private IP address range into one or more public or private subnets to facilitate running applications and services in your VPC.
- **ACL's:** Control inbound and outbound access to and from individual subnets using network access control lists.
- **Multiple IP's:** Attach an Amazon Elastic IP address to any instance in your VPC so it can be reached directly from the Internet.
- **VPN connectivity:** Bridge your Amazon VPC and your onsite IT infrastructure with an encrypted VPN connection, extending your existing security and management policies to your Amazon VPC instances as if they were running within your infrastructure.

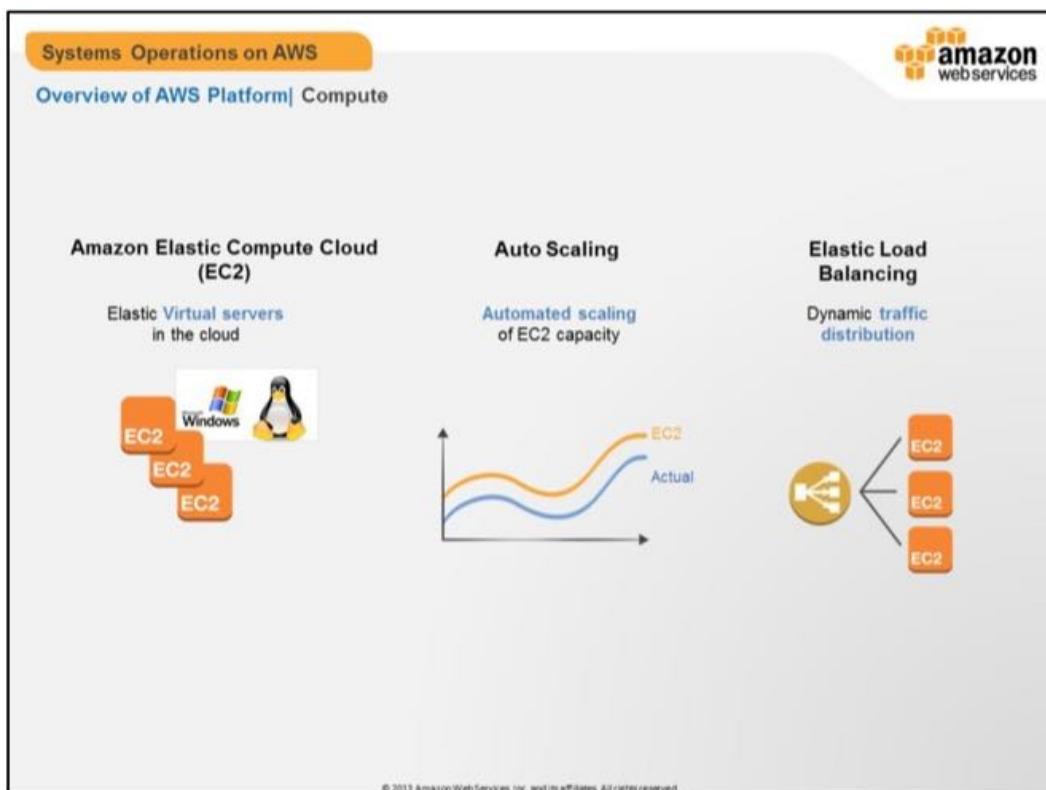
AWS Direct Connect links your internal network to an AWS Direct Connect location over a standard 1 gigabit or 10 gigabit Ethernet fiber-optic cable.

One end of the cable is connected to your router, the other to an AWS Direct Connect router.

- Dedicated, private network connection
- Can reduce costs
- Increased bandwidth
- More consistent

Amazon Route 53 is a highly available and scalable DNS service. Routes end users to Internet applications by translating human readable names like `www.example.com` into the numeric IP addresses like `192.0.2.1` that computers use to connect to each other.

- Highly available
- Scalable
- Integrates with other services well, but can be used as a stand alone service.



AWS offers several services that address the computational needs of users operating in the cloud. The three services that we will discuss are Amazon EC2, Auto Scaling and Elastic Load Balancing

Amazon EC2: a web service that provides resizable compute capacity in the cloud. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. It reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change.

- **Building block** for computing needs.
- **AMI's:** An Amazon Machine Image (AMI) used to initiate one or more instances.
- **Several Instance classes and types:** Instances are deployed in EC2 Public or Virtual Private Cloud in an Availability Zone within a region. We will discuss instances more in depth later in this course.

Auto Scaling: Auto Scaling allows you to scale your Amazon EC2 capacity automatically up or down according to conditions you define. With Auto Scaling, you can ensure that the number of Amazon EC2 instances you're using increases seamlessly during demand spikes to maintain performance, and decreases

automatically during demand lulls to minimize costs. Auto Scaling is particularly well suited for applications that experience hourly, daily, or weekly variability in usage.

- Integrates with Cloudwatch to Minimize cost while scaling with load.
- Use this with spot instances or HPC clusters.
- Free.

Elastic Load Balancing:

- **Managed load balancers:** Elastic Load Balancing automatically distributes incoming application traffic across multiple Amazon EC2 instances. It enables you to achieve even greater fault tolerance in your applications, seamlessly providing the amount of load balancing capacity needed in response to incoming application traffic.
- **Health checks:** Elastic Load Balancing detects unhealthy instances within a pool and automatically reroutes traffic to healthy instances until the unhealthy instances have been restored.
- **Route 53 Failover to other regions:** You can configure [Amazon Route 53](#) to perform DNS failover for your load balancer endpoints. If the load balancer or the application instances registered with the load balancer become unavailable, Route 53 will direct traffic to another load balancer or destination.



AWS offers several services that address the storage needs of users operating in the cloud. The four services that we will discuss are Amazon Elastic Block Store, Amazon Simple Storage Service, Amazon Glacier and AWS Storage Gateway.

Amazon Elastic Block Store (Amazon EBS) provides block level storage volumes for use with Amazon EC2 instances. Amazon EBS volumes are highly available and reliable storage volumes that can be attached to any running instance in the same Availability Zone.

- **Block level storage:** Suited for applications that require a database, file system, or access to raw block level storage.
- Network attached and persistent: The Amazon EBS volumes attached to an Amazon EBS instance are exposed as storage volumes that persist independently from the life of the instance. When not attached to an EC2 instance, you pay only for the cost of storage.
- **1GB to 1TB**
- Can specify the **number of IOPs** you'd like
- **Easy backups:** EBS volumes can be saved via "Snapshots".

Amazon Simple Storage Service (S3): Provides a simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web.

- **Object storage from 1 byte to 5TB:** Amazon S3 stores data as objects within buckets. An object is comprised of a file, an optionally any metadata that describes that file. To store an object in Amazon S3, you upload the file you want to store to a bucket. The number of objects you can store is unlimited.
- **Accessible from anywhere on the internet:** It gives any developer access to the same highly scalable, reliable, secure, fast, inexpensive infrastructure that Amazon uses to run its own global network of web sites.
- **Eleven 9's of durability:** Designed for 99.99999999% durability and 99.99% availability of objects over a given year.

Glacier: Allows you to offload the administrative burdens of operating and scaling archival storage to AWS, and makes retaining data for long periods, whether measured in years or decades, especially simple.

- **Lifecycle directly from S3:** Lifecycle management defines how Amazon S3 manages objects during their lifetime.
- Some objects that you store in an Amazon S3 bucket might have a well-defined lifecycle. Some documents are frequently accessed for a limited period of time. After that, you might not need real-time access to these objects, but your organization might require you to archive them for a longer period and then optionally delete them later. For such objects, you can define rules that identify the affected objects, a timeline, and specific actions you want Amazon S3 to perform on the objects. The lifecycle configuration enables a one-way transition to Glacier.
- **\$0.01 per GB per month:** Amazon Glacier is optimized for data that is infrequently accessed and for which retrieval times of several hours are suitable. With Amazon Glacier, customers can reliably store large or small amounts of data for as little as \$0.01 per gigabyte per month.
- **Automatically encrypted AES-256:** Amazon Glacier supports secure transfer of your data over Secure Sockets Layer (SSL) and automatically stores data encrypted at rest using Advanced Encryption Standard (AES) 256, a secure symmetric-key encryption standard using 256-bit encryption keys.

AWS Storage Gateway: A service architecture that enables integration between your organization's on-premises IT environment and AWS's storage infrastructure.

- **On-premise storage appliance for cloud storage:** Two options. 1) Utilize Amazon S3 for your primary data, while retaining some portion of it locally in a cache for frequently accessed data, or 2) store your primary data locally, while asynchronously backing up that data to AWS.
- Great for DR, or reducing local storage needs with cached volumes

Systems Operations on AWS

Overview of AWS Platform | Database

Amazon RDS

Managed relational database service



- Managed DB
- Automatic backups, patches
- Multi-AZ

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

AWS provides a number of database alternatives for developers. You can run fully managed relational and NoSQL services or you can operate your own database in the cloud on Amazon EC2 and Amazon EBS. This course touches on RDS, but will not focus on the other database services.

Amazon RDS: A web service that makes it easy to set up, operate, and scale a relational database in the cloud.

- **Managed DB:** It provides cost-efficient and resizable capacity while managing time-consuming database administration tasks and gives you access to the capabilities of a familiar MySQL, Oracle, or SQL Server database engine.
- **Automatic backups, patches:** Automatically patches the database software and backs up your database, storing the backups for a user-defined retention period and enabling point-in-time recovery.
- **Multi-AZ:** Gain enhanced database availability, protect your latest database updates against unplanned outages, and scale beyond the capacity constraints of a single DB Instance for read-heavy database workloads.

Systems Operations on AWS

Overview of AWS Platform Application Services

Amazon SNS

Delivering messages to HTTP or e-mail endpoints

The diagram illustrates the Amazon SNS service. On the left, there is a purple cloud-like icon labeled "Amazon SNS". A blue arrow points from this icon down to a purple rectangular box labeled "Email". Inside the "Email" box, there is a red circular icon with a white dot and a smaller purple rectangular icon below it.

- Delivering messages to HTTP or e-mail endpoints
- Set up, operate, and send notifications
- Publish messages from an application and immediately deliver them to subscribers or other applications

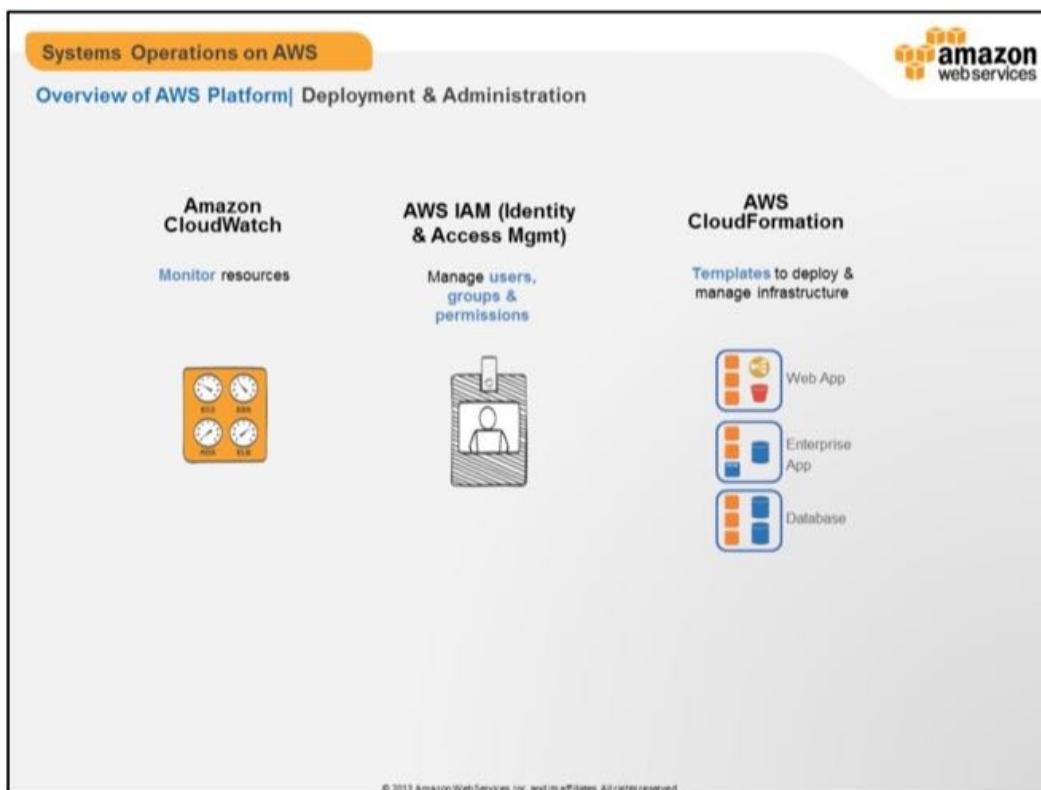
© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Messaging is a very important concept when developing applications that scale well in the cloud. For applications to scale effectively – especially in the horizontal direction – they should be decoupled (i.e., broken into their simplest components). Messages are used by an application's decoupled components to communicate things like state, tasks, etc.

Though AWS has several messaging and application services, in this training we will look at Amazon Simple Notification Services (SNS)

Amazon Simple Notification Service (SNS): Amazon Simple Notification Service is a web service that makes it easy to set up, operate, and send notifications from the cloud.

- Delivering messages to HTTP or e-mail endpoints
- Set up, operate, and send notifications
- Publish messages from an application and immediately deliver them to subscribers or other applications



CloudWatch:

- **Detailed monitoring for all AWS services:** For Amazon EC2 instances, Amazon CloudWatch **Basic Monitoring** collects and reports metrics for CPU utilization, data transfer, and disk usage activity from each Amazon EC2 instance at a five-minute frequency. Amazon CloudWatch **Detailed Monitoring** provides these same metrics at one-minute intervals, and also enables data aggregation by Amazon EC2 AMI ID and instance type.
- No cost for **5 minute** metrics
- As **granular as 1 minute**
- Custom metrics to monitor anything: View **graphs** and **statistics** for any of your metrics, and get a quick overview of all your alarms and monitored AWS resources in one location on the Amazon CloudWatch dashboard.
- Alarms, and **integration** with tools like Auto Scaling

AWS Identity & Access Management (IAM): A web service that enables AWS customers to manage users and user permissions in AWS. With IAM, you can centrally manage users, security credentials such as access keys, and permissions that control which AWS resources users can access.

- **Web Identity Federation:** Manage federated users and their permissions
 - You can enable identity federation to allow existing identities (for

example, users) in your enterprise to access the AWS Management Console, to call AWS APIs, and to access resources, without the need to create an IAM user for each identity.

- **Resource level permissions:** You can create roles in IAM, and manage permissions to control which operations can be performed by the entity, or AWS service, that assumes the role. You can also define which entity is allowed to assume the role.
- **Multi-Factor Authentication**

CloudFormation:

Deploy AWS infrastructure with a template: AWS CloudFormation gives developers and systems administrators an easy way to create and manage a collection of related AWS resources, provisioning and updating them in an orderly and predictable fashion.

- A consistent scripting interface for **creating/configuring** AWS resources.
- A framework for **lifecycle management** of resources created using scripts.
- You can deploy and update a template and its associated collection of resources (called a stack) via the AWS Management Console, CloudFormation command line tools or APIs.
- Free

Systems Operations on AWS

Overview of AWS Platform | Scope of AWS Resources

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Scope	AWS Resource
Global	<ul style="list-style-type: none">AWS IAM Users, Groups, and RolesAmazon Route 53 Hosted Zones and Record SetsAmazon CloudFront Distributions
Region	<ul style="list-style-type: none">Amazon S3 BucketsAmazon Machine Images (AMIs)Amazon CloudWatch MetricsEBS SnapshotsAmazon Virtual Private Cloud (VPC)
Availability Zone	<ul style="list-style-type: none">Amazon EC2 InstancesEBS VolumesRDS Database InstancesAmazon ElastiCache ClustersSubnet

When you create AWS resources, those resources exist at a scope which varies depending on the service – either Global, Region, or Availability Zone. Resources with Global and Regional scope are automatically distributed across multiple Availability Zones by AWS. Some examples of resources at each scope include:

- AWS Identity and Access Management (IAM) users, groups and roles
- Amazon CloudFront distributions
- Amazon Route 53 hosted zones and record sets

Some services provide cross-region copy commands:

- S3 Buckets
- Amazon Machine Images (AMIs)
- CloudWatch Metrics
- EBS Snapshots
- Virtual Private Cloud (VPC)

Availability Zones

- EC2 Instances
- EBS Volumes

- **RDS Database Instances**
- **Amazon ElastiCache Clusters**
- **Subnet**



Interacting with the AWS Platform

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Let's review different ways to interface with AWS services

- AWS Management Console
- Command Line Interface
- Software Development Kits

Systems Operations on AWS

Overview of AWS Platform | Interacting with the AWS Platform

The AWS Management Console, Command Line Interface (CLI) and Software Development Kits (SDKs) address the underlying REST API. Ultimately the REST API is the direct interface.

The diagram illustrates the interaction between three components:

- AWS Management Console:** A screenshot of the AWS Management Console interface, showing various service links like CloudWatch, Lambda, and S3.
- AWS CLI:** A terminal window showing two commands: "aws ec2 create-volume --availability-zone [YourAZ] --size 100" and "aws ec2 create-volume --availability-zone [YourAZ] --size 100".
- REST API:** A central icon depicting two people at a table, symbolizing communication or interaction.

Arrows point from both the Management Console and the CLI interface towards the central REST API icon, indicating that both interfaces interact with the REST API directly.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The AWS Management Console, Command Line Interface (CLI) and Software Development Kits (SDKs) address the underlying REST API. Ultimately the REST API is the direct interface.

Systems Operations on AWS

Overview of AWS Platform | Interacting with the AWS Platform

Three Ways of Accessing AWS Services

1 AWS Management Console

2 Command Line Interface (CLI)

3 Software Development Kits (SDKs)

The AWS Management Console: easy-to-use graphical interface to manage your compute, storage, and other cloud resources.

Supports majority of functionality for each service and most AWS products can be used from within the console.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

There are three methods of accessing and using AWS Services that will discuss today. AWS console, Command line tools, and SDKs for using AWS Services,

AWS Management Console

The AWS Management Console provides an easy-to-use graphical interface to manage your compute, storage, and other cloud resources. It supports majority of functionality for each service and most AWS products can be used from within the console.

Since this course is considered an introduction to Systems Operations on AWS, we will focus on the AWS Management Console as the web-based user interface.

The screenshot shows a slide titled "Three Ways of Accessing AWS Services". At the top left is the "Systems Operations on AWS" logo. To the right is the Amazon Web Services logo. The slide content includes three numbered boxes: 1. AWS Management Console, 2. Command Line Interface (CLI), and 3. Software Development Kits (SDKs). A callout box below the second item provides additional information about the CLI.

Three Ways of Accessing AWS Services

- 1 AWS Management Console
- 2 Command Line Interface (CLI)
- 3 Software Development Kits (SDKs)

The CLI is for administrators and those who are more comfortable typing in commands. For links to command line tools and their documentation for AWS products, visit the web site mentioned on the screen.

<http://aws.amazon.com/cli>

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The Command Line Interface is for administrators and those who are more comfortable typing in commands. For links to command line tools and their documentation for AWS products, visit the web site mentioned on the screen.

The screenshot shows a section titled "Command Line Interface". It contains a paragraph about the CLI, a "Documentation Conventions" box with examples for Linux and Windows, and a note that AWS CLI commands are the same regardless of operating system.

Systems Operations on AWS

Overview of AWS Platform | Interacting with the AWS Platform

Command Line Interface

Command line interface (CLI) tools provide a convenient scripting interface to the AWS API directly at the command prompt. As a convention, command line text is prefixed with a generic PROMPT> command line prompt.

Documentation Conventions

Linux
\$aws ec2 describe-regions

Windows
C:\>aws ec2 describe-regions

AWS CLI commands are the same regardless of operating system

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

A command line interface (or CLI) is a means of interaction with a computer program where the user issues commands to the program in the form of successive lines of text, or, command lines.

Command line interface (CLI) tools provide a convenient scripting interface to the AWS API directly at the command prompt. As a convention, command line text is prefixed with a generic PROMPT> command line prompt.

Documentation Conventions

Linux

\$aws ec2 describe-regions

Windows

C:\>aws ec2 describe-regions

AWS CLI commands are the same regardless of operating system

The slide is titled 'Three Ways of Accessing AWS Services'. It lists three methods: 1. AWS Management Console, 2. Command Line Interface (CLI), and 3. Software Development Kits (SDKs). A callout box points to the SDKs section, stating: 'SDKs are used to simplify accessing AWS services from your preferred development language. For links to the SDKs, visit the web page here.' Below the slide is a link: <http://aws.amazon.com/tools>.

Systems Operations on AWS

Overview of AWS Platform | Interacting with the AWS Platform

Three Ways of Accessing AWS Services

1 AWS Management Console

2 Command Line Interface (CLI)

3 Software Development Kits (SDKs)

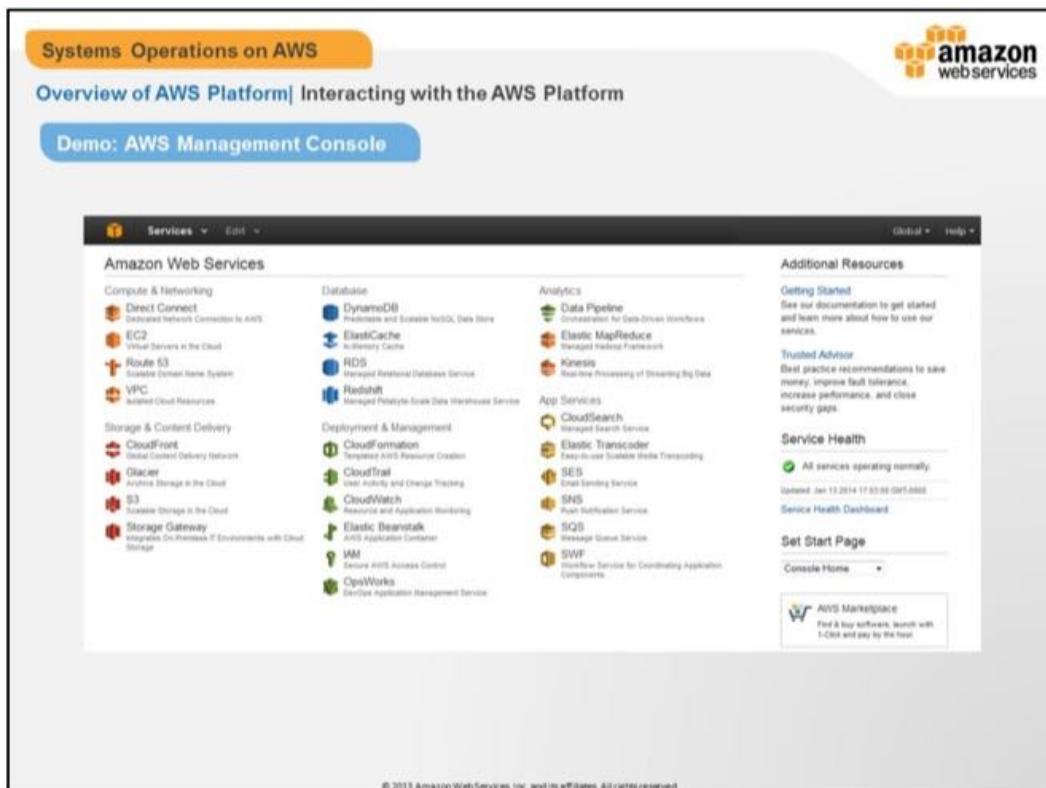
SDKs are used to simplify accessing AWS services from your preferred development language. For links to the SDKs, visit the web page here.

<http://aws.amazon.com/tools>

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

An SDK is used to simplify accessing AWS services from your preferred development language. This is done by providing a wrapper for the underlying REST API.

For links to the SDKs, visit the visit the web page here. aws.amazon.com/tools.



AWS Management Console Overview and Demonstration

- Access the AWS Management Console
- Provide a tour – below are guidelines for an overview of the key features.

After logging into the AWS Management Console you reach the Console Home, which displays a welcome message followed by links to reference material.

Use the Getting Started Guides to access AWS documentation and reference architectures to get guidance and information about best practices to build highly scalable and reliable applications.

Below the links is the Set Start Page drop-down list.

[Click drop-down box]

This list helps you select which screen you would like to see when you login to your AWS account.

[Click any menu item]

The Amazon Web Services section gives you access to most of the AWS products and services. These services are categorized as:

- Compute & Networking
- Storage & Content Delivery
- Database
- Deployment & Management
- Application Services

To use any of the listed products and services, click the corresponding icon. Each service listed has a corresponding service console that can be accessed by clicking on service name. The service-specific console shows you service status and allows you to interact with the service.

Let's take a look at the service console for the Identity and Access Management service.

[Click IAM link]

With the Identity and Access Management Service, or IAM, you can create and manage AWS user credentials to control access to your AWS resources.

[Back to console]

The AWS console does not include a console for each service. When a service is missing from the console, you can still access it using the command line interface or the APIs.

Use the Service Health Dashboard link to view Region-wise real-time status information about AWS service availability. Scrolling down the same page shows the status history as well.

Systems Operations on AWS

Overview of AWS Platform | Summary

Summary

- Regions are geographically isolated collections of AWS Availability Zones.
- Deploying across regions will lower latency.
- Deploy across availability zones to increase availability.
- Compute, Storage, and Database are the core AWS services, while
- AWS Networking services extend enterprise infrastructure into the cloud.

There are three ways of accessing AWS services.

- The AWS Management Console is a point and click web interface to help you access and manage most compute, storage, and other AWS services.
- Command line tools are available to help developers to give commands directly at the command prompt.
- An SDK is used to simplify accessing AWS services from your preferred development language.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

You have reached the end of this section. In this section, you learned that:

Regions are geographically isolated collections of AWS Availability Zones.

Deploying across regions can improve stability and lower latency.

Compute, Storage, and Database are the core AWS services, while AWS Networking services extend enterprise infrastructure into the cloud.

The three ways of accessing AWS services are –

1. AWS Management Console
2. AWS Command Line Interface
3. AWS Software Development Kits, or , SDKs



Module 2: Amazon Virtual Private Cloud (VPC)

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Amazon VPC I Topics

Amazon web services



- What is a Virtual Private Cloud?
 - Creating a VPC
 - Public and private subnets
 - Default VPC
 - Elastic Network Interfaces (ENIs)
- Gateways and Routes
 - Routing and route tables
 - Internet Gateways and NAT
 - Virtual Private Gateways
 - Extending corporate datacenters
- Security
 - Security Groups
 - NACLs
- Using CloudFormation to provision VPCs
- Limits

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Virtual Private Cloud, or VPC, is a unique construct within an AWS region. It allows the provisioning of a discrete, compartmentalized private network that may then be divided up into multiple subnets. This module will focus on:

- What is a VPC?
- Creating a VPC
 - Creating public and private *subnets* within a VPC
 - Default VPC vs. what we will be creating as part of the lab
 - Elastic Network Interfaces (ENIs)
- Routing between subnets using *route tables*
 - Internet Gateways and NAT (Network Address Translation)
 - Virtual Private Gateways
 - Extending corporate datacenters via VPN (Virtual Private Network) connections
- Securing VPCs with *security groups* and *network access control lists*

- Limitations of VPC
- Using CloudFormation to provision VPCs



What is a VPC?

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Amazon VPC | What is a VPC?

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

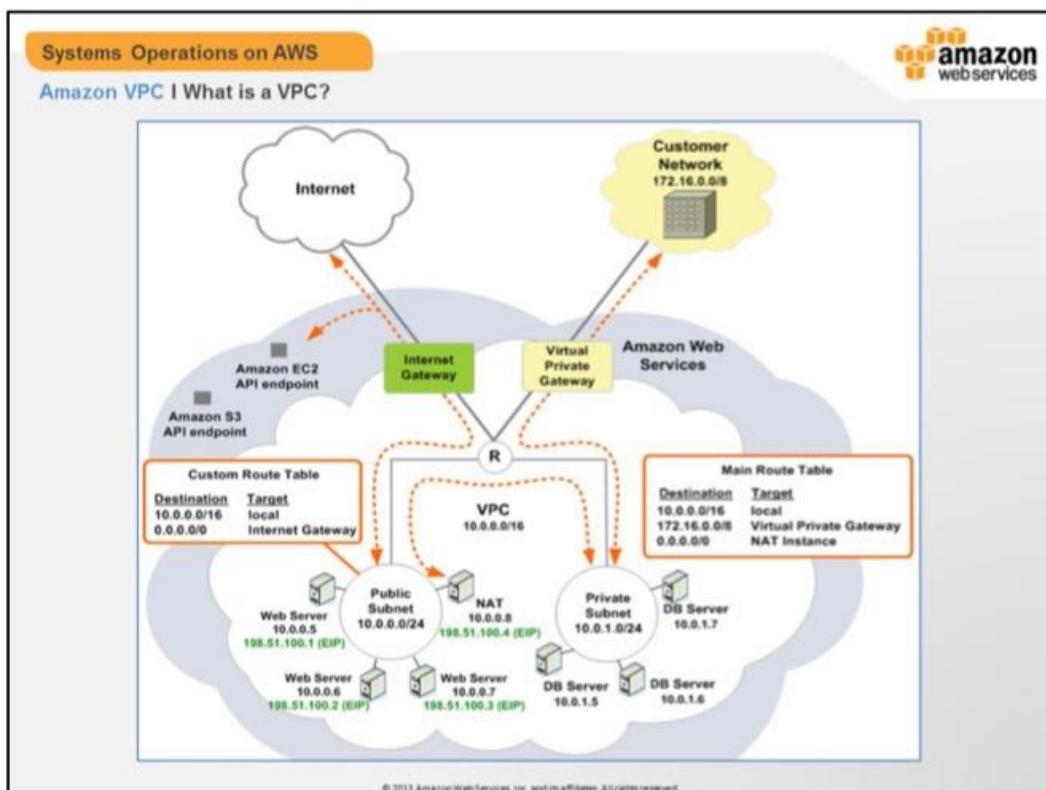
- Virtual network; isolated portion of AWS cloud for EC2 instances
- Optional *dedicated* tenancy
- Supports logical separation with subnets
- Fine grained security
 - Security Groups
 - Network ACLs
- Examples use-cases:
 - Typical 3-tier architectures (web, app, db)
 - Extending on-premise corporate datacenters

Virtual Private Clouds (VPCs) are simply isolated portions of the AWS cloud that a customer may provision for instantiating their AWS infrastructure in (i.e. EC2 instances, RDS instances etc). They are virtual networks and as such, they support multiple subnets, routing and fine grain security mechanisms. VPCs marked as dedicated, constrain the VPC to only allowing dedicated EC2 instances to be launched, i.e. dedicated instances will not be hosted on droplets that host other customer's instances, i.e. they are single tenancy.

Some common use-cases are:

- Setting up of a typical 3-tier (web, application and database) system where each tier resides in its own subnet and is locked down only to those TCP ports that facilitate communication between the tiers.
- Extending an on-premise corporate datacenter by creating a VPN connection to the VPC. Provisioned AWS resources and on-premise infrastructure may

then communicate with each other as if they were part of the same network. Additionally, implementing a direct connection with AWS will allow the customer to provision a dedicated network connection in turn improving bandwidth and reducing associated network costs.



Here is a nice high-level overview of a typical VPC architecture. Note some of the key-terms and abbreviations we've already touched on in the slides at the start of this module.

This VPC has the following characteristics:

- Two subnets, one public facing (i.e. from the Internet) and the other private
- Web servers sit in the public Internet accessible subnet
- Database servers are sitting in the private (or backend) subnet (not accessible from the Internet)
- Web servers access the Internet through an Internet Gateway (IGW) which we'll discuss shortly
- In-bound access is facilitated by Security Groups, NACLs in conjunction

with Elastic IP addresses (EIPs)

- Database servers access the Internet through a NAT device (we'll also talk about this very shortly too!)
- Connectivity back to the customer's corporate network via a VPN connection (routed via the Virtual Private Gateway)
- Connectivity to the rest of the AWS service end-points such as S3 and so on

Local Router: "R" A virtual component that allows traffic to flow between subnets. Every route table including the main route table comes with a route targeting the local router.

Don't be too concerned if everything is not familiar – we'll go through everything in the course of this slide deck. It's important to note that really VPC is a set of additional control structures that support the launching of EC2 instances.

Systems Operations on AWS

Amazon VPC | What is a VPC?

Abbreviations

Industry-Standard Terms

- CIDR – Classless Inter-Domain Routing
- NACL – Network Access Control List
- NAT – Network Address Translation
- VPN – Virtual Private Network

AWS Terms

- VPC – Virtual Private Cloud
- VGW – Virtual Private Gateway
- EIP – Elastic IP Address
- ELB – Elastic Load Balancer
- ENI – Elastic Network Interface
- IGW – Internet Gateway

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

These abbreviations are used commonly when working with VPCs; get used to seeing them!

The items in black are industry-standard terms, the ones in orange are specific to AWS.

The screenshot shows a slide titled 'VPC Resource Types' under the 'Amazon VPC | What is a VPC?' section of the 'Systems Operations on AWS' guide. The slide features a table with eight rows, each containing a term and its description. The table has a blue header row. The terms listed are: Virtual Private Cloud (VPC), Subnet, Route Table, Security Group (SG), Network Access Control List (NACL), VPN Connection, NAT Instance, and Internet Gateway. The descriptions provide a brief overview of each resource's function within a VPC.

Term	Description
Virtual Private Cloud (VPC)	Dedicated isolated virtual network
Subnet	IP address range within a VPC
Route Table	Determines where traffic is directed
Security Group (SG)	Instance level stateful firewall
Network Access Control List (NACL)	Subnet level stateless firewall
VPN Connection	Connects a VPC to other networks
NAT Instance	Enables Internet connectivity for instances in private subnets
Internet Gateway	Enables Internet connectivity for instances in public instances

Here is a list (not an exhaustive one, but the main ones!) of the VPC resource types we will be talking about in the context of VPC. Feel free to ask students how these may map to physical constructs in the traditional on-premise environment.

The two diagrams of VPC architectures (default and our lab one) are coming up in a couple of slides; you'll see where these key terms fit as part of the bigger picture.



Creating a VPC

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The screenshot shows a slide from the AWS Systems Operations on AWS course. At the top left, there's a yellow header bar with the text "Systems Operations on AWS". Below it, a blue bar contains the title "Amazon VPC I Creating a VPC". Underneath these, a blue tab labeled "Configurations" is selected. The main content area lists several bullet points about VPC configurations:

- Default VPC created for new accounts
- Additional VPCs created via the AWS Management Console
 - Manually
 - VPC Wizard
- Based on a CIDR IP address range
 - Specific to customer's environment
 - Can be any range no larger than /16
 - Leverage corporate IP address ranges
- Can create from CloudFormation template
 - See lab!

At the bottom of the slide, there's a small note: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

A VPC will be created by default for new accounts. However, additional VPC's may be created using the Management Console (either manually or with the VPC Wizard) or via a CloudFormation template. There are several configurations that can be chosen when creating a VPC with the VPC Wizard:

- VPC with a Single Public Subnet Only
- VPC with Public and Private Subnets
- VPC with Public and Private Subnets and Hardware VPN Access
- VPC with a Private Subnet Only and Hardware VPN Access

Before any subnets can be created, a CIDR (Classless Inter-Domain Routing) block specifying the IP address range of the VPC must be chosen; and as the VPC is isolated in the cloud any IP address range may be used.

A customer can decide the CIDR range to use; this could be an extension of their own existing IP address range, the only restriction is that the range *not* be larger (or more wide) than /16.

Systems Operations on AWS

Amazon VPC I Creating a VPC

Creating Subnets in a VPC

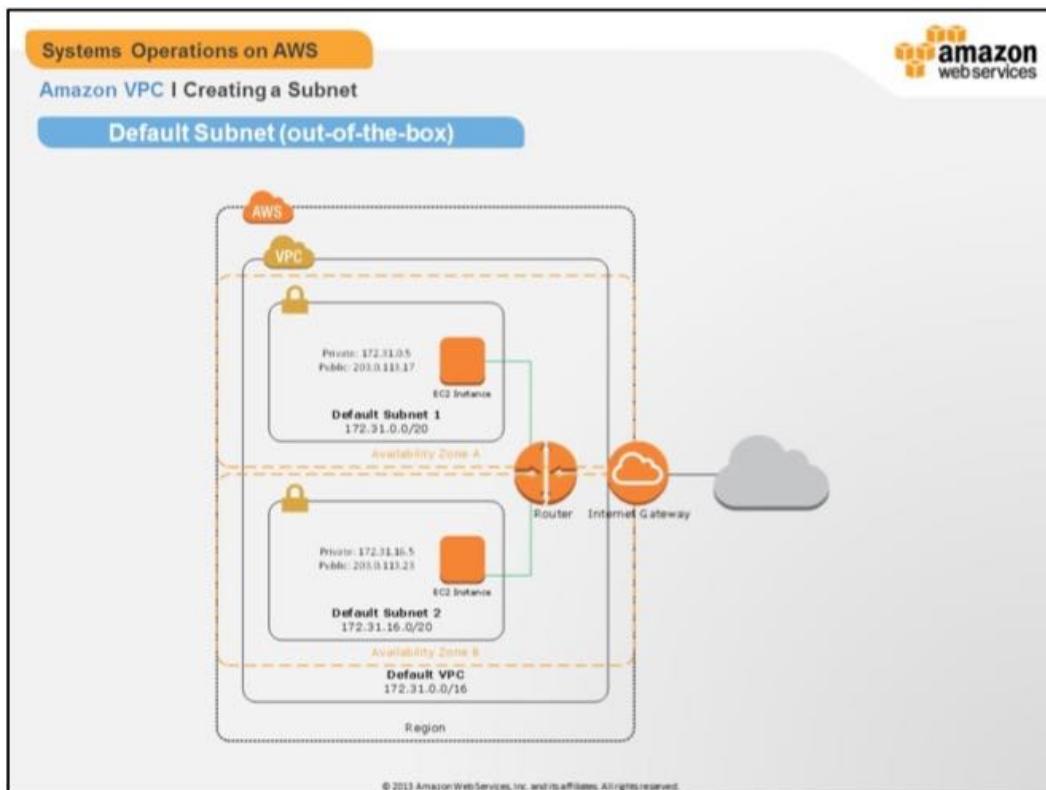
- Public and private subnets
- Create from the AWS Management Console
 - Manually
 - VPC Wizard
- CIDR block to define IP address range
 - Subset of VPC IP address range
- Also can be created via CloudFormation

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

These types of subnets can be created using the VPC Wizard, or alternatively manually using the Management Console (remember, CloudFormation templates are an option for provisioning most kinds of AWS resources including VPCs, subnets etc. CloudFormation will be discussed later). In order to manually create a subnet, regardless of its scope:

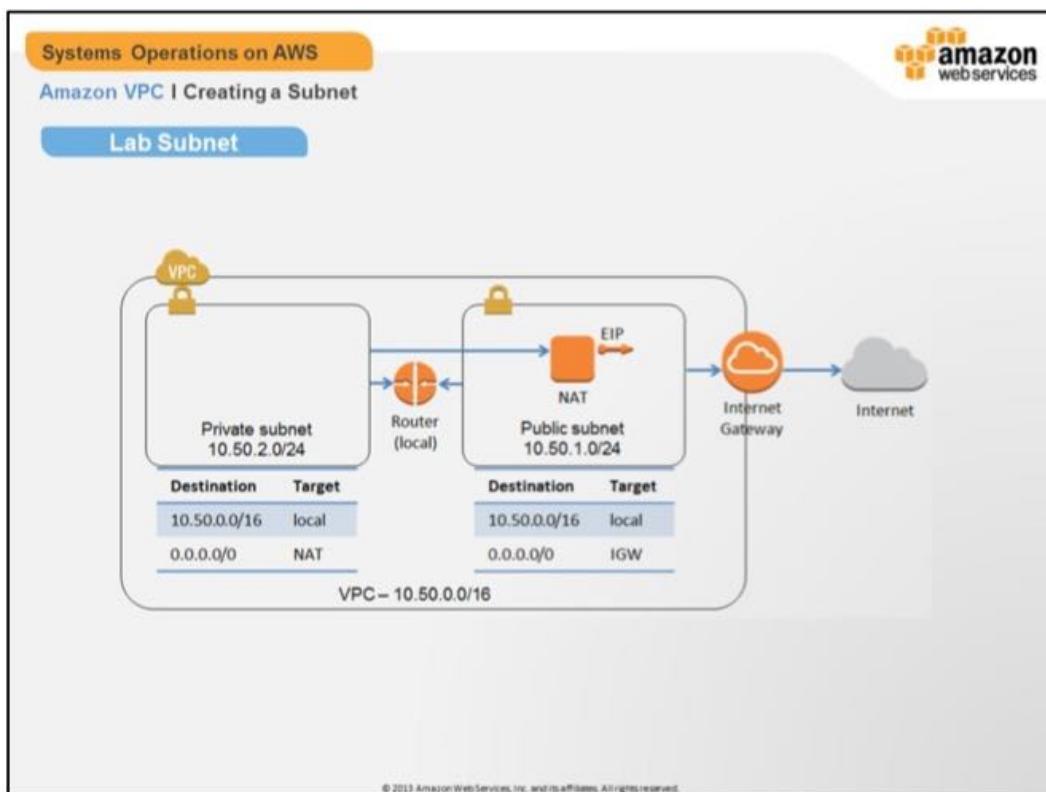
- Select the VPC that it will be part of
- The AZ (Availability Zone) it will reside in
- The CIDR block (or IP address range) of the subnet

The chosen CIDR block of the subnet will be a subset of the VPCs IP address range. Keep in mind it can be problematic to redefine IP address ranges at a later date, so ensure that adequate ranges are selected to allow for the maximum numbers of IP addresses that may be required. Public and private subnets are differentiated by their route tables in place on them.



This is the default subnet that is provisioned for new accounts.

The default VPC consists of multiple subnets with all Internet destined traffic routed through the Internet Gateway (IGW). Note that the instances shown in this diagram are not launched on your behalf; the point we're trying to drive home here is that any EC2 instances launched into a VPC-by-default account will be spun up in one of these subnets.



This is the VPC scaffolding that we'll be creating; you'll create this in this module's lab and subsequent modules will build upon this.

Note the similarities between this and the default VPC just shown. All of the same constructs exist albeit with more subnets and the addition of a NAT EC2 instance. Private subnets are segregated and are not public/external facing; in order for EC2 instances launched in the private subnets to have Internet connectivity traffic from them will need to be routed via a NAT instance. The role of the router is the same, but rather than routing traffic via the Internet Gateway it will be routed to the NAT instance sitting in the public subnet (which in turn will forward route the traffic via the Internet Gateway).

The lab you will do at the end of this module will take you through building a VPC and NAT instance as shown above. By the end of this course you will have a fully functioning application running within the confines of this VPC scaffolding.

Systems Operations on AWS
Amazon VPC I Creating a VPC

Elastic Network Interfaces

Attributes of an ENI:

- Primary IP address
- Optional secondary IPs
- MAC address
- Security groups
- Src/Dest check flag
- Description

The diagram illustrates the AWS VPC architecture. It shows a VPC boundary with two subnets: Default Subnet 1 (172.31.0.0/20) and Default Subnet 2 (172.31.16.0/20). Each subnet contains an EC2 instance connected to an ENI (Elastic Network Interface). A Router is also present, connected to both ENIs. The entire VPC is located within an Availability Zone (AZ), which is part of a Region. The Default VPC (172.31.0.0/16) is also indicated.

ENIs are a VPC networking construct and component of EC2. By default each instance provisioned in a VPC will have an ENI (Elastic Network Interface) attached to it. Additional ENIs can be created and attached to EC2 instances as required; an EC2 instance may have multiple ENIs attached to it (dependent on instance type).

This allows architectures (such as multi-homed) where one interface has a private IP from one subnet and the other interface has a private IP from another (as shown in the diagram above). It also allows network architectures to be deployed via CloudFormation without actually having the EC2 instances running as ENIs can be created independent of instances.

***** NOTE:** ENIs are not like real interfaces where in the traditional sense adding a new NIC would give you another 1gb connection for example. An EC2's "physical" NIC encapsulates the ENIs; the more ENIs on an instance the more chance of performance issues across the physical NIC.

Systems Operations on AWS

Amazon VPC | Creating a VPC

Comparison to EC2 Classic

Characteristic	EC2 Classic	Default VPC	Non-Default VPC
Public IP Addresses	Your instance receives a public IP address.	Instance receives a public IP address by default, unless specified otherwise.	Instance doesn't receive a public IP address by default.
Private IP Addresses	Instance receives a private IP address from the EC2-Classic range each time it's started.	Instance receives a static private IP address from the address range of your default VPC.	Instance receives a static private IP address from the address range of your VPC.
Multiple IP Addresses	Select single private IP address for your instance; multiple IP addresses not supported.	Assign multiple private IP addresses to your instance.	Assign multiple private IP addresses to your instance.
Elastic IP Addresses	EIP is disassociated from your instance when stopped.	EIP remains associated with your instance when stopped.	EIP remains associated with your instance when stopped.

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-instance-addressing.html#differences>

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

So, how does EC2 VPC compare with EC2 classic? Well we've already mentioned that VPC provides an additional set of control structures, thus allowing additional control over the instances deployed within AWS. This table shows just the differences between the two 'modes' of EC2. Essentially EC2 is identical albeit a few minor changes; VPC just adds additional control as well as some other capabilities such as VPN, Direct Connect etc. We'll touch on these later in the presentation.

Depending your region and/or the age of your account, will determine which of these options you can use.

VPC offers everything that EC2-Classic offers, and much more. Therefore, newer accounts and regions don't have access to EC2-Classic anymore. Since it was easier to launch instances and start using AWS with EC2-Classic than it is with manually created VPCs, there are 2 kinds of VPCs:

1. Default VPC, which comes by default with your account in each region that doesn't have EC2-Classic, and
2. Non-Default VPCs, which are the VPCs you create manually

The Default VPC tries to mimic EC2-Classic. The differences are noted in the above table.



Gateways and Routes

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Amazon VPC I Gateways & Routes

Route Tables

- Specifies how traffic is routed between subnets.
- Rules determine what is allowed / denied
 - Also known as routes
 - Routes consist of a *destination* and *target*
- A target can be an:
 - EC2 Instance
 - Elastic Network Interface (ENI)
 - IGW (Internet Gateway)
 - VGW (Virtual Private Gateway)
 - Local router
- Network destination represented as a CIDR range
 - IGW
 - VGW
- Main route table (default)
 - Used if another route table not associated

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

A route table is used to configure how traffic is routed within a VPC. It is comprised of rules (also called routes) that determine where traffic is sent. Up to ten (10x) route tables with a maximum of twenty (20x) rules may exist for a VPC and each route table may be associated with more than one subnet.

When a VPC is created a *main* route table is also created which simply routes local traffic anywhere within the VPCs IP address range (however, additional routes can be added if required). Any subnet that is not explicitly associated with custom route table will be associated to the *main* route table.

Routes within a route table consist of a *destination* and a *target*. A rule is evaluated as such: “Any traffic going to *destination* route it via *target*”. The destination is a CIDR range representing where the traffic should ultimately end up and a target may be a specific instance ID, an ENI (Elastic Network Interface) ID, an Internet Gateway or a Virtual Private Gateway (VGW); a special VPC construct that routes traffic to the Internet.

The screenshot shows a slide from the 'Systems Operations on AWS' course, specifically the 'Amazon VPC I Gateways & Routes' module. The title of the slide is 'IGWs and NAT Instances'. The content is organized into three main bullet points:

- IGWs are routers that route traffic to the Internet
 - Each VPC may have one (1x) IGW
- NAT instances are normal EC2 instances
 - Build your own
 - Use a NAT AMI
 - They reside in a public subnet
 - Provide Internet access to instances in private subnets
- IGWs and NAT instances are used as targets of routes to allow internet access.
 - If the default route (0.0.0.0/0) targets an IGW it defines a public subnet
 - If the default route targets a NAT instance it defines a private subnet

At the bottom of the slide, there is a small copyright notice: © 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Each VPC may have an Internet Gateway attached to it. Think of an Internet Gateway (IGW) as a router that routes traffic to the Internet. A default VPC with a public subnet will already have a route table entry to route traffic via an IGW.

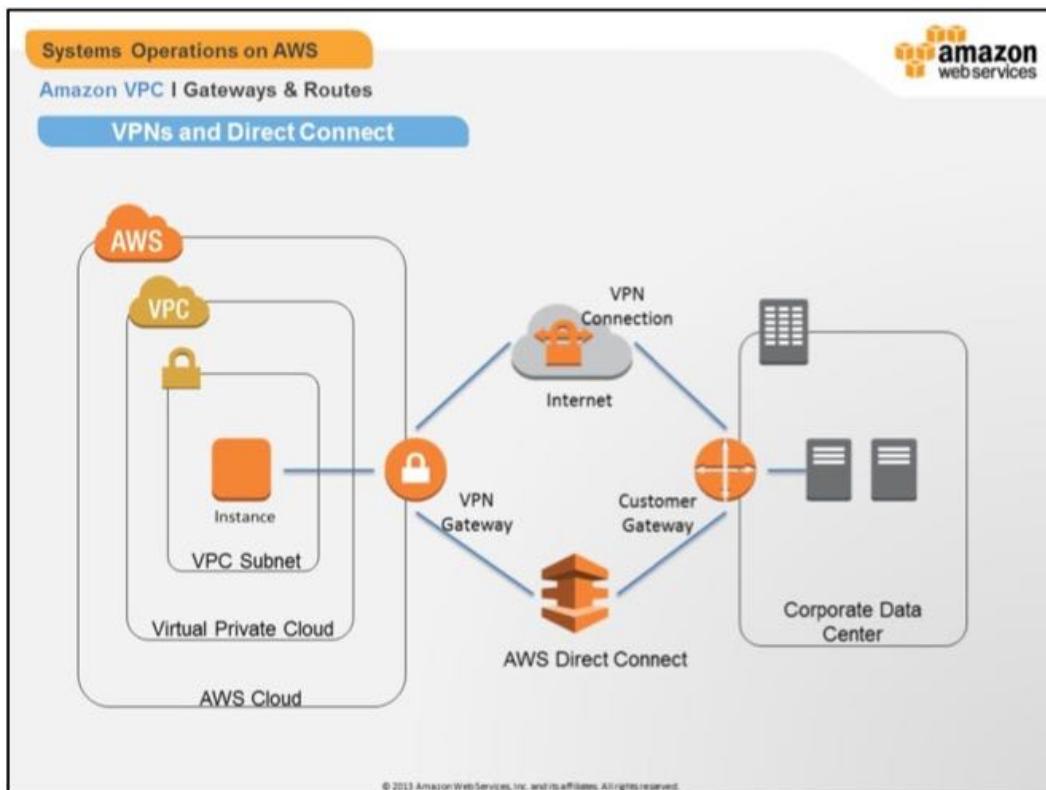
If no IGW exists for a VPC, simply create one from the Management Console and attach it to a VPC.

If Internet access is required by instances residing in non-public subnets within a VPC, then a NAT (Network Address Translation) instance will need to be provisioned in a public subnet that forwards traffic on via the IGW. A route table for the non-public subnet will need to be configured to route Internet traffic via the NAT instance. In order to set up a NAT instance:

- Create a security group for the NAT instance
- Launch a NAT instance into a public subnet:
 - Use a pre-built NAT AMI (Amazon Machine Image) –or–

- Use a standard Linux AMI and enable IP forwarding/masquerading
- Disable the `SrcDestCheck` attribute for the NAT instance
- Associate an Elastic IP (EIP) address with the NAT instance
- Ensure that the subnet the NAT instance is deployed in has a rule to route traffic via the IGW

Once the NAT instance has been launched route table entries associated with the private subnets may route via it.



This diagram is showing the two connectivity options, a VPN and a AWS Direct Connect. Note that the end-points of the AWS Direct Connect and VPN may potentially be the same device (albeit different configurations). The next slide will go over these components in more details.

Systems Operations on AWS
Amazon VPC | Gateways & Routes
VPNs and Direct Connect

- Treat the VPC as an extension of a datacenter
 - i.e. existing corporate network
- Connect via Internet (VPN) or AWS Direct Connect
- VPN is an IPSEC connection
 - Between customer's DC and AWS
 - Two tunnels are provisioned for additional availability
- Components:
 - Customer Gateway
 - Hardware or software appliance
 - Virtual Private Gateway (VGW)

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

By connecting other networks to the VPC the customer is effectively extending their network into the AWS cloud. Traffic can be routed across the connection as if it was all on the same physical network, e.g. EC2 instances running application servers in a VPC can connect to mainframe systems sitting in the corporate data-center.

AWS offers two mechanisms of connecting on-premise corporate datacenters to AWS Virtual Private Clouds. The first is a VPN (Virtual Private Network) that is established over the Internet. With a VPN connection the customer gets two tunnels for additional availability (also allowing for AWS to bring one down for maintenance if need be without causing interruption to service). The VPN is configured from the AWS Management Console and consists of configuring:

- A VGW (Virtual Private Gateway) – a virtual router on the AWS side to facilitate one end of the connection
- Customer Gateway – a hardware or software appliance on the customer side for the other end of the connection

The second is AWS Direct Connect which is essentially a 1 or 10 Gb connection

between the customers data-center and an AWS co-lo provider. The method of connectivity mitigates any of the concerns/issues with connecting via a VPN over the Internet.



Security

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Amazon VPC I Security

NACLs and Security Groups

- Network ACLs (Access Control Lists)
 - Stateless, inbound/outbound rules must be defined
 - Associated with subnets
- Security Groups
 - Stateful
 - Inbound/outbound rules at the ENI (interface) level
 - By default, all inbound traffic denied
 - By default, all outbound traffic allowed

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Within a VPC, security is controlled using security groups and NACLs (Network Access Control Lists). NACLs are associated with specific subnets within a VPC. As they are stateless; both inbound and outbound rules (ingress and egress) must be defined.

Inbound and outbound rules are defined by:

- Specifying the type of rule, e.g. custom TCP
- Providing a rule number (rules are processed lowest to highest)
- Defining the port range that will be allowed or denied in/out
- Specifying the source or destination IP address/range that will be allowed or denied in/out

Once the NACL has been defined it can be associated to subnets within the VPC.

In addition to NACLs, security groups can also be used to restrict traffic in and out of ENIs (Elastic Network Interfaces) that are associated with them. Some important things to note with VPC security groups:

- Only 5x security groups can be associated with an ENI
- Limit of 100x security groups per VPC
- By default no inbound traffic is allowed
- By default all outbound traffic is allowed
- All outbound traffic in response to an inbound request is permitted
- Instances that are part of the same security group cannot communicate with each other by default

NACLs vs. Security Groups

- Network Security groups are responsible for maintaining NACLs
- Software Developer groups are responsible for maintaining Security Groups

We will talk about security in more detail later in this course!



Amazon VPC + Amazon CloudFormation

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Amazon VPC | VPC and CloudFormation (CFN)

Using CloudFormation

- Valid JSON syntax
- Meta-programming
- Supports:
 - references
 - parameters
- Functions
- Well suited for VPC scaffolding!

```
  "PublicSubnet1" : {  
    "Type" : "AWS::EC2::Subnet",  
    "Properties" : {  
      "VpcId" : {  
        "Ref" : "VPC"  
      },  
      "CidrBlock" : {  
        "Ref" : "PublicSubnet1CIDR"  
      },  
      "AvailabilityZone" : {  
        "Fn::Select" : {  
          "1" : {  
            "Ref" : "EnabledVPCAZs"  
          }  
        }  
      }  
    }  
  }  
}
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

This module doesn't provide definitive in depth coverage of CloudFormation; but it's important to point out:

- CloudFormation is valid JSON (JavaScript Object Notation)
- It's essentially meta-programming; i.e. programming AWS resources with data descriptions
- Blocks of CloudFormation can be referenced, much the way language include definitions work (C macros for example); you only need to change something in one place and anything referencing it will adjust accordingly
- CloudFormation provides support for parameterization as opposed to hard-coding certain variables; this encourages re-use and flexibility when deploying AWS resources
- There are also rudimentary functions that can be used to work with for example lists, maps, strings and AWS constructs such as regions etc.
- Often the VPC scaffolding/network is the least dynamic component of an AWS environment (after all, you want the network layer to typically be the most stable

piece right?), VPC topology is often “replicated” in other regions or for other purposes, such as staging and development environments that need to be as close to production as possible. Leveraging the power of CloudFormation is beneficial in these situations.

Systems Operations on AWS

Amazon VPC | VPC and CloudFormation (CFN)

Using CloudFormation

- Makes provisioning VPCs:
 - Consistent
 - Reliable
- CloudFormation provisions "resources"
 - VPCs
 - Subnets
 - Security Groups, NACLs etc.
 - Instances
- CloudFormation stacks can be "updated"
- Encourages re-usability
- Enables version control of CloudFormation templates

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



CloudFormation allows developers and administrators to manage and create collections of AWS resources. In the context of VPC, these resources may include:

- VPCs themselves
- Subnets
- NACLs
- Security Groups
- Elastic Load Balancers

In fact, almost everything discussed in this module can be described as a resource in a CloudFormation template. CloudFormation templates are simply JSON text files. It's not a programming language in the traditional sense, more of a meta-programming language.

As the templates are simply text files they can be:

- Put under version control
- Be updated
- Re-used to deploy other similar resource constructs in other AWS regions
- Programmatically created if required

CloudFormation ensures that resources are created reliably and consistently every time. Deployment events and resources are subsequently logged, so at any point in time what's been deployed and what hasn't is available to the end user.

The last part of the lab for this module will focus on building out a fully populated VPC using a pre-built CloudFormation template.

Systems Operations on AWS

Amazon VPC | VPC and CloudFormation (CFN)

Some VPC Limits

- 5 VPCs per AWS region
- 200 subnets per VPC
- 10 route tables per VPC
- 50 NACLs per VPC
- 100 Security Groups per VPC
- 50 VPN connections per region
- 5 EIPs per region for each AWS account
- And there is more...

URL to view AWS VPC Limits

http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Appendix_Limits.html

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Note:

Most of these are soft limits and can be raised on a per customer basis should the requirement arise.

Some VPC limitations have already been discussed; but there are other limits worth mentioning in regards to VPC that are highlighted below:

<u>Component</u>	<u>Limit</u>
<u>Comments</u>	
VPCs for AWS region	5
Subnets per VPC	200
Internet Gateways per region VPC	5 1x per
Virtual Private Gateways per region VPC	5 1x per

Customer gateways per region	50	
VPN connections per region Virtual Private Gateway	50	10x per
Route tables per VPC main route table	10	Including
Rules/routes per route table	20	
EIPs per region for each AWS account	5	
Security groups per VPC	100	
Rules per security group	50	
Security groups that can be assigned to an ENI in a VPC	5	
Network ACLs per VPC	50	
Rules per network ACL	20	
BGP advertised routes per VPN	100	

URL to VPC Limits

http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Appendix_Limits.html

Systems Operations on AWS

Amazon VPC I Summary

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



- VPCs are foundation components of AWS architectures
 - Isolated virtual networks within the cloud
- They facilitate the creation of subnets (public & private)
 - Routing of traffic is facilitated using route tables
- Constructs exist that enable securing of resources
 - NACLs and security groups
- Customers can extend their own networks into VPCs
- All new accounts will automatically get a default VPC
- Use CloudFormation for consistent deployment of VPCs

Summary

- VPCs are foundation components of AWS architectures
 - Isolated virtual networks within the cloud
- They facilitate the creation of subnets (public & private)
 - Routing of traffic is facilitated using route tables
- Constructs exist that enable securing of resources
 - NACLs and security groups
- Customers can extend their own networks into VPCs
- All new accounts will automatically get a default VPC
- Use CloudFormation for consistent deployment of VPCs

Bullet #5: NOTE: All new accounts will automatically get a default VPC - this is **not yet in us-east-1!**



LAB

Amazon VPC

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS
Amazon VPC I Lab Exercise



Part One:

- Provisioning a VPC
- Creating two (2x) subnets in the VPC
- Locking down access to instances within the VPC
- Deploying and configuring NAT instance

Part Two:

- Demonstrate the ease at which a VPC can be provisioned using CloudFormation Templates

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Part One:

- Provisioning a VPC
- Creating two (2x) subnets in the VPC
- Locking down access to instances within the VPC
- Deploying and configuring NAT instance

Part Two:

- Demonstrate the ease at which a VPC can be provisioned using CloudFormation Templates



Module 3: Identity & Access Management (IAM)

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

AWS IAM | Learning Objectives

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



The slide contains a list of learning objectives for AWS IAM. The objectives are:

- Secure your AWS account
- Know when and why to use IAM
- Perform the following:
 - properly enable IAM
 - build policies
 - create and manage users and groups
 - create and manage roles and instance profiles
 - create and manage credentials
- Understand temporary credentials, account federation and delegation
- Differentiate IAM policies and resource-specific policies

Learning Objectives

- Secure your AWS account
- Know when and why to use IAM
- Perform the following:
 - properly enable IAM
 - build policies
 - create and manage users and groups
 - create and manage roles and instance profiles
 - create and manage credentials
- Understand temporary credentials, account federation and delegation
- Differentiate IAM policies and resource-specific policies



AWS IAM Introduction

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

AWS IAM | Introduction

What is IAM

- AWS IAM controls access to AWS services and resources
- Provides user and group permission management
- Provides password complexity requirements
- Enables identity federation between your corporate directory and AWS services
- Deep integration into some services such as EC2, S3 and DynamoDB
- Supports logon through AWS Management Console
- Not for operating systems or applications

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



AWS Identity and Access Management (AWS IAM) enables a customer to create multiple users and manage the permissions for each of these users within their AWS Account. A user is an identity (within a customer AWS Account) with unique security credentials that can be used to access AWS Services. AWS IAM eliminates the need to share passwords or access keys, and makes it easy to enable or disable a user's access as appropriate.

AWS IAM enables customers to implement security best practices, such as least privilege, by granting unique credentials to every user within their AWS Account and only granting permission to access the AWS Services and resources required for the users to perform their job. AWS IAM is secure by default; new users have no access to AWS until permissions are explicitly granted.

IAM also enables identity federation between your corporate directory and AWS services. This lets you use existing corporate identities to grant secure access to AWS resources, such as Amazon S3 buckets, without creating new AWS identities for those users.

AWS IAM enables customers to minimize the use of their AWS Account credentials. Instead all interactions with AWS Services and resources should

be with AWS IAM user security credentials. More information about AWS Identity and Access Management (AWS IAM) is available on the AWS website: <http://aws.amazon.com/iam/>

Systems Operations on AWS

AWS IAM | Introduction

What Does IAM Provide

- A username for each user
- Groups to manage multiple users
- Centralized access control
- Optional provisions:
 - Password for console access
 - Policies to control access to AWS APIs
 - Two methods to sign API calls:
 - Access Key ID + Secret Access Key
 - Multifactor Authentication

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Using IAM you can create and manage AWS users and groups and use permissions to allow and deny their permissions to AWS resources.

Add IAM users to your AWS account, then create groups to easily manage permissions for multiple IAM users under your AWS account.

Note: Within your account, a friendly name for a user or group must be unique.

In the example shown here, users and applications (service accounts provided to resources) have been grouped together by logical function.

Signing certificates (not shown above):

Signing certificates are used in only some services. For example, in Amazon EC2 you can use a signing certificate for SOAP requests. However, Amazon EC2 is in the process of deprecating SOAP support, and recommends instead that you use the Query API. In Amazon EC2, Auto Scaling, Elastic Load Balancing, and Amazon CloudWatch, you can use a certificate to configure credentials for the CLI. As an alternative to using a certificate, you can use access credentials (an Access Key ID and a Secret Access Key) to configure the CLI.

Systems Operations on AWS

AWS IAM | Introduction

Multifactor Authentication (MFA)

- Helps prevent anyone with unauthorized knowledge of your credentials from impersonating you
- Additional protection for account information
- Also available as virtual token
- Works with
 - Master Account
 - IAM Users
- Integrated into
 - AWS API
 - AWS Management Console
 - S3 (Secure Delete)



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Amazon Account Security Features

AWS provides a number of ways for customers to identify themselves and securely access their AWS Account. A complete list of credentials supported by AWS can be found on the Security Credentials page under Your Account. AWS also provides additional security options that enable customers to further protect their AWS Account and control access: AWS Identity and Access Management (AWS IAM), Multi-Factor Authentication (MFA) and Key Rotation.

AWS Multi-Factor Authentication (AWS MFA)

AWS Multi-Factor Authentication (AWS MFA) is an additional layer of security that offers enhanced control over AWS Account settings and the management of the AWS Services and resources for which the account is subscribed. When customers enable this opt-in feature, they will need to provide a six-digit single-use code in addition to their standard username and password credentials before access is granted to their AWS Account settings or AWS Services and resources. Customers get this single use code from an authentication device that they keep in their physical possession. This is called Multi-Factor Authentication because two factors are checked before access is granted: customers need to provide both their username (Amazon e-mail in the case of the AWS Account) and password (the first “factor”:

something you know) and the precise code from their authentication device (the second “factor”: something you have). Customers can enable MFA devices for their AWS Account as well as for the users they have created under their AWS Account with AWS IAM.

It is easy to obtain an authentication device from a participating third party provider and to set it up for use via the AWS website. More information about Multi-Factor Authentication is available on the AWS website:

<http://aws.amazon.com/mfa/>

The screenshot shows a slide from the AWS Systems Operations on AWS – Student Guide. At the top left, there's a breadcrumb navigation: "Systems Operations on AWS" → "AWS IAM" → "Introduction". Below that, a blue header bar contains the title "Server Certificates". The main content area contains a bulleted list of points about server certificates:

- IAM stores Server Certificates for Elastic Load Balancing
- Used to terminate SSL using ELB when you choose HTTPS or SSL listener
 - If you wish to pass SSL through to the servers, choose TCP listener
- Upload server certificates into IAM
 - X.509 from a legitimate Certificate Authority (CA)
- Stored in IAM and fetched by ELB
- Configuration components:
 - Private key (.pem)
 - Public key (.pem)
 - (optional) Public key certificate chain (.pem)

At the bottom of the slide, there's a small note: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

Server certificates are also known as public key certificates.

Currently, Amazon Elastic Load Balancing is the only service to support the use of server certificates with IAM.

Elastic Load Balancing supports the load balancing of applications using HTTP, HTTPS, TCP, and Secure Sockets Layer (SSL) protocols. SSL protocol establishes secure connections between a client and the back-end server and ensures that all the data passed between your client and your server is private and integral. You can specify the protocols for the front-end connections (client to load balancer) and the back-end connections (load balancer to back-end instance) independently. If you choose HTTPS/SSL protocols for your front-end connection, the back-end connection to the instance can either be in plain text or HTTPS/SSL. If you choose HTTP for your front-end connection, the back-end connection to the instance can be HTTP or HTTPS.

You need to specify an SSL certificate for HTTPS or SSL listeners. Specifying a cipher policy is optional. A default policy will be used if none is specified.

You may select a previously uploaded certificate, or define a new SSL Certificate by supplying certificate name, a private key (pem encoded), and a public key certificate (pem encoded). You may also provide an optional public key certificate chain (pem encoded).

Systems Operations on AWS

AWS IAM | Introduction

Limitations

- Some entity naming restrictions associated with special characters
- By default, IAM limits resources, such as:
 - **Number of users**
 - Number of user policies
 - Size of each user policy
 - **Number of groups**
 - **Number of groups per user**
 - Number of group policies
 - Size of each group policy
 - **Number of roles**
 - **Number of instance profiles**
 - **Number of server certificates**
- Limits in **bold** above can be modified via limit increase request form

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Name Restrictions:

- Names of users, groups, roles, instance profiles, and server certificates must be alphanumeric, including the following common characters: plus (+), equal (=), comma (,), period (.), at (@), and dash (-).
- Path names must begin with a forward slash (/).
- Policy names must be unique to the user, group, or role they are attached to, and can contain any Basic Latin (ASCII) characters, minus the following reserved characters: backward slash (\), forward slash (/), asterisk (*), question mark (?), and white space. These characters are reserved according to RFC 3986 (for more information, see <http://www.ietf.org/rfc/rfc3986.txt>).
- User passwords (login profiles) can contain any Basic Latin (ASCII) characters.
- AWS account ID aliases must be unique across AWS products, and must be alphanumeric following DNS naming conventions. An alias must be lowercase, it must not start or end with a hyphen, it cannot contain two consecutive hyphens, and it cannot be a 12 digit number.

Names for entities are case sensitive and must be unique within the scope of your AWS account (regardless of the path you might give the entity).

Resource Limits:

By default your AWS (root) account has initial quotas set for all IAM-related entities. It is currently not possible to set usage quotas on IAM users.

Please note these quotas are subject to change. If you require an increase you can always use the IAM Limit Increase Contact Us Form.

The screenshot shows the AWS Systems Operations on AWS interface. In the top navigation bar, the 'Systems Operations on AWS' tab is highlighted. Below it, the 'AWS IAM | Introduction' page is displayed. On the left, under the 'Quotas' tab, there are two sections: 'CLI command:' and 'API action:', each with an orange circular icon. The 'CLI command:' section contains the command 'aws iam get-account-summary'. The 'API action:' section contains the action 'GetAccountSummary'. To the right, a terminal window titled 'PROMPT>' shows the execution of the command 'aws iam get-account-summary', followed by a JSON object representing the account summary data. The JSON output includes various quota details such as 'AccessKeysPerUserQuota', 'AssumeRolePolicySizeQuota', 'UsersQuota', etc. At the bottom of the terminal window, a copyright notice reads '© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.'

```
PROMPT> aws iam get-account-summary
{
    "SummaryMap": {
        "AccessKeysPerUserQuota": 2,
        "AssumeRolePolicySizeQuota": 2048,
        "UsersQuota": 5000,
        "GroupsPerUserQuota": 10,
        "Users": 4,
        "Roles": 5,
        "MFADevices": 1,
        "InstanceProfilesQuota": 100,
        "AccountMFAEnabled": 0,
        "ServerCertificates": 1,
        "UserPolicySizeQuota": 2048,
        "RolePolicySizeQuota": 10240,
        "MFADevicesInUse": 1,
        "GroupsQuota": 100,
        "Groups": 2,
        "InstanceProfiles": 5,
        "GroupPolicySizeQuota": 5120,
        "SigningCertificatesPerUserQuota": 2,
        "ServerCertificatesQuota": 10,
        "RolesQuota": 250
    },
    "ResponseMetadata": {
        "RequestId": "947e3826-09e4-11e3-a828-89dcd5606dba"
    }
}
```

To retrieve account level information about entity usage and quotas, use the GetAccountSummary API action or the iam-accountgetsummary CLI command.

The screenshot shows a slide titled "AWS IAM | Introduction" under the "Systems Operations on AWS" course. The slide has a blue header bar with the course name. Below it, a blue button-like bar contains the title "Basic Definitions". The main content area contains two bullet points under the heading "Amazon Resource Name (ARN)".

- Amazon Resource Name (ARN)
 - Uniquely identifies AWS resources
 - Used to specify a resource unambiguously across all of AWS, such as in IAM policies, and API calls

Below these definitions, there is a "Format:" section with the placeholder "arn:aws:service:region:account:resource" and an "Example:" section with the placeholder "arn:aws:iam::123456789012:user/Bob". At the bottom of the slide, a small note reads: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

Before we proceed, we need to go over some basic definitions.

Amazon Resource Name (ARN)

- Uniquely identifies AWS resources
- Used to specify a resource unambiguously across all of AWS, such as in IAM policies, and API calls

The “region” value for IAM is always left blank. “Why?” (Answer: IAM is region independent)

Principal

- Specifies the user, account, service, or other entity that is allowed or denied access to a resource

Systems Operations on AWS

AWS IAM | Introduction

Knowledge Check



1. True/False: I can require users to use a MFA when deleting objects from S3.
2. True/False: I should use IAM for OS level authentication.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

1. True.
2. False.



Using AWS IAM

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The screenshot shows a slide from the AWS Systems Operations on AWS course. The slide title is "Protecting Access Keys". It contains a bulleted list of best practices for managing access keys:

- Unique Access Keys can be created for each IAM user
- Used to sign API calls
- Access Keys have two components:
 - Access Key ID: AKIAIOSFODNN7EXAMPLE
 - Secret Access Key: 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
- Rotate Access Keys regularly
 - Two active pairs allowed to enable rotation
- Places Access Keys don't belong
 - GitHub
 - Baked into AMIs
 - Word docs (printers)
 - Unencrypted email

At the bottom of the slide, there is a small copyright notice: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

Key Rotation

For the same reasons as it is important to change passwords frequently, AWS recommends that customers rotate their access keys and certificates on a regular basis. To let customers do this without potential impact to their applications' availability, AWS supports multiple concurrent access keys and certificates.

With this feature, customers can rotate keys and certificates into and out of operation on a regular basis without any downtime to their application. This can help to mitigate risk from lost or compromised access keys or certificates. The AWS IAM APIs enables a customer to rotate the access keys of their AWS Account as well as for users created under their AWS Account using AWS IAM.

The screenshot shows a slide from the 'Systems Operations on AWS' course. At the top left, there are three navigation items: 'Systems Operations on AWS' (orange), 'AWS IAM | Using IAM' (blue), and 'User Groups' (blue, currently selected). At the top right is the Amazon Web Services logo. The main content area contains a single bullet point under a yellow circle:

- Configure and use user groups
 - Assign permissions to logical and functional grouping of your organization
 - Bulk permissions management (scalable)
 - Easy to change permissions as individuals change teams (portable)

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Why do we want to use groups? If you have used groups in other directory systems, the reasons are similar – operational efficiency.

Suppose you add a new S3 bucket, and want to assign permissions. What's simpler? Assigning permissions to hundreds or thousands of users to that need write access to the bucket, or assigning permissions to a few groups?

When Lisa moves from QA to development, it's much easier to move her from one group to another instead of reconfiguring permissions on all associated resources.

Similarly, when deprovisioning a user who has left the company, it is easier to remove the user from a group than remove permissions from all associated resources.

Systems Operations on AWS

AWS IAM | Using IAM

For AWS Administrators

- AWS account = Root account (email login)
- Create and use IAM admin accounts for daily operations
- Why?
 - No ability to limit privileges to the AWS root account
 - Prevent using shared accounts for multiple admins
 - Enables deprovisioning of admin accounts
 - Some features only available with IAM users
 - Password policies
 - Group membership
 - Ability to assume a role on another AWS account
 - Tighter, least-privilege permissions

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Each AWS account is registered using one email address. When you first log in using the AWS console, you'll use this email address and password. This account should NOT be used for daily administration purposes. Instead, create IAM accounts, put them into an administrative group, and assign administrative privileges to the group.

In short, the AWS root account is for:

- Creating initial administrative IAM users (who can then create additional users)
- Configuring billing and account information
- Initial setup and preparation of the AWS account (next slide)

Systems Operations on AWS

AWS IAM | Using IAM

For AWS Administrators

- First, prepare the environment using the root account:
 - Add multifactor authentication (MFA)
 - Password with complexity and high entropy
 - Configure your security challenge questions
 - Optional configuration:
 - Enable IAM admin user access to AWS Support Center [1]
 - Enable IAM admin user access to billing account activity [2]
 - Enable IAM admin user access to billing usage reports [2]
- Configure IAM admin user(s)
- Lock the root account password and MFA in a safe

[1] <https://aws.amazon.com/premiumsupport/iam>
[2] <http://docs.aws.amazon.com/awsaccountbilling/latest/about/ControllingAccessWebsite.html>

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



What is password entropy? It is usual in the computer industry to specify password strength in terms of information entropy, measured in bits, a concept from information theory. Instead of the number of guesses needed to find the password with certainty, the base-2 logarithm of that number is given, which is the number of "entropy bits" in a password. A password with, say, 42 bits of strength calculated in this way would be as strong as a string of 42 bits chosen randomly, say by a fair coin toss. Put another way, a password with 42 bits of strength would require 242 attempts to exhaust all possibilities during a brute force search. Thus, adding one bit of entropy to a password doubles the number of guesses required, which makes an attacker's task twice as difficult. On average, an attacker will have to try half the possible passwords before finding the correct one.

Which services do not support IAM?

Some services and features do not support IAM and must use the AWS root account credentials:

- AWS Import/Export
- Vulnerability Scanning Coordination Form
- Visibility of all Data Pipelines
- Specific account management pages, such as account profile, AWS account security credentials, payment methods and management of consolidated billing
- AWS DevPay
- CloudFront keypair creation

Support permissions for IAM users are available for Business and Enterprise-level support customers. Basic, Developer and Silver-level support customers can continue to open cases using their main AWS account credentials and if applicable, Named Contacts users. IAM policies configured for AWS accounts without Business or Enterprise-level support will have no effect - users will be redirected to this page when they navigate to the AWS Support Center or Trusted Advisor.

The screenshot shows the AWS Management Console interface for IAM Password Policy. At the top, there are navigation links: 'Systems Operations on AWS' (highlighted), 'AWS IAM | Using IAM', and 'IAM Password Policy'. On the left, a sidebar lists 'Dashboard', 'Details', 'Groups', 'Users', 'Roles', and 'Password Policy' (which is highlighted with a red box). The main content area is titled 'Password Policy' and contains the following text: 'A password policy is a set of rules that define the type of password an IAM user can set. For more information about password policies, go to Using IAM.' Below this, a message states: 'Currently, this AWS account does not have a password policy. Specify a password policy below.' A 'Minimum Password Length' input field is set to '6'. Underneath, there is a list of five options, each with a checkbox and a help icon: 'Require at least one uppercase letter', 'Require at least one lowercase letter', 'Require at least one number', 'Require at least one non-alphanumeric character', and 'Allow users to change their own password' (which is checked). At the bottom are 'Apply Password Policy' and 'Delete Password Policy' buttons.

To display your current password policy, in the navigation pane of the console, click Password Policy. If you do not have a password policy, you will see a message indicating that a password policy hasn't been set for your AWS account.

Note the options in the screenshot here. Start with your organization's security policy – which of these options will fulfill those requirements?

Systems Operations on AWS

AWS IAM | Using IAM

Password Policy

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



● Special Characters: ! @ # \$ % ^ & * () _ + - = [] { } | ' .

● "Allow users to change their own password"

- Global IAM setting – all users
- Enables users to change password via API, CLI or Console
- Alternatively, you can limit to specific user groups by attaching a policy statement

Special Characters: ! @ # \$ % ^ & * () _ + - = [] { } | '

"Allow users to change their own password"

- Global IAM setting – all users
- Enables users to change password via API, CLI or Console
- Alternatively, you can limit to specific user groups by attaching a policy statement

Systems Operations on AWS

AWS IAM | Using IAM

Password Policy

- Sample policy statement to allow password changing
- Attach this IAM policy to a user group

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "iam:ChangePassword",  
                "iam:GetAccountPasswordPolicy"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Sample policy statement to allow password changing. Attach this IAM policy to a user group.

(This may be the first time we've seen an access policy.)

Access policies are written as Java Script Object Notation (JSON)

documents. In the Security module, we will further discuss the structure, policy elements, and applications of the access policy language.

The screenshot shows a slide from the AWS Systems Operations on AWS course. At the top left, there are navigation links: 'Systems Operations on AWS' (highlighted in orange), 'AWS IAM | Using IAM', and 'Policy Variables' (highlighted in blue). At the top right is the Amazon Web Services logo. The main content area contains a bulleted list of benefits of using policy variables:

- Access Policy placeholders
- Saves you from having to explicitly list all policy components
- Variables replaced with values when Access Policies are evaluated
- Can greatly reduce the necessary number of policies
 - Ex: locking down users' access to specific S3 folders determined by their username

At the bottom of the slide, a small note reads: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

In IAM policies, you provide a name for the specific resources that you want to control access to. In some cases, you might not know the exact name of the resource when you write the policy. For example, you might want to allow each user to have his or her own objects in an Amazon S3 bucket, as in the previous example. However, instead of creating a separate policy for each IAM user that specifies the user's name as part of the resource, you want to create a single group policy that works for any user in that group.

You can do this by using policy variables, which is a feature that lets you specify placeholders in a policy. When the policy is evaluated, the policy variables are replaced with values that come from the request itself.

Note: Must include Version element in statement, with value of 2012-10-17 or later.

Variables for Version 2012-10-17

- *aws:CurrentTime* (for date/time conditions)
- *aws:EpochTime* (the date in epoch or UNIX time, for use with date/time conditions)
- *aws:principalType* (a value that indicates whether the principal is an account, user, federated, or assumed role)

- *aws:SecureTransport* (Boolean representing whether the request was sent using SSL)
- *aws:SourceIp* (the requester's IP address, for use with IP address conditions)
- *aws:UserAgent* (information about the requester's client application, for use with string conditions)
- *aws:userid* (the unique ID for the current user—see the following chart)
- *aws:username* (the friendly name of the current user)

Full Details:

<http://docs.aws.amazon.com/IAM/latest/UserGuide/PolicyVariables.html>

Systems Operations on AWS

AWS IAM | Using IAM

Policy Variables Comparison

● Applicable to David:

```
{ "Version": "2012-10-17", "Statement": [ { "Action": ["iam:*AccessKey*"], "Effect": "Allow", "Resource": ["arn:aws:iam::123456789012:user/David"] } ] }
```

● Applicable to others:

```
{ "Version": "2012-10-17", "Statement": [ { "Action": ["iam:*AccessKey*"], "Effect": "Allow", "Resource": ["arn:aws:iam::123456789012:user/${aws:username}"] } ] }
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

You can use policy variables in a similar way to allow each IAM user to be able to manage his or her own access keys. A policy that allows a user to programmatically change the access key for user David looks like the example shown in the first policy on this slide. If this policy is attached to IAM user David, that user can change his own access key. As with the policies for accessing user-specific Amazon S3 objects, you'd have to create a separate policy for each user that includes the user's name, and then attach each policy to the individual users.

When you use a policy variable for the user name as shown in the second policy, you don't have to have a separate policy for each individual user. Instead, you can attach this new policy to an IAM group that includes everyone who should be allowed to manage their own access keys. When a user makes a request to modify his or her access key, IAM substitutes the user name from the current request for the \${aws:username} variable and evaluates the policy.

Systems Operations on AWS

AWS IAM | Using IAM

Knowledge Check

1. True/False: Permissions should be assigned to each IAM user.

2. True/False: I can allow specific user groups to change their own IAM passwords.

3. True/False: IAM can force users to select a password that is different from previously used passwords.

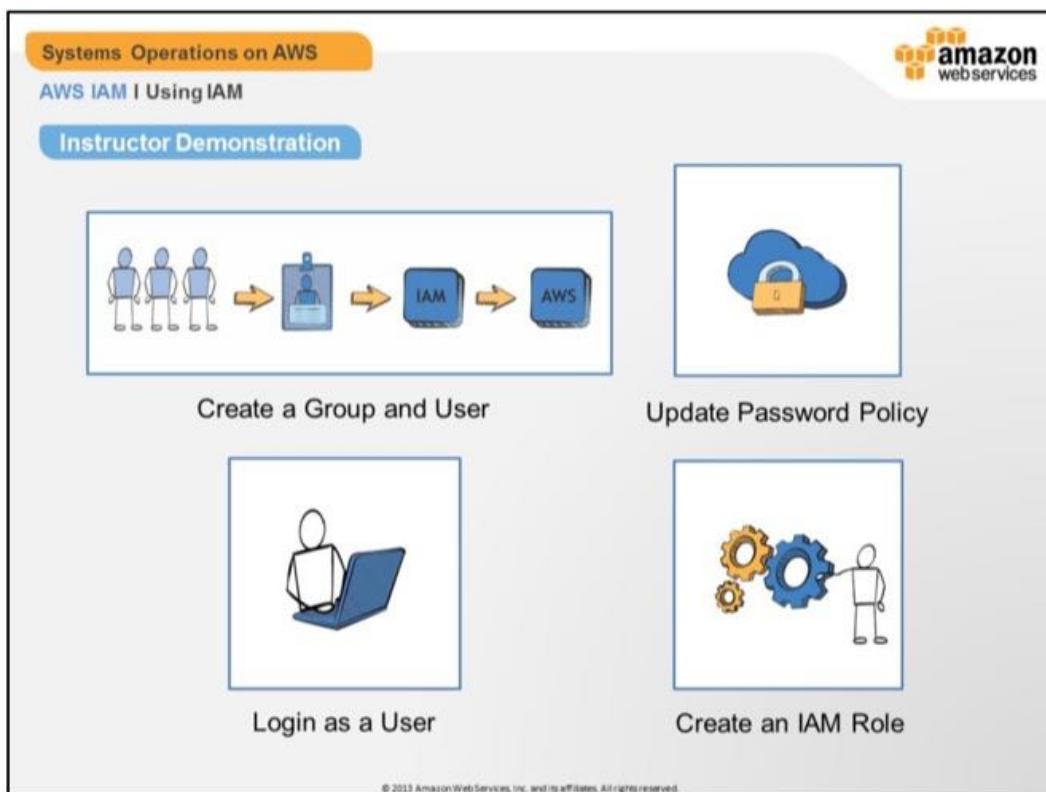
© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

1. False. Users should be placed in groups. Permissions should be assigned to groups.
2. True, through use of access policy.
3. False (although possible with some third party products)



IAM Instructor Demonstration

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Now let's take a look at IAM within the AWS Console.

Demonstration:

1. Create a Group and User with full permissions (administrator)
2. Update Password Policy and generate a password
3. Login as a User
4. Create an IAM Role

Systems Operations on AWS

AWS IAM | Using IAM

Create an Administrators Group and User

Create a Group and User

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- In the IAM console, click “Create a new group of users”.
- Name this group Administrators.
- Click “Continue”.
- Permissions are defined in a document called a policy. In a policy, you can define what actions are allowed or denied for specific AWS resources. You can use a custom policy or use a pre-defined set of permissions by selecting a policy template.
- Set the permissions for the Administrators group using the Administrator Access policy template.
- Click Continue.
- In the next screen, you will see that you can customize the permissions by editing the following policy document. In this example, keep the default. Click Continue.
- Click the “Create New Users” tab.
- Type <name> in the users box.
- Because <name> will only be accessing AWS through the management console, we do not need to create access keys. Uncheck the generate access keys box.
- Click Continue.
- Verify that the information for the group is correct, then click “Finish” to create the Administrators group and the IAM user, <name>.

Systems Operations on AWS

AWS IAM | Using IAM

Password Policy

Update Password Policy

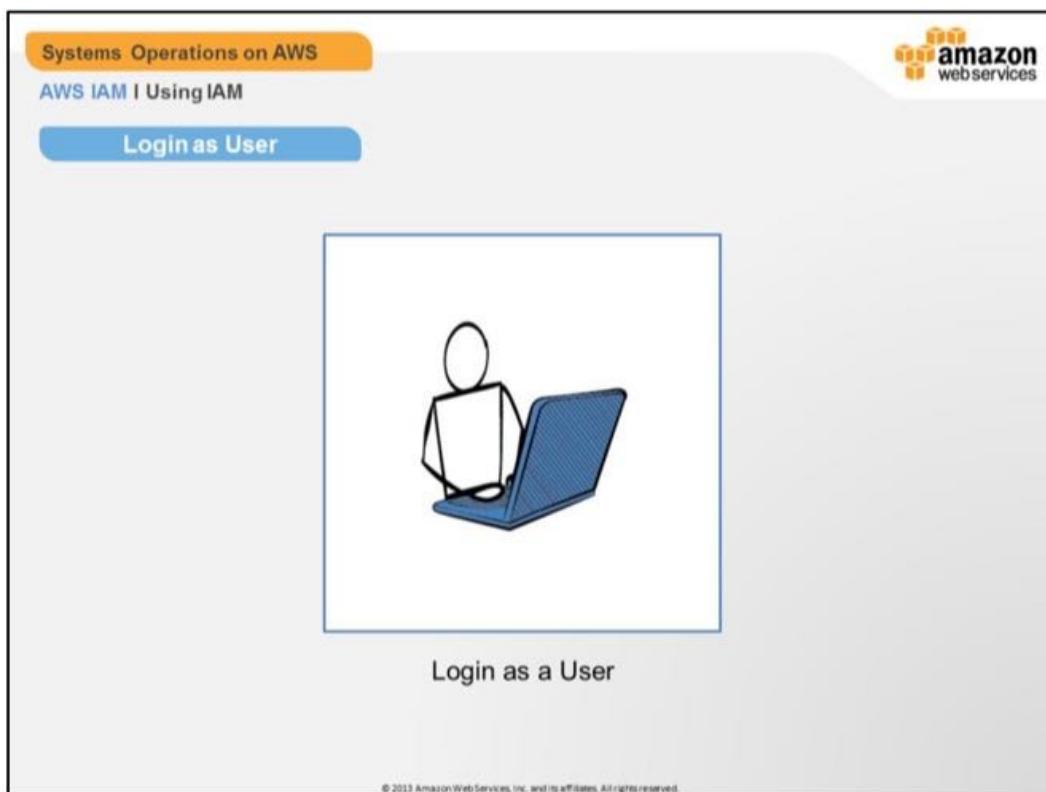
Update Password Policy

1. Click “Password Policy” under Details in the left column.
2. Change the Minimum Password Length to 8.
3. Check all policies:
 - a. Require at least one uppercase letter
 - b. Require at least one lowercase letter
 - c. Require at least one number
 - d. Require at least one non-alphanumeric character
 - e. Allow users to change their own password
4. Click “Apply Password Policy”.

Generate a Password for <Name>

1. Click “Users” under Details in the left column.
2. Check the box next to <Name>.
3. Under “User Actions”, click “Manage Password”.
4. Choose “Assign an auto-generated password” and click “Apply”.
5. Click “Download Credentials” button in the Manage Password window.
6. Click “Close Window”.
7. Click back to the IAM Dashboard by clicking “Dashboard” in the left column.
8. Under IAM User Sign-In URL, copy the link shown. If you want the URL

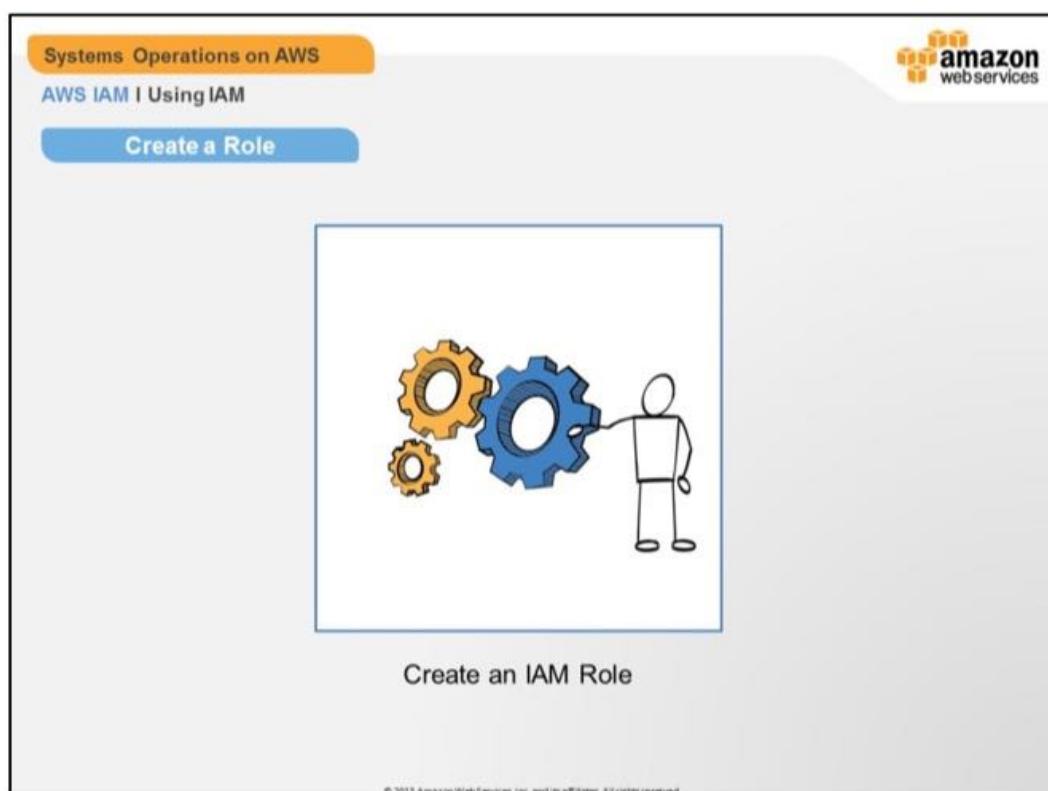
for your sign-in page to contain your company name (or other friendly identifier), you can click “Create Account Alias” to change this.



Log in with the IAM Sign-In URL as an Administrator

Login as the Administrator <Name>

1. Use the IAM User Sign-In URL that you copied in the last step.
2. Login with <Name's> username and password from the credentials that you downloaded earlier.



Create the Role

Use the available IAM lab in qwikLAB to run this demonstration.

- Create an S3 bucket and add an object such as a Word document
- Navigate to the IAM section of the AWS Management Console at <https://console.aws.amazon.com/iam/home>
- Click the **Roles** link
- Click the **Create New Role** button
- In the **Create Role** dialog box, enter a name for the role (for example “S3role”). The role name cannot contain spaces.
- Click **Continue**.
- In the **Select Role Type** screen, select the AWS Service Role of **Amazon EC2** and press **Select**.
- In the **Set Permissions** screen, select **Custom Policy** and click **Select**.
- Enter a name for the policy
- Review the role information and click **Create Role**.
- Launch an EC2 instance with the assigned role
- Login to the instance
- Browse files in the S3 bucket

IAM Policy next slide

Systems Operations on AWS

AWS IAM | Using IAM

Create a Role: AWS S3 Policy



Create an IAM Role

AWS S3 Policy

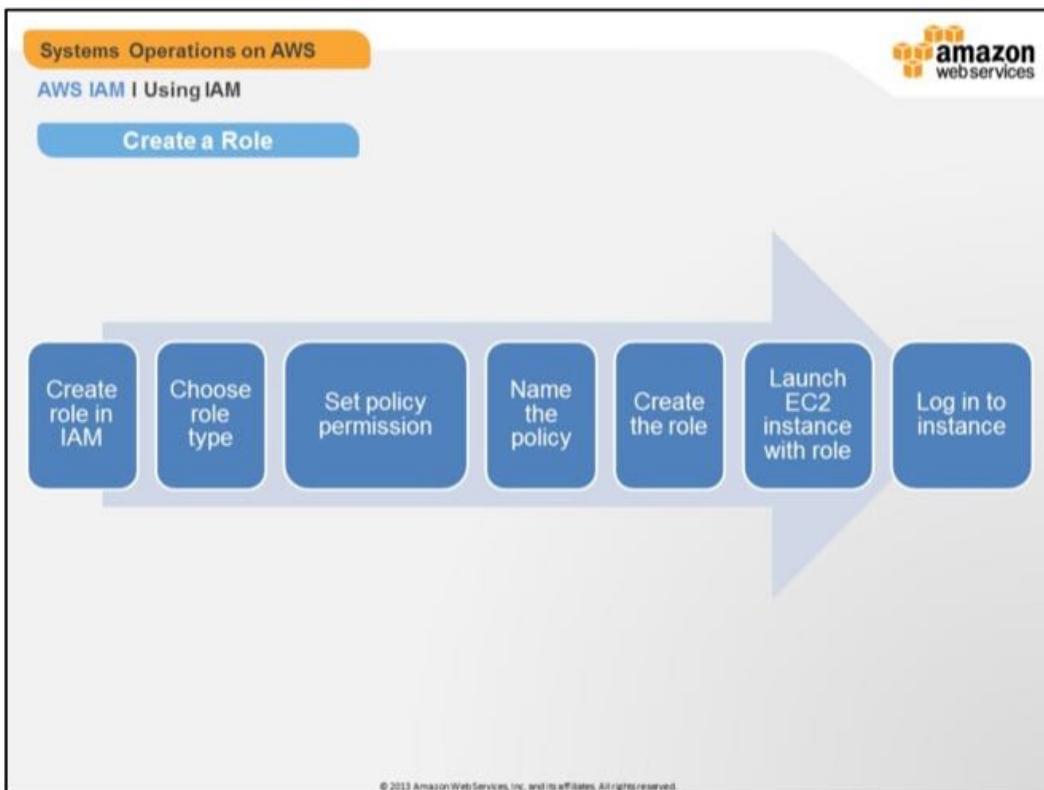
- Use the policy in the notes
- Replace the <bucket_name> with your bucket name

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

IAM S3 Policy

```
{  
  "Statement": [  
    {  
      "Sid": "Stmt1393452355346",  
      "Action": [  
        "s3>ListBucket"  
      ],  
      "Effect": "Allow",  
      "Resource": "arn:aws:s3:::<bucket_name>"  
    },  
    {  
      "Sid": "Stmt1393452379908",  
      "Action": [  
        "s3GetObject"  
      ],  
      "Effect": "Allow",  
      "Resource": "arn:aws:s3:::<bucket_name>/*"  
    }  
  ]}
```

}



Diagram

Create an IAM role



Roles, Delegation and Federation

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

AWS IAM | Roles, Delegates and Federation

Cross Account Authentication & Authorization

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- Applicable trusted entities:
 - IAM user groups from another AWS account
 - IAM user groups within the current account
 - Third parties
 - Root AWS accounts
- Applicable methods to grant access:
 - IAM roles
 - Resource-based policies for Amazon S3, SNS, SQS

IAM users groups from another AWS account:

You can establish cross-account access by using IAM roles. Using roles for cross-account access lets you grant access to any resources that the trusting account (Account A) has access to, as long as the resource is in a service that supports roles.

IAM users within the current account:

For mission critical permissions that IAM users might not frequently use, you can separate those permissions from their normal day-to-day permissions by using roles. Users would then have to actively assume a role, which can prevent them from accidentally performing disruptive actions.

For example, you might have Amazon EC2 instances that are critical to your organization. Instead of directly granting administrators permission to terminate the instances, you can create a role with those privileges and allow administrators to assume the role. Administrators won't have permission to terminate those instances; they must first assume a role. By using a role, administrators must take an additional step to assume a role before they can stop an instance that's critical to your organization.

Third Parties:

When third parties require access to your organization's AWS resources, you can use roles to delegate API access to them. For example, a third party might provide a

service for managing your AWS resources. With IAM roles, you can grant these third parties access to your AWS resources without sharing your AWS security credentials. Instead, they can assume a role that you created to access your AWS resources. Third parties must provide you the following information for you to create a role that they can assume:

- The AWS account ID that the third party's IAM users use to assume your role. You specify their AWS account ID when you define the trusted entity for the role.
- An external ID that the third party can associate you with your role. You specify the ID that was provided by the third party when you define the trusted entity for the role.
- The permissions that the third party requires in order to work with your AWS resources. You specify these permissions when defining the role's permission policy. This policy defines what actions they can take and what resources they can access.

After you create the role, you must share the role's Amazon Resource Name (ARN) with the third party. They require your role's ARN in order to assume the role.

Root AWS Accounts

We'll discuss this on the following slides.

You can establish cross-account access in these ways:

- By using IAM roles. You define a role in Account A that can be assumed by a user in Account B. Using roles for cross-account access lets you grant access to any resource in Account A that is in a service that supports roles.
- For some AWS services, you can put a policy directly on the resource you want to share, and specify that another account has access to the resource. The resource you want to share must support resource-based policies.

Systems Operations on AWS

AWS IAM | Roles, Delegates and Federation

AWS Root Trust and Delegation Example

- Root AWS Account A trusts Root AWS Account B for an S3 bucket
- Account B admin narrows down the access to the S3 bucket on Account A
- AWS root account B may then delegate access to its IAM users for resources in AWS root account A
 - But not beyond the permissions AWS root account B has been given

```
Account B (defined by Account A)
Root Account Access:
{
  "Statement": [
    {
      "Principal": "AWS" : "111122223333",
      "Action": "s3:*",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::mybucket/*"
    }
  ]
}

Account B IAM Users Delegated Access:
{
  "Statement": [
    {
      "Action": "s3>List*",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::mybucket/*"
    }
  ]
}
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Account A gives Account B full access to Account A's Amazon S3 bucket. As a result, Account B is authorized to perform any action on Account A's bucket, and Account B can grant (delegate) access to users under Account B. Account A's Amazon S3 bucket is named *mybucket*, and Account B's account number is 1111-2222-3333.

Account B gives its User Group 1 read access to Account A's Amazon S3 bucket. User Group 1 in Account B can view the objects in Account A's bucket. The level of access Account B can delegate is equivalent to, or less than, the access it has. In this policy, the Action element is explicitly defined to allow only List actions, and the Resource element of this policy matches the Resource for the bucket policy implemented by Account A.

Account B does not give access to User Group 2. Because User Group 2 has not been granted access to Account A's Amazon S3 bucket by Account B, by default, User Group 2 cannot access the bucket or the objects in the bucket.

Important

If Account B had used wildcards (*) to give a user full access to all its resources, that user would automatically have access to any resources that Account B has access to, including access granted by another account to its own resources. In this case, the user would have access to Account A's resources even though Account B did not specifically apply the permission.

IAM evaluates a user's permissions at the time the user makes a request. Therefore, if you use wildcards (*) to give users full access to your resources, users are able to access any resources your AWS account has access to, even resources you add or gain access to after creating the user's policy.

Systems Operations on AWS

AWS IAM | Roles, Delegates and Federation

Account Replication and Federation

- Replication
 - You can sync your directory accounts with IAM
 - Challenge: Accounts live in two locations
 - Requires re-sync after password changing
 - Requires two-step account deprovisioning
- Federation using AWS Security Token Service (STS)
 - Use your directory accounts without creating IAM accounts
 - Grant temporary access to corporate users for AWS
 - Uses a identity broker
 - Retrieves Access Policy for authenticated directory user
 - Retrieves access keys and session token
 - Can provide detailed logging
 - Cross-account access between AWS accounts
 - Grant permissions to other AWS accounts using resource-based policies
 - Uses an Identity Broker

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

By now, you know you can manually create IAM users with passwords for access to AWS resources using the AWS Management Console, AWS CLI or API calls. This is a great way to start, but you may already have an identity store (such as Active Directory) that you wish to leverage for AWS resource access. Two methods are possible: Replication and Federation.

With replication, you copy account credentials from your identity store into AWS IAM. This is useful for smaller organizations who need a quick solution without building ongoing connectivity between the two identity stores. While fast to execute, this method has some drawbacks such as:

- Limited to the number of accounts supported by IAM (5,000 default)
- Changes between identity stores are not automatically propagated. This includes password changes, and disabling or deleting of accounts.

Using federation with STS, you can grant temporary access to people in a corporate network without having to define individual IAM identities for each corporate user. You can also let federated users log into the AWS Management Console without having to be defined as IAM users, which we refer to as single sign-on (SSO).

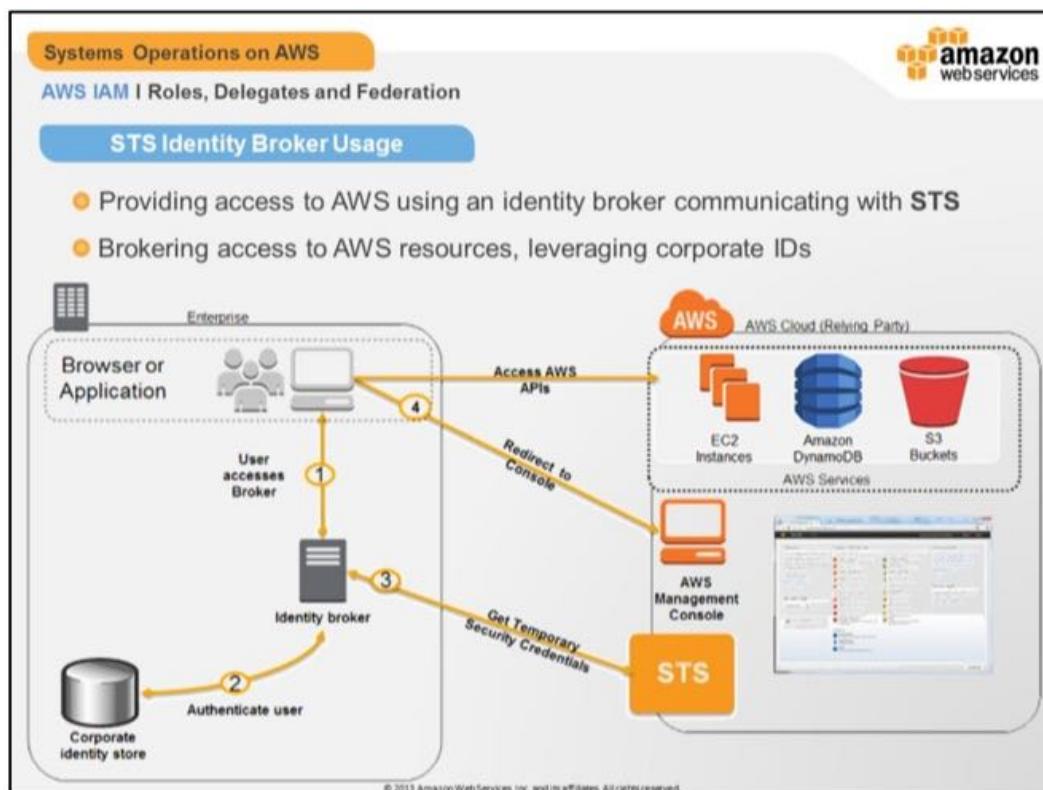
Identity Broker

AWS provides a C# sample that demonstrates how to build a Microsoft Active Directory proxy enabling Amazon Web Services (AWS) customers to leverage their existing identity directory in controlling access to AWS APIs.

<http://aws.amazon.com/code/1288653099190193>

Replication is a feasible option for small organizations. Because credentials are copied from one store to another, the accounts live in two places. When suspending or removing accounts, you must now remember to perform those steps in two places.

Federation requires additional setup, but scales to thousands of users.



In this scenario:

- The identity broker application has permissions to access the AWS STS API to create temporary security credentials.
- The identity broker application is able to verify that employees are authenticated within the existing authentication system.
- Users are able to get a temporary URL that gives them access to the AWS Management Console (which is referred to as single sign-on).

IAM users groups from another AWS account:

You can establish cross-account access by using IAM roles. Using roles for cross-account access lets you grant access to any resources that the trusting account (Account A) has access to, as long as the resource is in a service that supports roles.

IAM users within the current account:

For mission critical permissions that IAM users might not frequently use, you can separate those permissions from their normal day-to-day permissions by using roles. Users would then have to actively assume a role, which can prevent them from accidentally performing disruptive actions.

For example, you might have Amazon EC2 instances that are critical to your organization. Instead of directly granting administrators permission to terminate the instances, you can create a role with those privileges and allow

administrators to assume the role. Administrators won't have permission to terminate those instances; they must first assume a role. By using a role, administrators must take an additional step to assume a role before they can stop an instance that's critical to your organization.

Third Parties:

When third parties require access to your organization's AWS resources, you can use roles to delegate API access to them. For example, a third party might provide a service for managing your AWS resources. With IAM roles, you can grant these third parties access to your AWS resources without sharing your AWS security credentials. Instead, they can assume a role that you created to access your AWS resources.

Third parties must provide you the following information for you to create a role that they can assume:

- The AWS account ID that the third party's IAM users use to assume your role. You specify their AWS account ID when you define the trusted entity for the role.
- An external ID that the third party can associate you with your role. You specify the ID that was provided by the third party when you define the trusted entity for the role.
- The permissions that the third party requires in order to work with your AWS resources. You specify these permissions when defining the role's permission policy. This policy defines what actions they can take and what resources they can access.

After you create the role, you must share the role's Amazon Resource Name (ARN) with the third party. They require your role's ARN in order to assume the role.

Root AWS Accounts

We'll discuss this on the following slides.

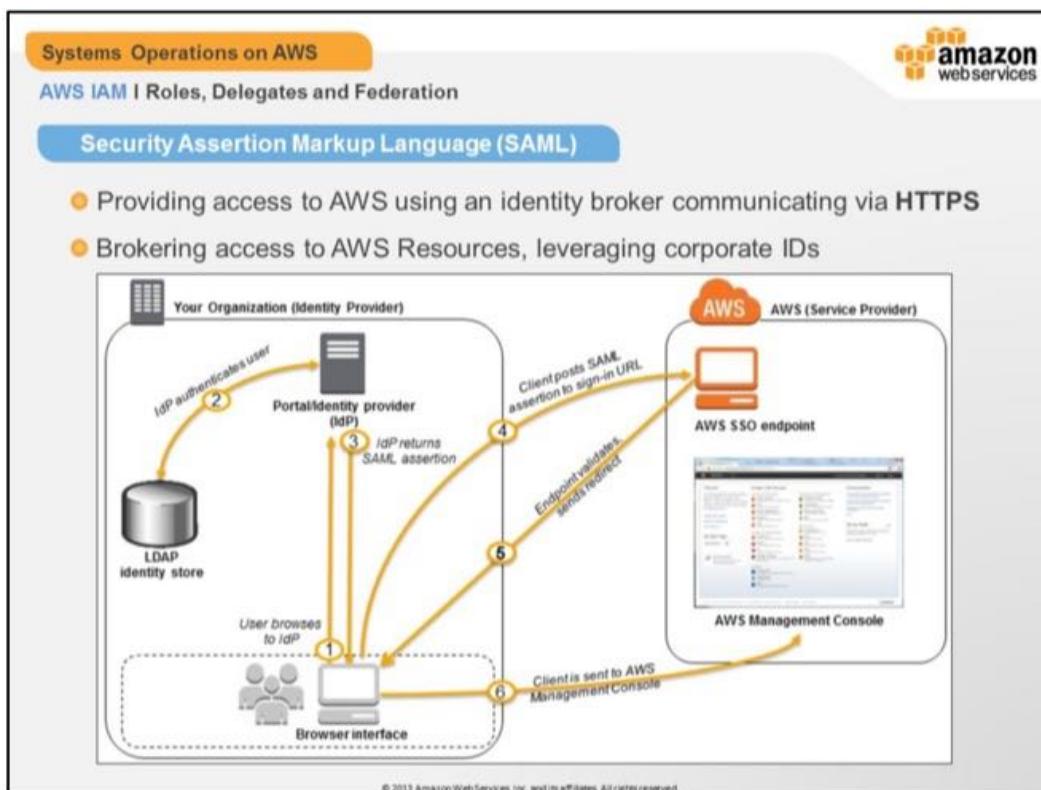
You can establish cross-account access in these ways:

- By using IAM roles. You define a role in Account A that can be assumed by a user in Account B. Using roles for cross-account access lets you grant access to any resource in Account A that is in a service that supports roles.
- For some AWS services, you can put a policy directly on the resource you want to share, and specify that another account has access to the resource. The resource you want to share must support resource-based policies.

Notes about the Identity Broker:

- Used to query STS
- Determines user from a web request
- Uses AWS credentials (service account) to authenticate to AWS
- Issues temporary security credentials to access AWS APIs (through STS)
- AWS permissions are configured by the administrator of the identity broker
- Configurable timeout; 1-36 hours
- Sample IIS authentication proxy code provided

here:<http://aws.amazon.com/code/1288653099190193>



SAML-based federation lets you provide access to AWS resources for users in an organization that uses [SAML](#) 2.0 (Security Assertion Markup Language 2.0) to exchange authentication and authorization information. To configure SAML-based federation in AWS, you create a role that determines which organization can access AWS (that is, assume an IAM role) and what federated users from that organization are allowed to do.

Before you create a role for SAML-based federation, you must create a SAML provider in IAM. For more information, see [SAML Providers](#).

The role-creation wizard in the IAM console provides two paths. One path is for creating a role for single sign-on (SSO) to the AWS Management Console. The other path is for creating a role that can be assumed programmatically. The following procedure describes both paths. The roles created by both paths are similar, but the path for SSO creates a role whose trust policy includes a condition that explicitly ensures that the SAML audience (aud attribute) is set to the AWS sign-in endpoint for SAML (<https://signin.aws.amazon.com/saml>).

Systems Operations on AWS

AWS IAM | Roles, Delegates and Federation

Web Identity Federation

- Useful for building and integrating your own applications
- Users log in using a public identity: Amazon.com, Facebook, or Google
- Users don't have to have IAM identities
- Temporary credentials last 15 min to 1 hour
- No backend credential services to maintain

The diagram shows a yellow rectangular box labeled 'Session'. Inside, there is a dashed-line box containing four items: 'Access Key Id' (green key icon), 'Secret Access Key' (red key icon), 'Session Token' (purple circle icon), and 'Expiration' (clock icon). To the right of the 'Session' box, a brace groups the four items inside the dashed box, with the text 'Temporary Security Credentials' written next to it.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Web identity federation enables you to create cloud-backed mobile and web apps that use public identity providers such as Facebook, Google, or the newly launched Login with Amazon service for authentication. With this new feature, you now have an easy way to integrate Amazon.com, Facebook, or Google sign-in into your apps without having to write any server-side code and without distributing long-term AWS security credentials with the app.

AssumeRoleWithWebIdentity:

Returns a set of temporary security credentials for users who have been authenticated in a mobile or web application with a web identity provider, such as Login with Amazon, Facebook, or Google. *AssumeRoleWithWebIdentity* is an API call that does not require the use of AWS security credentials. Therefore, you can distribute an application (for example, on mobile devices) that requests temporary security credentials without including long-term AWS credentials in the application or by deploying server-based proxy services that use long-term AWS credentials.

The temporary security credentials consist of an Access Key ID, a Secret Access Key, and a Session Token. Applications can use these temporary security credentials to sign calls to AWS service APIs. The credentials are valid for the duration that you specified when calling *AssumeRoleWithWebIdentity*, which can be from 900 seconds (15 minutes) to 3600 seconds (1 hour). By default, the temporary security credentials are valid for 1 hour.

The temporary security credentials that are returned from the `AssumeRoleWithWebIdentity` response have the permissions that are associated with the access policy of the role being assumed. You can further restrict the permissions of the temporary security credentials by passing a policy in the request. The resulting permissions are an intersection of the role's access policy and the policy that you passed. These policies and any applicable resource-based policies are evaluated when calls to AWS service APIs are made using the temporary security credentials.

Before your application can call `AssumeRoleWithWebIdentity`, you must have an identity token from a supported identity provider and create a role that the application can assume. The role that your application assumes must trust the identity provider that is associated with the identity token. In other words, the identity provider must be specified in the role's trust policy.

Systems Operations on AWS

AWS IAM | Roles, Delegates and Federation

Knowledge Check



1. Why is federation favored over replication?
2. Are IAM resource accounts required for each User who is provided temporary security credentials?
3. When one AWS root account trusts another, who is responsible for further delegating permissions to IAM users for resources in the trusting account?

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

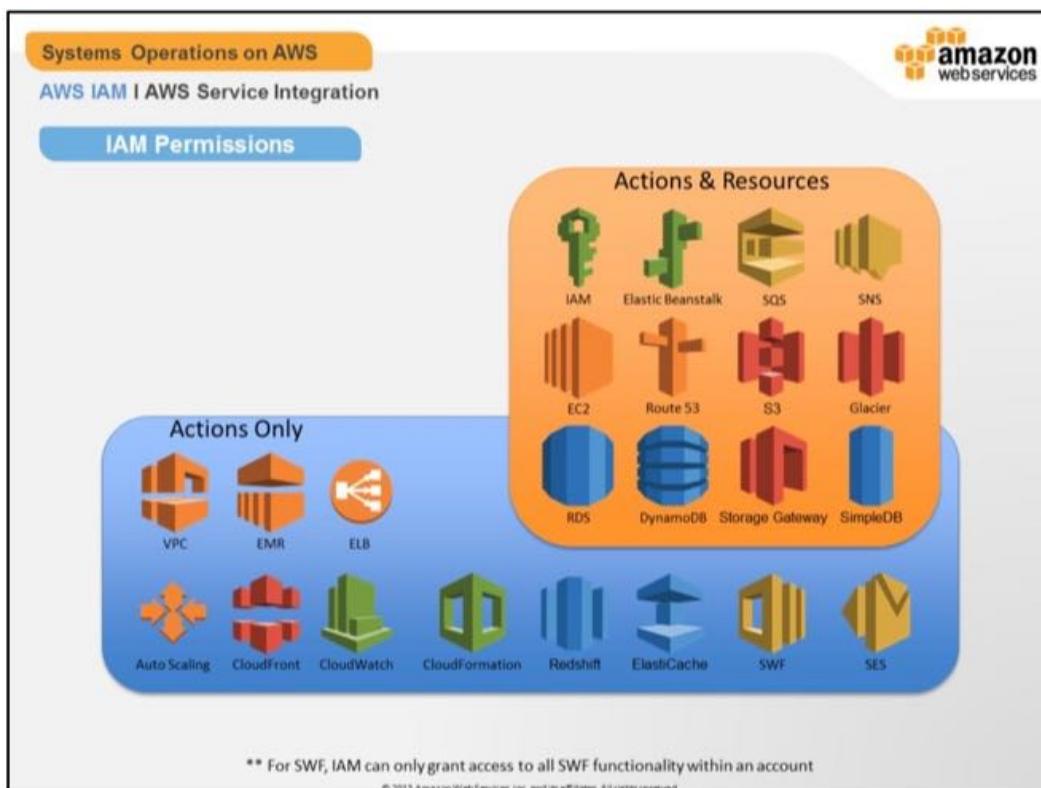
1. Replication copies credentials from one identity provider to another. This means that account management, such as deprovisioning, must be done in two places.
2. No. A service account is required for an identity broker to retrieve temporary security credentials via STS.
3. The administrator* of the trusted account.

*Note: An AWS root account or an IAM user with permissions to create roles may configure the delegation.



AWS IAM Service Integration

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.



This visual summarizes whether you can grant IAM permissions that control access to a service's actions, resources, or both. In this section, we will discuss the resource-level capabilities supported by many of the services as shown here.

Note that we are continually providing improvements to services, including deeper support of IAM. Check our website for the latest information at <http://aws.amazon.com>

Systems Operations on AWS

AWS IAM | Resource-Level Permissions | EC2

Resource Level Permissions – EC2

- Strictly control which IAM users or groups can start, stop, reboot, and terminate specific EC2 instances
 - Also control: EBS volumes, Images, and Elastic IPs
- Categorize your instances using tags
 - E.g. stack=prod, stack=dev, stack=test
 - Use Access Policy conditions to enforce permissions
- Restrict which users may apply/remove tags
- Helps enforce security principles
 - Least privilege
 - Separation of duties

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

With EC2-based resource permissions, customers can strictly control which IAM users or groups can start, stop, reboot, and terminate specific EC2 instances. This ability to assign control of an individual instance to a specific user or group helps organizations implement important security principles like separation of duties (preventing critical functions from being in the hands of one user) and least privilege (providing each user access only to the minimum resources they need to do their job).

For example, you probably don't want to give everyone in your organization permission to terminate business-critical production instances, so you can assign that privilege to only a few trusted administrators.

Systems Operations on AWS

AWS IAM | Resource-Level Permissions

EC2 Sample Policy

Allow actions on production environment in us-east-1

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Action": [  
                "ec2:StartInstances",  
                "ec2:StopInstances",  
                "ec2:RebootInstances",  
                "ec2:TerminateInstances"  
            ],  
            "Condition": {  
                "StringEquals": {  
                    "ec2:ResourceTag/stack": "prod"  
                }  
            },  
            "Resource": [  
                "arn:aws:ec2:us-east-1:123456789012:instance/*"  
            ],  
            "Effect": "Allow"  
        }  
    ]  
}
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Here is an example of how to use tags to control access to EC2 instances.

Because IAM policies are based on least privilege, users will not be able to manage your critical instances unless you give them permission to do so. To create a policy that defines permissions to start, stop, reboot, and terminate specific instances (e.g., the ones with a specific tag that you gave them, like “prod”), construct the policy as shown here. As you can see, this policy applies only to all instances within the *us-east-1* region.

As with other access policies, you are able to add more conditions to provide more granular control. You may want to require the use of multi-factor authentication within a time period (e.g. the last 15 minutes). You can also force users to come from a trusted source IP range when making the start, stop, reboot, or terminate requests.

Note: If you choose to use tags as a basis for setting permissions on instances, you will want to restrict which users have permissions to apply and remove tags. For EC2, you will want to restrict which users have permissions to use the *ec2:CreateTags* and *ec2:DeleteTags* actions, so that only these users will be able to change your instance inventory.

Systems Operations on AWS

AWS IAM | Resource-Level Permissions

Amazon Dynamo DB

- Control all DynamoDB actions
 - CreateTable, GetItem, DeleteItem, PutItem, etc.
- Example using policy variables

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": ["dynamodb:*"],  
            "Resource": "arn:aws:dynamodb:us-west-2:123456789:table/${aws:username}_ProductCatalog"  
        },  
        {  
            "Effect": "Allow",  
            "Action": ["dynamodb>ListTables", "dynamodb:DescribeTable", "cloudwatch:", "sns:"],  
            "Resource": "*"  
        }  
    ]  
}
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The Amazon DynamoDB API supports the following operations:

- BatchGetItem
- BatchWriteItem
- CreateTable
- DeleteItem
- DeleteTable
- DescribeTable
- GetItem
- ListTables
- PutItem
- Query
- Scan
- UpdateItem
- UpdateTable

Example

You can use policy variables to write a single policy and attach it to a group. You will need to create a group and, for this example, add multiple users, e.g. Alice and Bob, to the group. This example allows all Amazon DynamoDB actions on the \${aws:username}_ProductCatalog table. The placeholder policy variable \${aws:username} is replaced by the requester's user name when the policy is evaluated. For example, if Alice sends a request to add an item, the action is

allowed only if Alice is adding item to the Alice_ProductCatalog table.

Systems Operations on AWS

AWS IAM | Roles, Delegates and Federation

Knowledge Check



1. Resource-level permissions can help enforce which security best practices?
2. True/False: I can use EC2 resource-based permissions to restrict which groups of users can mount specific EBS volumes.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

1. Least privilege, separation of duties (or segregation of duties).
2. True.

Systems Operations on AWS

AWS IAM | AWS Security Resources

amazon
webservices

- AWS Site Links
 - <http://docs.aws.amazon.com>
- IAM Best Practices & Use Cases
 - <http://docs.aws.amazon.com/IAM/latest/UserGuide/IAMBestPracticesAndUseCases.html>
- Web Identity Federation Playgroun
 - <https://web-identity-federation-playground.s3.amazonaws.com/index.html>
- AWS Professional Services
 - <http://aws.amazon.com/enterprise-it>
- AWS Partner and Marketplace Solutions for Security
 - http://aws.amazon.com/search?searchQuery=security&searchPath=solution_providers&x=-915&y=-95
 - https://aws.amazon.com/marketplace/b/2649363011/ref=gtw_navlift_node_2649363011

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Module 4: Amazon Elastic Compute Cloud (EC2)

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The screenshot shows a slide titled "Amazon EC2 I Overview" under the "Systems Operations on AWS" category. The slide has a blue header bar with the title and a blue footer bar. The main content area contains a bulleted list of topics:

- Amazon Elastic Compute Cloud (EC2) Overview
- EC2 Technical Operations
- EC2 Networking and Security
- EC2 Storage Options
- EBS Performance Best Practices
- Elasticity
- Interfacing with EC2
- Lab

At the bottom right of the slide, there is an "amazon webservices" logo with a target icon.

Amazon Elastic Compute Cloud, otherwise known as EC2, sits at the core of AWS's compute platform. It allows for the provisioning of compute resources that you'd have access to in a traditional data center, hosted or virtual environment, but provides the additional flexibility of easily scaling up and down your compute infrastructure, and paying only for the compute power you need on an hourly basis.

This module will focus on:

- An overview of EC2, including **EC2 Instance attributes**, as well as **technologies** and **pricing options** unique to AWS customers.
- A more detailed look at EC2 technical operations, including **Protecting EC2 Instances** and **Bootstrapping**.
- We'll then take a more in depth look at details around **EC2 Networking** and **Security**,
- **EC2 Storage Options** and the **Performance Best Practices** surrounding storage.
- We'll go through a brief introduction to the elasticity features available in an AWS environment, such as Amazon Elastic Load Balancing (ELB) and Amazon Auto Scaling.
- Finally we'll go through a brief review of the different ways to **interface with EC2** which will lead us to a hands-on lab where we'll employ the skills we learned in this module to build a highly performant database server that will be used throughout the rest of this course.



Systems Operation on AWS

EC2 Overview

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Amazon EC2 I Overview

Key Terms

Term	Description
Amazon Elastic Cloud Compute (EC2) Instance	A virtual server instance running within AWS
Amazon Machine Image (AMI)	A launch configuration for an EC2 instance
EC2 Instance Storage	Ephemeral local storage available to EC2 Instances
Amazon Regions and Availability Zones (AZs)	Physical locations within a given region, allowing for increased fault tolerance within the AWS infrastructure.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Before we dive any deeper, let's quickly cover a couple of the key terms we'll be discussing over the next few minutes.

- Amazon Elastic Cloud Compute, or EC2 instance can be viewed as a virtual server running within your AWS environment.
- An Amazon Machine image, otherwise known as an AMI, is essentially a launch configuration for an EC2 instance, including the operating system, installed software, configuration of the operating environment and metadata associated with the EC2 instance. Some might think of this as a 'Golden Image' or template for virtual machines, or in this case, EC2 instances.
- Instance Storage is local, ephemeral storage available to your EC2 Instances. We'll dive into this in much more detail in the Storage Options section
- Finally Amazon Regions and Availability Zones are the keys to fault tolerant and localized deployments. Availability Zones exist within Regions, which there are currently 9 of world-wide. Each Region will have at least 2 Availability Zones which are separate physical locations within a given region, allowing for increased fault tolerance within the AWS infrastructure.

Systems Operations on AWS

Amazon EC2 I Overview

EC2 Key Concepts

Amazon Elastic Compute Cloud (EC2)

- EC2 instance = virtual server
- Provision and resize compute capacity in **16+ instance types**
- Reduces the time required to obtain and boot new server instances to minutes or seconds
- Scale capacity as requirements change; Only pay for capacity that you actually use
- Deploy across Availability Zones and Regions
- Windows, Linux; RHEL, SLES, Amazon Linux, etc.
- Programmatic control of infrastructure
- i-xxxxxxxx identifier



Amazon EC2

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- As we previously discussed, at a very high level EC2 instances can be thought of as virtual servers that exist within your AWS infrastructure.
- However when you begin working with EC2, you'll quickly realize that there's much more to EC2 instances than simply virtual servers, such as the ability to programmatically or manually provision massive amounts of compute capacity within moments, and the ability to resize EC2 instances to meet changing technical needs.
- As with all AWS services, you only pay for the capacity that you are actually using, and again, can quickly scale that capacity up and down in order to reduce compute contention during peak periods, and reduce cost during periods of lower utilization. You'll have the ability to even completely turn off infrastructure when it's not needed, and reduce your costs to \$0, something that just isn't possible anywhere other than AWS.
- Within an EC2 instance, there's great flexibility to run the Operating System of your choice, including several different flavors of enterprise-grade Linux, different versions of Microsoft Windows, and even our own Amazon Linux.
- With EC2, you'll also find new and unique ways to operate your infrastructure such as the ability to programmatically remap IP addresses, network interfaces and other resources associated with EC2 Instances.

- Finally, every resource in AWS has a unique identifier, and in the case of EC2 instances, this will be in the form of i dash, followed by an 8 character unique identifier



Systems Operations on AWS

Amazon EC2 I Overview

EC2 Instance Attributes

- Memory, compute Units, I/O performance and storage
- 16+ Instance types available, and growing

Name	Memory	Compute Units	Virtual Cores	Instance Store Volumes*	Architecture	Network Performance	Available for Spot Instance	API Name
Cluster Compute								
Cluster Compute Eight Extra Large	60.5 GiB	88	16 (2 x Intel Xeon E5-2670, eight-core with hyperthread)	3360 GB (4 x 840 GB)	64-bit	Very high (10 Gbps Ethernet)	Yes	cc2.8xlarge
Cluster Compute Quadruple Extra Large	22.5 GiB	33.5	8 (2 x Intel Xeon X5570, quad-core with hyperthread)	1680 GB (2 x 840 GB)	64-bit	Very high (10 Gbps Ethernet)	Yes	cc1.4xlarge
Cluster GPU								
Cluster GPU Quadruple Extra Large**	22.5 GiB	33.5	8 (2 x Intel Xeon X5570, quad-core with hyperthread), plus 2 NVIDIA Tesla M2050 GPUs	1680 GB (2 x 840 GB)	64-bit	Very high (10 Gbps Ethernet)	Yes	cg1.4xlarge
High CPU								
High-CPU Extra Large	7 GiB	20	8 (with 2.5 ECUs each)	1680 GB (4 x 420 GB)	64-bit	High	Yes	c1.xlarge

* © 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

EC2 Instances can be thought of as programmatic virtual machines. As such, much of the same terminology that you're accustom to, such as Memory and Virtual Cores are applicable when discussing resources available to EC2 instances.

Beyond that however, there are a few additional characteristics that you should be aware of.

- **EC2 Compute Units**, otherwise known as ECUs are a way to help better understand the compute characteristics of a given EC2 instance. Because Amazon EC2 is built on commodity hardware, over time there may be several different types of physical hardware underlying EC2 instances. The amount of CPU that is allocated to a particular instance is expressed in terms of these EC2 Compute Units. One EC2 Compute Unit provides the equivalent CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor.
- In the case of our Cluster compute and GPU instances, we are very specific about the processor types available on these instances.
- Instance Store volumes, which we'll get into in much more detail later on, is the local on-board storage available to an Instance. The amount of Instance storage available scales with the other resources available to an instance.
- Like Instance storage, network performance also scales with other resources

available to instances, and is measured in terms of low to Very High throughput. While network throughput is rarely a source of contention, you should do the proper benchmarking to ensure the instance type you have chosen is appropriate for the expected network IO.

- Finally, as AWS is a programmatic infrastructure, all EC2 instance types can be referred to by their API name, such as `cg1.4xlarge`, which translates to Cluster GPU, or `t1.micro`, for our smallest available instance, which has about the same resources available as your favorite smartphone.

Unlike a physical server or other Virtual Machine that you may have managed in the past, EC2 instances are static in terms of the attributes mentioned above. If you need more computing power or additional memory, you can easily move between different instance types.

Systems Operations on AWS

Amazon EC2 I Overview

9 Instance Families

- Grouped based on targeted applications
- Flexible, combinable and easy to change type

Family	Description
Cluster Compute	<ul style="list-style-type: none"> • Large amount of CPU coupled with increased networking performance. • Designed for High Performance Compute (HPC) and other network-bound applications.
Cluster GPU	<ul style="list-style-type: none"> • General-purpose graphics processing units (GPUs) • Well-suited for HPC applications as well as rendering and media processing applications.
High CPU	<ul style="list-style-type: none"> • Have proportionally more CPU resources than memory (RAM). • Well-suited for compute-intensive applications.
High I/O	<ul style="list-style-type: none"> • Provide tens of thousands of low-latency, random I/O operations per second (IOPS) to an application. • Well-suited for NoSQL databases, clustered databases, etc.
High Memory	<ul style="list-style-type: none"> • Proportionally more memory resources. • Well-suited for high-throughput applications, such as database and memory caching applications.
High-Memory Cluster	<ul style="list-style-type: none"> • Large amounts of memory coupled with high CPU and network performance. • Well-suited for in-memory analytics, graph analysis, and scientific computing applications.
High Storage	<ul style="list-style-type: none"> • Very high storage density and high sequential read and write performance per instance. • Well-suited for data warehousing, Hadoop/MapReduce, and parallel file systems.
Micro	<ul style="list-style-type: none"> • Small amount of consistent CPU resources with ability to burst when additional cycles are available. • Well-suited for lower throughput applications that consume significant compute cycles periodically
Standard	<ul style="list-style-type: none"> • Have memory-to-CPU ratios suitable for most general-purpose applications.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

To follow on from the last slide, the 16+ available instances types that we offer are divided into 9 instance families **grouped together based on their targeted applications**. The instance families each contain at least 1 instance type that is built for a specific use case.

(Read through list of available families, choosing at least 1 bullet point from each category)

That's not to say that you can't use an Instance in the Cluster Compute for a database, or any other application of your liking. It does however show that AWS is well suited for most any workload imaginable.

As we mentioned before, it's very easy to migrate to another instance type or even family, and can usually be performed with minimal downtime. An example might be creating an AMI from your current instance, launching a new instance from the AMI in the desired instance family and type, and then flipping EIP and/or reassigning ENI. If your applications logic is such that it will retry and a short network failure, this can be completed with no downtime.

Systems Operations on AWS

Amazon EC2 I Overview

On-Demand Instances	Reserved Instances	Spot Instances
<p>Unix/Linux instances start at \$0.08/hour USD in the US East Region</p> <p>Pay as you go for compute power</p> <p>Benefit:</p> <ul style="list-style-type: none"> Low cost and flexibility <p>Pay only for what you use, no up-front commitments or long-term contracts</p> <p>Use Cases:</p> <ul style="list-style-type: none"> Applications with short term, spiky, or unpredictable workloads; Application development or testing Initial POC's and benchmarking before committing. 	<p>1 or 3-year terms</p> <p>Light/Medium/Heavy instance types</p> <p>Pay low up-front fee, receive significant hourly discount</p> <p>Benefit:</p> <ul style="list-style-type: none"> Cost / Predictability <p>Helps ensure compute capacity is available when needed</p> <p>Use Cases:</p> <ul style="list-style-type: none"> Applications with steady state or predictable usage Applications that require reserved capacity Users able to make upfront payments to reduce total computing costs even further 	<p>Bid on unused EC2 capacity</p> <p>Spot Price based on supply/demand, determined automatically</p> <p>Benefit:</p> <ul style="list-style-type: none"> Cost / Large Scale, dynamic workload handling <p>Spot Price below bid, instances start</p> <p>Spot Price above bid, instances stop</p> <p>Use Cases:</p> <ul style="list-style-type: none"> Applications with flexible start and end times Applications only feasible at very low cost Users with needs for short term, large amounts of additional capacity

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



There are a couple of different pricing options with EC2, and depending on how long you need the resources, and the applications ability to handle instance termination, that can allow you to save significantly on your hourly costs;

- **On-Demand instances** are where we recommend most customers start. This allows customers to get working with EC2 with no up front commitment and operates in a utility style pricing model where you can simply turn off the instance to stop paying for resources.
- **Reserved Instances** are where customers can realize significant savings over traditional infrastructure, all while helping to ensure capacity is available when needed. There are several pricing models within Reserved Instances and tools to help you understand the most cost effective model for your needs. An important note on Reserved Instance flexibility here, in that Reserved instances are not tied to a specific running instance, rather they are like a coupon. You do not need to launch an instance as a Reserved Instance.
- One very unique offering available to EC2 users is our **Spot Instances**, where customers can bid on excess AWS capacity in order to run short term jobs that need large amounts of compute capacity, or have applications that are tolerable to flexible start and stop times. Spot instances are a great in many cases, but you must ensure that your application can handle instance termination at any point in which the current market price exceeds your spot bid.

Systems Operations on AWS

Amazon EC2 I Overview

Amazon Machine Image (AMI)

- Template containing software configuration
- Region specific
- AMI publishers:
 - Amazon
 - Self
 - Community
 - Marketplace
- Copy AMIs to other Regions
- ami-xxxxxxxx identifier

Amazon Machine Image

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Lastly in the EC2 overview section, let's briefly talk about Amazon Machine Images, or AHM-EE's.

- Just as an EC2 instance is to a virtual machine or server, an AMI is to a Virtual Machine Template or Image. Even more accurately, you can describe an AMI as a launch configuration which contains all of OS, installed software, configurations and other metadata associated with an EC2 instance.
- AMI's are regions specific, which allow you to launch similar EC2 instances within multiple availability zones. Some customers however want the ability to treat AMI's as 'Golden Images' and will often use our AMI copy feature to move AMI's between regions.
- Most customers start with publicly available AMI's, such as those provided by Amazon Web Services. These will generally be somewhat general in their configurations to allow for the end-user to customize to their applications needs. Some AMI's will also be more purpose built, and may have applications such as Microsoft SQL server already bundled in, along with the licensing costs for SQL server built into the hourly fee.
- Using a Public AMI as a starting point, you also have the ability to customize to your specific needs, installing applications and configurations specific to your environment, expanding the volume space, etc., at which point you can then

create your own AMI. This is helpful for situations where you wish to minimize the bootstrap process, improve boot times, or overall simplify this portion of your infrastructure.

- Sometimes, you'll have created your own AMI and wish to share it with other customers or another account for a variety of reasons. This is as simple as adding the target account ID to the AMI attributes. Remember, AMI's are Region specific.
- You can also use the AWS marketplace to find paid AMI's. Paid AMI's will often provide additional functionality from 3rd party vendors, such as the ability to run purpose built software packages such as enterprise ready Load Balancers. In other environments, these are often referred to as virtual appliances.
- Finally, AMI's have an identifier starting with ami- followed by 8 unique identifying digits....



Amazon EC2 Technical Operations

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Amazon EC2 | EC2 Technical Operations

Managing EC2 Instances

- Key Pairs
 - Public-key cryptography
 - Amazon-generated or imported
 - Region specific
 - Used to decrypt Windows password
- Termination Protection
 - Prevent accidental termination
 - Can enable at any time
 - Not applicable if in Auto Scaling group
- Tagging EC2 Instances
 - Discoverability, cost allocation and resource-level permissions
- Use existing tools

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Managing your EC2 infrastructure properly is critical to success in AWS. While standard infrastructure best practices absolutely apply, there are some additional tools in AWS to help make managing your infrastructure easier.

- By default, initial access to any EC2 instances is controlled by Key Pairs as opposed to username and password credentials. Key pairs consist of a public key stored on the EC2 instance, and a private key that only you should have access to. This key pair can be generated by Amazon or you can create your own key pair and import this into an AWS region. The key pair is generated or imported prior to launching your first instance, and the private key can only be downloaded 1 time if generated by Amazon. Because of this, it's very important to keep your key pair in a safe location, but one that is accessible to you, as AWS cannot help you recover this key to login to your instances. In the case of Windows, Microsoft only allows username and password credentials to authenticate, however we still make use of the keypair to decrypt the Windows password. After your first login, you should follow normal company best practices, which may include disabling the administrator, creating a new user and using a strong password.
- In AWS, you're able to terminate instances just as easily as you can launch instances. While this allows for great programmatic flexibility in managing your environment, this can also lead to lost data or unintended interruptions in your application. Termination protection is a flag that can be enabled on any EC2

instance which helps prevent this. Termination Protection can be enabled at launch, or while an instance is running, and introduces an additional layer of protection which must be disabled before terminating an instance. 1 note here; termination protection will be ignored if the instance is being managed by an Auto Scaling group. We'll talk more about Auto Scaling groups later.

- As you begin to build your AWS infrastructure, you may quickly find that your infrastructure becomes confusing, and difficult to understand what resources are performing what business actions. The ability to tag AWS resources, and in this case EC2 instances is paramount to discoverability and cost allocation. Additionally, you now have the ability to set resource-level permissions by leveraging tagging and IAM users. We'll introduce Identity and access management (IAM) in a bit more detail later in this module.
- Finally, while we won't touch on anything specific, I'd like to point out the fact that nearly any tool or process that you currently use today to manage your compute infrastructure should also apply in EC2. Whether this is simply tracking resources in an Excel sheet, or integrating 3rd party automation and deployment tools, you'll be able to manage your EC2 infrastructure much as you do today.

Systems Operations on AWS
Amazon EC2 | EC2 Technical Operations

Bootstrapping EC2 Instances

- Metadata
 - Data about the Instance, accessible within the Instance; **Who Am I?**
 - AMI ID, hostname, IPv4 address, available public keys, etc.
 - *PROMPT> GET http://169.254.169.254/latest/meta-data/*
- User Data
 - User supplied data, accessible within the Instance; **What should I do?**
 - Script to run, files to download from S3, etc.
 - Limited to 16 KB, in raw form

● Metadata + User Data = Self configuration

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

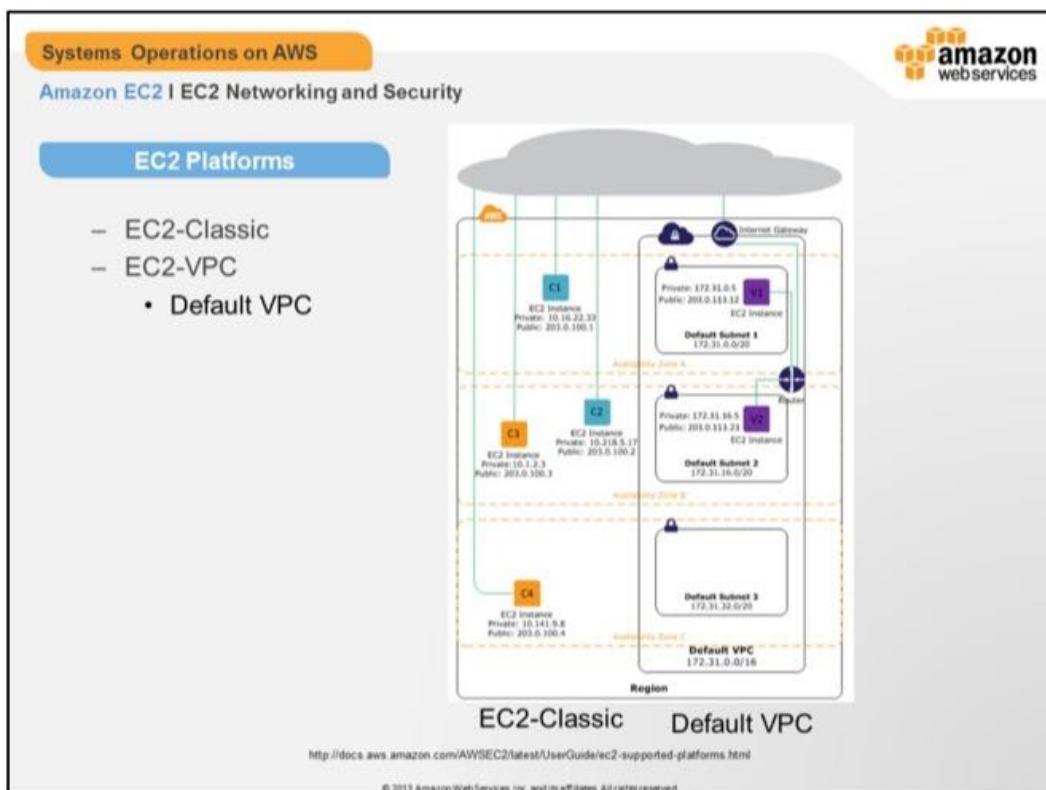
Just as you can bootstrap servers and virtual machines in a traditional infrastructure, you also have this ability in AWS, but with increased agility and flexibility by using some features specific to EC2.

- Every EC2 instance will have access to both Metadata and User Data. Metadata is information about the specific instance, available locally on the instance, that enables you to manually or programmatically ask **Who am I?** In other words, you can get information about the AMI ID and Key pair used to launch the specific instance, the DNS name, IP addresses and other information that might be helpful in order to more easily operate and manage your EC2 instances. This is available by opening a browser or using a console to GET or Curl the 169.254.159.254 endpoint.
- User data is information and/or instructions that you the operator can provide to the instance at boot time. This might be everything from a script to run at launch, or files that should be downloaded from S3 and placed in the users folder for future operations. Keep in mind that user data is limited to 16KB in raw form.
- Together, metadata and user data can be used for self configuration, allowing the EC2 instance to answer the question **“OK, who am I, and what should I be doing”**. As you become more comfortable with operating EC2, you'll find these features extremely helpful in automating your infrastructure. We'll be demonstrating this in more detail later in the lab.



EC2 Networking and Security

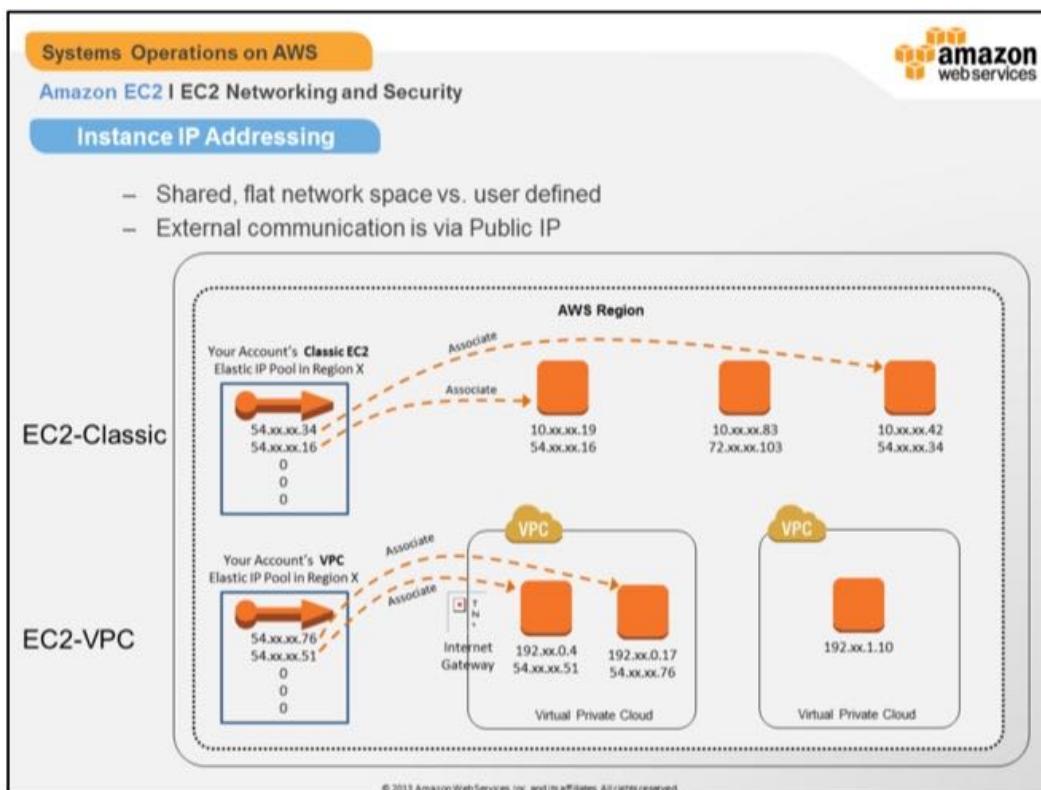
© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.



So far, we've discussed EC2 as a service independent of a platform. The reality is that EC2 as a service can run on 1 of 2 platforms in terms of the network infrastructure.

- EC2-Classic is original platform for EC2, and was the only option available until around the 201x timeframe. EC2-Classic is a flat network space, where all customers within a region run in the same 10.x.x.x/8 network. This meant that customers and instances in other accounts could easily communicate with each other by allowing access via Security Group (firewall) rules. While this is very useful for many customers, this also gave some customers the feeling that AWS was not as secure as it could be, and also broke the traditional model that network administrators were familiar with in terms of placing servers in specific vLAN's and DMZ's.
- More recently, AWS has launched VPC, which provides increased manageability in terms of private networking space, subnets, routing tables, network ACL's and improved Security Groups where you can run your EC2 instances in a more similar model to that of a traditional infrastructure. In-fact, with the right planning in place, VPC can be setup to be an extension of your current datacenter and/or users, and none will ever be the wiser that this is running within AWS.
- Today, all new accounts are launched into a VPC for the greater control and private networking space benefits. Some customers however want the ease of

use of EC2-Classic. In order to provide this ease of use, we've launched VPC by default or Default VPC. Unless you specifically build a VPC, upon launching your first EC2 instance, a VPC infrastructure will be created in the background that you should never need to manage any further.



- Within EC2-Classic or a VPC, you can communicate between resources using private IP addresses and internal DNS which is associated with an instance at launch time. In EC2-Classic, this IP address will fall in the 10.x range, which is the flat network space shared by all AWS customers. In VPC, this is a user defined IP space that is only accessible to resources within the same VPC. Even if you have multiple VPC's in the same region using the 192.168.0.0 network space, these resources cannot communicate using internal networking.
- In order to communicate with external resources you must use a Public IP address or external DNS name. This includes resources hosted in your datacenters, customers remote locations, and also other AWS services, such as Amazon Simple Storage Service (S3). In EC2-Classic and default VPC's, a public IP is allocated and associated when an instance is launched. This will remain with the instance until it is stopped and/or terminated. In EC2 Classic, when you stop/start your instance both your public and private IP addresses will change.
- In a non-default VPC/subnet, instances will not be given a public IP address and in order to communicate externally with these, you'll need to request and associate an Elastic IP Address. EIP's can be associated with any instance, whether they are running in a VPC or not, and will allow you to retain a consistent public IP address associated with your Instance. Like all other IP addresses, private and public, there is a DNS name associated with your EIP that can also

be referenced.

Systems Operations on AWS

Amazon EC2 | EC2 Networking and Security

Security Groups (SGs)

- Controls traffic allowed to EC2 Instances
- Can assign multiple SG's to EC2 Instance
- EC2-Classic only:
 - Ingress only
 - Can only specify SG associations at EC2 Instance launch

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-network-security.html>

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Briefly mentioned a few slides ago, Security Groups are essentially stateful firewalls that surround individual EC2 instances, allowing you to control the traffic allowed to pass to the instance. It's important to understand that Security Groups are applied to specific instances rather than at the entry point to your network, which can help increase security and be more granular with granting access.

You can assign multiple Security Groups to an instance, such as an admin SG allowing port 22, and a database server SG allowing port 3306. You can also apply a security group to multiple instances, such as a Web server SG, allowing port 80 to all the web servers in your network.

As mentioned before, instances running on the VPC platform allow improved Security Group control allowing you to apply rules to egress traffic, as opposed to only ingress in EC2-Classic. Another limitation of EC2-Classic is that SG associations can only be applied at the launch of an EC2 instance, and must be terminated and re-launched to change membership. In VPC, you can change membership at anytime. You can however modify SG rules at any time, and these changes will be pushed to your EC2 instances



EC2 Storage Options

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Amazon EC2 | EC2 Storage Options

Types of Storage

- Amazon Elastic Block Store (EBS) Volume
 - Persistent, network attached, block level storage volumes used by EC2
 - IOPS needs can be provisioned
- Amazon EBS Snapshot
 - A point-in-time copy of an EBS volume stored in S3
- Instance Storage
 - Ephemeral local storage
- Root device and Secondary Volumes
- Block Device Mappings
- Amazon Simple Storage Service (S3)
 - Unlimited object storage accessible anywhere
- EBS Optimized Instances

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

A few key terms that we'll be discussing in this portion are;

- Amazon Elastic Block Store, or EBS volumes. These are persistent, network attached block level storage volumes that can be attached and mounted to EC2 instances.
- An EBS snapshot allows you to take point-in-time, incremental backups of your EBS volumes and store this in S3
- Instance storage is ephemeral, or non-persistent, storage that is locally attached to EC2 instances as opposed to over a network connection.
- We'll talk about what to use for Root and secondary volumes, and how EC2 instances work with block device mappings.
- Amazon Simple Storage Service or S3 is an object storage accessible anywhere in the world, which allows you to manually storage objects or blobs. It provides eleven 9's of durability by default, allows the ability to easily use server side encryption, and set lifecycle policies to manage objects.
- Finally, we'll touch on what EBS optimized instances are and other ways to optimize your storage systems for performance in the next module.



Elasticity

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

In the last 2 sections, we'll briefly touch on Elasticity and ways to interface with EC2. Some of this will be covered elsewhere throughout the course, but we want to make sure we complete the circle in regards to EC2, and understand how to build a scalable system in AWS.

Systems Operations on AWS

Amazon EC2 | EC2 Elasticity

Amazon Elastic Load Balancing (ELB)

- Distributes requests among backend Instances
- Multi-AZ deployments
 - Automatically routes traffic away from failed AZ
- Health monitoring of backend EC2 Instances
- Support for SSL, sticky sessions, IPv6, etc.
- Monitoring of request count, request latency, etc.
- Available in VPC
- Integrates closely with Auto Scaling



Elastic Load
Balancing

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Amazon Elastic Load Balancing, is a managed, distributed load balancing service available within the EC2 realm.

- This helps distribute requests among backend instances, which can be web instances, application instances, or any other instances that can operate in a fairly stateless manner.
- ELB works hand-in-hand with multi-Availability Zone deployments, and can automatically route traffic away from failed AZ's.
- ELB monitors instances registered behind it, and will direct traffic away from instances that are not capable of receiving or returning traffic.
- It has support for SSL, sticky sessions, IPv6 and more. 1 exception to this is IPv6, which is not available within a VPC.
- While in EC2-Classic, you can only use ELB's as an external facing component, in VPC, these can be setup to distribute traffic internally as well, such as between your web instances and app instances.

Systems Operations on AWS

Amazon EC2 | EC2 Elasticity

Amazon Auto Scaling

- Scale in and out on demand
- Replace failed instances automatically
- Integrates well with other AWS services
- Leverage with AWS Spot Instances for cost optimization
- No additional charge

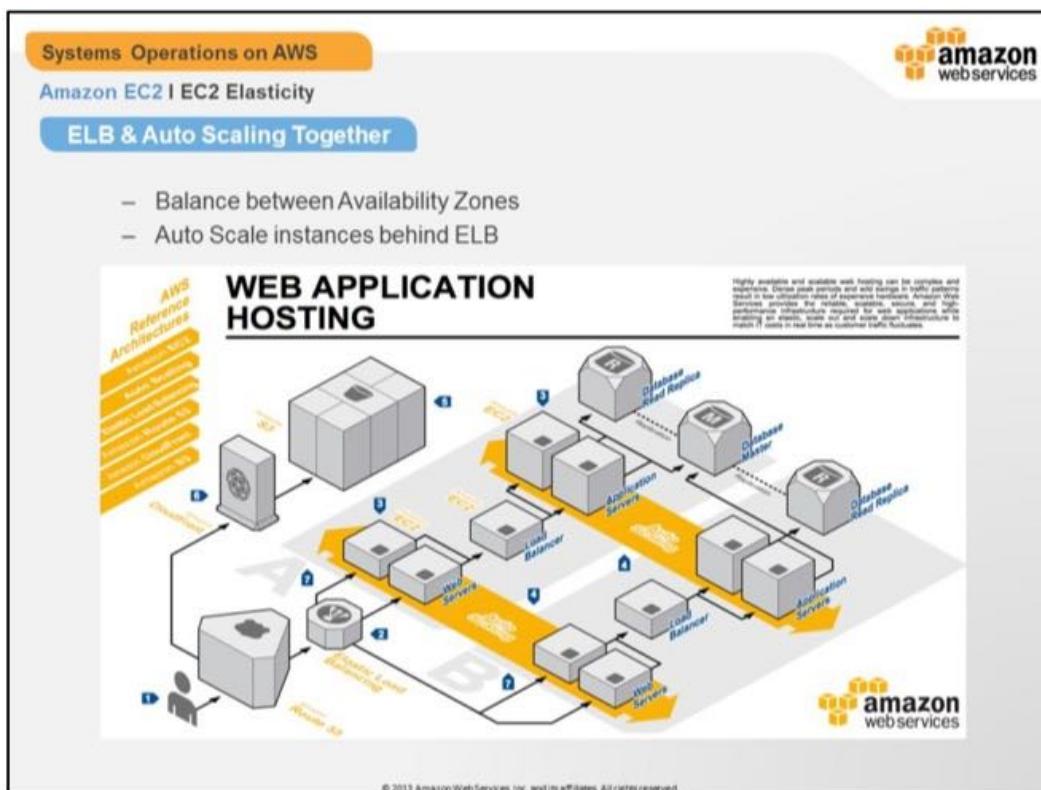


Auto Scaling

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Amazon Auto Scaling, is one of the core features of AWS that allows customers to build highly scalable, automated infrastructures in AWS.

- Using auto scaling, you can define policies that allow your application tiers to scale in and out as demand increases and subsides. Auto Scaling does this by integrating closely with CloudWatch, and also goes hand-in-hand with ELB, in order to allow your applications to scale in an elastic manner with load, just as your ELB will.
- Customers can also make use Auto Scaling without taking advantage of any actual scaling at all. Because Auto Scaling also monitors the health of your backend instances, you can use Auto Scaling to simply monitor for failed instances, and launch replacement instances without any manual intervention.
- Auto Scaling also integrates with Spot instances, which can allow you to take advantages of very low cost ways to scale your infrastructure up to handle increased demand, but only when the low cost permits.
- Auto Scaling is a completely free feature available to AWS customers, and you only pay for the resources that are created by your Auto Scaling groups.



Using Elastic Load Balancing along with Auto Scaling your EC2 instances, you can start to imagine how an infrastructure such as this one is possible. This specific diagram is one of the reference architectures listed at the AWS Architecture Center. While we're using many AWS services in this architecture, we can see the use of an Elastic Load Balancer on the front end, balancing traffic to EC2 instances in an Auto Scaling group, which spans multiple Availability Zones.

On the backend, the Application servers are also in an Auto Scaling group, but are receiving balanced traffic from the Web servers via a custom built Load balancer. If this architecture were running within a VPC, we'd actually be able to use an internal Elastic Load Balancer here as well.



Interfacing with EC2

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The screenshot shows the AWS Management Console with the 'Create a New Instance' wizard open. The instance type is set to 'Amazon Linux AMI 2013.09.4 (x86_64-v20130904)'. The configuration includes a Name: 'AWS-Test', Security Group: 'AWS-Test', and an IAM Role: 'AmazonEC2FullAccess'. The instance is being launched into a VPC with a Subnet ID: 'subnet-00000000'. The instance type is 'm3.xlarge' with 8 GiB of memory and 4 cores. The key pair is 'AWS-Test'. The volume type is 'Standard'. The availability zone is 'us-east-1a'. The root volume is 'Standard' with a size of 20 GiB. The root device name is '/dev/sda1'. The instance will be started automatically. The 'Next Step' button is visible at the bottom.

Tools for Windows PowerShell

- Management Console (Desktop and Mobile)
- EC2 CLI
- Unified AWS CLI
- SDK's
- PowerShell
- 3rd Party

Java **Python** **Ruby**

CLI

```
# aws ec2 describe-instances
=
# aws ec2 start-instances --instance-ids i-12345678
=
# aws sns publish --topic-arn arn:aws:sns:us-east-1:123456789012:OperationalError
--message "Script Failure"
=
# aws sqs receive-message --queue-url https://queue.amazonaws.com/123456789012/Test
```

There are multiple ways to interact with AWS, and in this case specifically, EC2.

- While most people are aware of the Desktop Management console, we also have a mobile app version available for most popular mobile devices.
- Along with other AWS services, EC2 has its own Command Line interface that can be installed, however going forward, the recommended tool for use is the Unified AWS CLI, which covers many AWS services, and is available for Windows, Linux, Mac, etc.
- We also have several SDK's available for our developer community, including the most popular development languages such as Java and Ruby, as well as PowerShell commandlets for our Windows community.
- Finally, there's many 3rd party options available from companies such as rightscale, which allow you to manage portions of your AWS infrastructure, as well as other vendors. Additionally, there's tools such as Asgard, the open sourced UI that Netflix uses to manage their environment.



Lab

Amazon EC2

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS
Amazon EC2 I Lab Exercise
Lab Objectives

In the following lab you will be building a high speed storage system for a database server.

- Configure the AWS CLI
- Create an EC2 Instance
- Use the EC2 Meta-data service
- Create EBS volumes
- Create multiple AMI's
- Stripe EBS Volumes
- Create Snapshots

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

In the following lab you will be building a high speed storage system for a database server.

- Configure the AWS CLI
- Create an EC2 Instance
- Use the EC2 Meta-data service
- Create EBS volumes
- Create multiple AMI's
- Stripe EBS Volumes
- Create Snapshots



Module 5: Elastic Block Store (EBS)

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Amazon EC2 I Overview

What We Will Cover

- EC2 storage options
- EC2 root devices and how they relate to Amazon Machine Images (AMIs)
- Volume Striping

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Amazon EBS allows you to create storage volumes and attach them to Amazon EC2 instances. Once attached, you can create a file system on top of these volumes, run a database, or use them in any other way you would use a block device. Amazon EBS volumes are placed in a specific [Availability Zone](#), where they are automatically replicated to protect you from the failure of a single component.

Amazon EBS provides two volume types: *Standard volumes* and *Provisioned IOPS volumes*. Standard and Provisioned IOPS volumes differ in performance characteristics and price, allowing you to tailor storage performance and cost to the needs of your applications.

This module will focus on:

- EC2 storage options
- EC2 root devices and how they relate to Amazon Machine Images (AMIs)
- Volume Striping



EC2 Storage Options

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Amazon EC2 | EC2 Storage Options

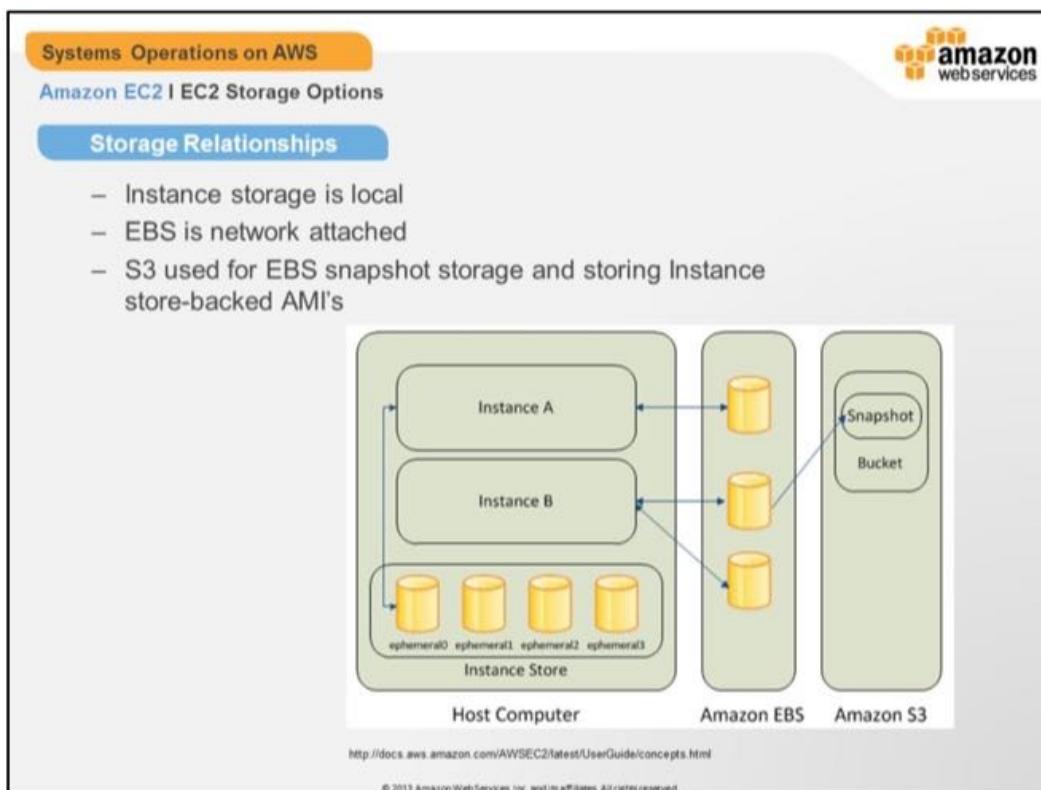
Types of Storage

- Amazon Elastic Block Store (EBS) Volume
 - Persistent, network attached, block level storage volumes used by EC2
 - IOPS needs can be provisioned
- Amazon EBS Snapshot
 - A point-in-time copy of an EBS volume stored in S3
- Instance Storage
 - Ephemeral local storage
- Root device and Secondary Volumes
- Block Device Mappings
- Amazon Simple Storage Service (S3)
 - Unlimited object storage accessible anywhere
- EBS Optimized Instances

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

A few key terms that we'll be discussing in this portion are;

- Amazon Elastic Block Store, or EBS volumes. These are persistent, network attached block level storage volumes that can be attached and mounted to EC2 instances.
- An EBS snapshot allows you to take point-in-time, incremental backups of your EBS volumes and store this in S3
- Instance storage is ephemeral, or non-persistent, storage that is locally attached to EC2 instances as opposed to over a network connection.
- We'll talk about what to use for Root and secondary volumes, and how EC2 instances work with block device mappings.
- Amazon Simple Storage Service or S3 is an object storage accessible anywhere in the world, which allows you to manually storage objects or blobs. It provides eleven 9's of durability by default, allows the ability to easily use server side encryption, and set lifecycle policies to manage objects.
- Finally, we'll touch on what EBS optimized instances are and other ways to optimize your storage systems for performance.



We'll go into each storage type in much more detail, but this image shows a good overview of where each storage type resides in relation to your EC2 Instances. From left to right:

- Instance storage is local to the host that your EC2 instance is running on. This prevents any network latencies when working with data. We'll get into more specifics of Instance storage on the next slide.
- In the middle we see EBS, which shows the physical separation from the EC2 instance. EBS volumes are written to and read from over a network connection.
- On the right we see Amazon S3, and in the case of EC2, this is used to snapshot EBS volumes, but also can be used to host "instance store-backed" AMI's, which we'll talk more about shortly.

Systems Operations on AWS
Amazon EC2 | EC2 Storage Options

Instance Storage

- Temporary (ephemeral), block-level storage
- Shared disk subsystem
- Free; 160 GB – 49 TB
- Root or secondary
- Good for:
 - Buffers, Cache, Scratch, etc.
 - Non-persistent data
- Not so good for:
 - Persistent Database
 - Valuable, long-term data

Host Computer 1

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/InstanceStorage.html#instance-storage-concepts>

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

As we mentioned, Instance storage is local to the host that your instance is running on. As such, network latencies when working with data are not present.

- The main take away regarding Instance storage should be to understand that the storage should be viewed as temporary. The data hosted on these volumes is only available while the instance is running on that host. If you stop and start an EC2 instance, by design the instance will migrate to new hardware, and the instance storage will no longer be available. Similarly, if the instance or hosts fails for any reason, you will not have access to the data on instance storage.
- While there's no network latencies to Instance storage, it is important to understand that this is a shared disk subsystem. In the diagram above, Instance A will have access to a portion of the local disks, as will Instance B. These are logically separated however, and Instance A has no way to access the data available to Instance B.
- This storage is 100% free and depending on the instance type/size, you'll have anywhere from 160GB to 49TB of Instance storage available. The only caveat to this is t1.micros and the M class of instances, which do not have any instance storage available.
- Instance storage can be used as the root volume or as secondary volumes for additional data. We'll talk more about this when we dive back into AMI's as they

relate to storage.

- Because instance storage is ephemeral, some good use cases for using it might be buffers, caches and scratch data. Therefore, without the proper planning, it's not going to be the best use-case for something like hosting a database. If that instance fails for any reason, or you accidentally stop/start your instance, that database will be gone, and is unrecoverable by AWS employees. On the other hand, some customers do in-fact store databases and other latency sensitive data on Instance storage, and use internal application level backup methods to back this data up to EBS volumes or other storage devices.

Systems Operations on AWS

Amazon EC2 | EC2 Storage Options

Instance Storage

- Exposed as block devices
- Pre-formatted with ext3
- Virtual devices are ephemeral [0-3]
- Must mount before accessing (Windows EC2Config auto-mounts and formats with NTFS)

Type	Name	Instance Store Volumes
Micro	t1.micro	None (use Amazon EBS volumes)
M1 Small	m1.small	1 x 160 GB
M1 Medium	m1.medium	1 x 410 GB
M1 Large	m1.large	2 x 420 GB (840 GB)
M1 Extra Large	m1.xlarge	4 x 420 GB (1680 GB)
M3 Extra Large	m3.xlarge	None (use Amazon EBS volumes)
M3 Double Extra Large	m3.2xlarge	None (use Amazon EBS volumes)
High CPU Extra Large	c1.xlarge	4 x 420 GB (1680 GB)
High-Memory Extra Large	m2.xlarge	1 x 420 GB
High-Memory Double Extra Large	m2.2xlarge	1 x 850 GB
High-Memory Quadruple Extra Large	m2.4xlarge	2 x 840 GB (1680 GB)
High I/O	hi1.4xlarge	2 x 1024 GB SSD (2048 GB)

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- Instance storage is exposed as block devices, which are pre-formatted with ext3. Once you've mounted the volumes however, you can reformat these to any filesystem type that you'd like to use.
- This table shows a sampling of instance types, their API name, and the number and size of Instance store volumes available to each instance type.
- As you can see, the total Instance storage available increases as other systems resources increase. Some instances also have SSD Instance Storage available, which might be appropriate for extremely latency sensitive operations.
- If the instance type you have supports 1 volume, it will be ephemeral0. If the instance type supports 2, it will be ephemeral0 and ephemeral1 and so on.
- While in Linux you must mount instance store volumes manually, the Windows EC2Config service discussed earlier will actually automatically mount all instance storage volumes available. Again, keep in mind that these will be mounted as drive letters within windows, but may not be permanent storage. Always be sure to know what underlying storage you're using with each drive letter in windows.

Systems Operations on AWS
Amazon EC2 | EC2 Storage Options
Amazon Block Store (EBS)

– Network attached, **persistent** block storage
– Independent lifetime of EC2 Instance
– 1 GB - 1 TB
– Provisioned IOPS to increase IOPS per volume
– Root or secondary volume
– Available within Availability Zone (AZ)
– Replicated within AZ to prevent data loss
– Pay for what you allocate
– Easy to resize (expand and shrink) EBS volume
– vol-xxxxxxxx identifier


Amazon EBS

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- Amazon Elastic Block Store, otherwise more widely known as EBS, is network attached, persistent block storage.
- Unlike Instance storage, EBS volumes have an independent lifetime of EC2 instances, which allows you to persist the data in an offline state even after you have terminated an instance. However, for manageability and ease of use, you do have the option to delete EBS volumes at instance termination.
- EBS volumes can be 1GB to 1TB in size, and you can stripe volumes together in order to achieve larger logical volumes should your application need it.
- When creating an EBS volume, you have the ability to choose a standard volume type, which typically runs at ~100 IOPS, or you can choose to provision your IOPS, up to 4000 IOPS. We'll discuss this in more detail when talking about Storage performance in the next section.
- As with Instance storage, EBS volumes can be your root device, or secondary volumes.
- Unlike Instance storage, EBS volumes are replicated within an AZ on the backend, to help prevent against data loss. There is no additional configuration necessary on your behalf, however to provide additional protection in case of failure, you have the option to:

- Take snapshots to S3, which will discuss in more detail on the next slide.
- With EBS volumes, you pay for what you allocate. Therefore, it's wise to not allocate well beyond what your application actually needs, and then expand or even shrink the volume in order to appropriately size for scale and cost efficiencies. Should you choose to expand an EBS volume, keep in mind that you'll also need to use a tool within the OS to resize the file system partition.
- Because EBS volumes can exist without EC2 instances, they also have a unique identifier, beginning with vol- followed by 8 hex digits. Just as you can tag EC2 instances, you can also tag Volumes for discoverability and billing reasons.

More details on shrinking:

<https://forums.aws.amazon.com/thread.jspa?messageID=413139>

Systems Operations on AWS
Amazon EC2 | EC2 Storage Options
EBS and Amazon S3 Integration

- Store point-in-time, **incremental** Snapshots of EBS volumes
- Restore snapshot to **any Availability Zone** within the Region
- Instance store-backed AMIs stored in S3

The diagram consists of two main components. On the left, there is a red rectangular block labeled "Amazon EBS". On the right, there is a red icon representing multiple stacked cubes, labeled "Amazon Simple Storage Service (S3)". A blue arrow points from the EBS block to the S3 icon.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Amazon Simple Storage Service, S3, is the final piece of our earlier diagram regarding storage relationships. S3 is a highly available object store service on its own, with several other use-cases outside of EC2. To understand why this is important to EC2 however, let's cover the high level features of S3 first:

- S3 is extremely durable, providing eleven 9's of durability and four 9's of availability. It achieves this by automatically replicating objects within a specific region to ensure multiple copies of an object exist.
- In the case of EC2, S3 can be leveraged to:
 - Store point-in-time, incremental snapshots of EBS volumes which are then stored in S3. You can leverage the snapshot process in everything from regular, ongoing backups to providing a checkpoint snapshot before making configuration changes. Some customers will even use snapshots to hydrate 'clones' of EBS volumes to get this data to multiple instances, even in other regions by using the snapshot copy process.
 - As we'll discuss in a few moments, S3 is also the location that stores the bits necessary for Instance store-backed AMI's

Systems Operations on AWS

Amazon EC2 | EC2 Storage Options

EC2 Instance Root Device

- AMI; Template or launch configuration
- Amazon **Instance store-backed** or **EBS-backed**
- Each have unique benefits:

Characteristic	Amazon EBS-Backed	Amazon instance store-backed
Boot Time	Usually less than 1 minute	Usually less than 5 minutes
Size Limit	1 TiB	10 GiB
Data Persistence	Data on Amazon EBS volumes persists after instance termination; you can also attach instance store volumes that don't persist after instance termination	Data on instance store volumes persists only during the life of the instance; you can also attach Amazon EBS volumes that persist after instance termination
Upgrading	The instance type, kernel, RAM disk, and user data can be changed while the instance is stopped.	Instance attributes are fixed for the life of an instance
Stopped State	Can be placed in stopped state where instance is not running, but the instance is persisted in Amazon EBS	Cannot be in stopped state; instances are running or terminated

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Now that we've covered Instance Storage, EBS and S3, let's discuss how these relate to Amazon Machine Images, AMI's, which we touched on earlier.

- As we know, an AMI is essentially a template or launch configuration describing the metadata, Operating System, configuration and so on, for an EC2 instance.
- There are 2 types of AMI's, Instance store-backed and EBS-backed. An Instance store-backed AMI describes root volume data that exists in S3, and is pushed to the local instance storage that hosts your instance. An EBS-backed AMI, describes a configuration where the resulting instance's root volume exists on an EBS volume, which is attached to the host. This volume is created from an EBS snapshot associated with the EBS-backed AMI.
- Each has their own unique characteristics, such as the time to boot, the size of the root volume, and how you can upgrade instances. The 2 main points I want you to take away from this table is that:
 - EBS-backed instances, being that the root volume is attached via a network, have the ability to stop and restart on new hardware, in order to move away from failed hardware. This is by design, and likely going to play a role in whether you choose to use EBS backed instance, or Instance store. If you choose to use instance store-backed and the hardware fails, the only option you have is to terminate the instance and

launch a replacement.

- On the other hand, Instance store-backed instances root volumes run on the local instance store, which is populated with data from an AMI that exists in S3. As this isn't a simple attach operation, this can take additional time to boot, has a smaller size limit, but most importantly isn't persistent, and any data stored on your root volume, or any other instance storage device, will be lost if the volumes fail.

Systems Operations on AWS

Amazon EC2 | EBS Performance Best Practices

EBS Options

- Standard EBS Volume
 - 1 GB – 1 TB size
 - ~100 IOPS, with ability to burst
 - Well-suited for boot volumes due to burst
- Provisioned IOPS EBS Volume
 - 10 GB – 1 TB size
 - Designed to be within 10% performance 99.9% of the time
 - Well-suited for I/O intensive applications, such as Databases
 - Hand-in-hand with EBS Optimized Instances
 - Additional cost

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

EBS volumes come with 2 options, either standard or Provisioned IOPS.

- From a performance perspective, standard volumes can be thought of as similar to a single network attached spindle.
 - They can be 1GB – 1TB in size, though you can add multiple volumes into a larger logical volume in order to achieve larger sizes.
 - They generally can achieve around 100 IOPS, however they do have the ability to burst well beyond this for short periods of time.
 - As such, they're well suited for boot volumes, as the bursting can help improve launch times.
- We also offer Provisioned IOPS volumes, also known as PIOPS.
 - PIOPS volumes start at 10GB in size, and go up to 1TB in size.
 - Unlike Standard volumes which can burst at times, PIOPs are designed to operate within 10% of their provisioned amount, 99.9% of the time. In other words, if you provision a 4,000 IOPS volume, 99.9% of the time, it will operate very near the 4k IOPS rather than standard EBS volumes, which cap at ~100IOPS and are somewhat more unpredictable in terms of guaranteed performance.

- Because PIOPS performance volumes are extremely predictable, they are going to be well suited for applications that require consistency in terms of performance. Specifically production databases come to mind here.
- EBS-optimized instances fit well with PIOP volumes, which we'll talk about in more detail in just a moment.
- PIOPS do come with an additional cost, which includes both the cost of the total storage, as well as the amount of IOPS that you provision. Although there is an additional cost for Provisioning IOPS, for steady state I/O over a certain threshold, a properly provisioned PIOPS volume can be more cost effective.

Systems Operations on AWS
Amazon EC2 | EBS Performance Best Practices

Volume Striping

- Increase number of volumes for greater I/O
- Can achieve 45,000+ IOPS with EBS
- Can achieve 100,000+ IOPS with local SSD volumes

vol-xxxx1 100 IOPS vol-xxxx2 100 IOPS vol-xxxx3 100 IOPS vol-xxxx4 100 IOPS

RAID = ~400 IOPS

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

So what if you require more performance than a single volume can provide on its own? Just as you can stripe volumes in a traditional environment, you can do this in AWS as well using software based RAID.

While we'd recommend performing your own testing, and finding what's necessary and cost effective for your applications, we've had customers go over 45k IOPS with EBS volumes, and even over 100k IOPS when using instances with SSD based local instance storage.

While there's nothing preventing you from running any type of RAID configuration within your instance, we find most customers land on RAID 0 and RAID 10, mostly for performance reasons.

Also, remember that if you require greater concentrations of random I/O, increase the total number of volumes. Running 8 x 250 GB volumes will give better performance than 2 x 1 TB volumes.

Supplementary details:

100k+ IOPS referenced on page 14 of

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

Systems Operations on AWS

Amazon EC2 | EBS Performance Best Practices

EBS Optimized Instances

- Dedicated capacity for EBS I/O
- 500 Mbps and 1,000 Mbps depending on Instance Type
- Additional hourly fee
- When to use:

EBS-Optimized instance	Volume type	Recommended use
Yes	Provisioned IOPS	Performance-sensitive production databases. Workloads with minimal variability and dedicated EBS traffic that requires high IOPS performance.
No	Provisioned IOPS	Not recommended, as Provisioned IOPS volumes are designed to deliver the expected performance only when attached to an EBS-Optimized instance.
Yes	Standard EBS	Scaling throughput from a number of volume stripes where I/O performance consistency isn't critical.
No	Standard EBS	Low-cost persistent storage for applications that are not performance sensitive, such as low transaction databases or log files. Also well suited for use as boot volumes.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

A few moments ago we hinted at the idea of EBS Optimized instances. This is a flag that can be enabled for some types of instances, which allow you to dedicate network I/O for EBS traffic. This can be enabled for 500 or 1,000 Mbps, depending on the instance type. There is an additional hourly fee, but are recommended when you have extremely performance sensitive applications, that require minimal variability in performance.

In-fact, in order to attain expected PIOPS performance, they must be attached to an EBS-Optimized instances.

While it might go without saying, something such as storing logs, or some other low transactional application likely won't require the additional costs associated with EBS optimized instances and PIOPS.

Systems Operations on AWS
Amazon EC2 | EBS Performance Best Practices

Other EBS Consideration

- **EBS First access penalty**
 - New Volume = Write all blocks
 - Restored from snapshot = Read all blocks
- **EBS Snapshot Strategy**
 - Degraded performance during snapshot
 - Incremental snapshots; More data change = longer snapshot
 - Ideally, create snapshots off-peak
 - Resume writes after snapshot returns '*pending*'



EBS Snapshot

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

There are a few other considerations to take into account when talking about storage performance in AWS.

- While this is well documented in our EC2 user guides, something very simple is often overlooked when working with storage. Both EBS and Instance storage volumes have what we call a first access penalty. This penalty can be a 5 – 50% reduction in performance on the first read or write. Therefore, we recommend that customers essentially ‘warm’ the volume in order to eliminate this penalty from affecting production workloads, though this is absolutely not required and in many cases may be unnoticeable. If you are creating a new volume from scratch, the recommendation is to write to all blocks. If you’re restoring to an EBS volume from a snapshot, the recommendation is to read all blocks from your volume.
- Another often overlooked fact is the performance penalty one can see while an EBS snapshot is in progress. Because snapshots are incremental, one strategy here is to take more frequent snapshots, reducing the time to completion. If this isn’t possible, ideally you’ll want to create your snapshots during off-peak hours, in order to avoid or at least minimize any performance impact on your volumes. One additional note regarding snapshots, while performance might be slightly impacted while the snapshot is in place, you can resume writing to your volume immediately after the API returns the pending status. You do not need to wait for the snapshot to complete to resume writes.



EBS Lab

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS
Amazon EC2 I Lab Exercise
Lab Objectives

In the following lab you will be building a high speed storage system for a database server.

- Create, configure, and attach EBS volumes to an EC2 instance
- Create and instantiate snapshots
- Use the EC2Config tool

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

In the following lab you will be building a high speed storage system for a database server.

- Create, configure, and attach EBS volumes to an EC2 instance
- Create and instantiate snapshots
- Use the EC2Config tool



Module 6: Tagging

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Tagging I Overview

Introduction

- Question: how do we organize resources on-premises?
 - Naming conventions
 - Subnets / VLANs
 - Physical separation
 - Spreadsheets / asset management systems
- Goal: To get our AWS resources organized with tags
- Activities: Presentation, short exercise, lab

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

If your organization already has a large on-premises infrastructure, then you probably already have a way that you organize your resources. This can be done in a variety of ways – often the naming convention of servers and databases will indicate some basic information about the owner and functionality of the resource, and you end up with hostnames like "AUS-FN-PTL-APP-02". If you've worked at multiple organizations then you've probably seen at least one instance where this scheme got out of control and you need a glossary just to figure out what the thing is.

Other organizations might try to physically or logically separate resources by department or project, with varying levels of success.

At a certain size, you're almost guaranteed to have some sort of IT Asset Management system; for some companies this might be a simple spreadsheet, for others it might be a full-blown Enterprise asset solution.

What we're going to learn in this module is how to replicate that functionality in AWS by using tags. We're going to discuss what tags are, why you should use them, how to implement them, and how to utilize them. We're also going to have a quick exercise, followed by a lab.

Systems Operations on AWS

Tagging | About Tags

Introduction to Tagging

● What are tags?

- Key / value
- Metadata
- Applied to a resource
- (Sometimes) inherited

● Why are tags important?

- Organization
- Billing
- IAM resource-level access

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

What are tags? They have a few important characteristics:

- Tags are key / value pairs – although more common in development, key/value pairs are becoming more and more common in Operations, so you better get used to them.
- Tags are metadata – that means that they don't actually do anything – they're purely for labeling purposes
- Each tag is associated with a single resource, and applies only to its resource
- And finally, tags are sometimes inherited; as we discussed in the AWS introduction, some services like Auto Scaling, Amazon CloudFormation, or AWS Elastic Beanstalk can create other resources like RDS or EC2 instances. Generally, whenever one of these services creates a resources, it will tag that resource with a reference to itself. We'll discuss this more in a little bit, and we'll see this in action in later modules.

The screenshot shows the AWS EC2 Instances page. In the top navigation bar, there is a yellow button labeled "Systems Operations on AWS". Below it, a blue button says "Tagging | Assigning". A blue header bar at the top of the main content area says "Assigning Tags with the Console". On the left, a sidebar lists categories like Instances, AMIs, and Network & Security. The main content area displays a table of instances. One instance, with the ID i-42a3d9b3, has its "Name" tag set to "web server". Below the table, a "Tags" section allows adding new tags. A tooltip explains that tags are key-value pairs used for organization and search. At the bottom, there are copyright notices for Amazon Web Services and a "Feedback" link.

So, how do we actually assign tags to an AWS resource? One way tags can be set is in the console – in this example the Name tag of an instance is being set to "web server".

The screenshot shows the AWS Systems Operations on AWS interface with the 'Tagging | Modifying Tags' section selected. It features a 'Add, Remove, or Edit' button. Three command examples are displayed in a code block:

```
$ aws ec2 describe-tags --filters \
Name=resource-id,Values=i-01234567--output text
instance    i-01234567 development   Environment
instance    i-01234567 Test instance Name

$ aws ec2 create-tags --resources i-01234567--tags \
key=Name,Value="Custom tagged server",\
key=Contact,Value="joe@example.com"
true

$ aws ec2 describe-tags --filters Name=resource-id,\
Values=i-01234567--output text
instance    i-01234567 joe@example.com Contact
instance    i-01234567 development   Environment
instance    i-01234567 Custom tagged server Name
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

As with all things AWS, we can use the CLI to add, remove, or edit the tags of an AWS resource. In the above example, the Name tag for an instance is changed from "web server" to "old web server", and the Contact tag is added.

For the Windows folks, you can run this exact command in PowerShell, substituting "Select-String" for "grep"

Also note: the second step uses the "ec2-create-tags" command – you can also use the ec2-delete-tags command to delete tags.

Commands:

```
$ aws ec2 describe-tags --filters Name=resource-id,Values=i-01234567--output text
instance    i-01234567 development Environment
instance    i-01234567 Test instance Name
```

```
$ aws ec2 create-tags --resources i-01234567--tags key=Name,value="Custom
tagged server",key=Contact,value="joe@example.com"
true
```

```
$ aws ec2 describe-tags --filters name=resource-id,values=i-01234567--output text
instance    i-01234567 joe@example.com Contact
instance    i-01234567 development Environment
instance    i-01234567 Custom tagged server Name
```

The screenshot shows the AWS Systems Operations on AWS console. In the top navigation bar, there's a yellow button labeled "Systems Operations on AWS". Below it, a blue header says "Tagging | Organize Instances". A blue button at the top right says "Change settings". The main area is a table of EC2 instances. On the left, there's a sidebar with links like "Instances", "AWS", "Amazon S3", and "Amazon VPC". A modal window titled "Change settings" is open over the table. It has two sections: "Show/Hide Columns" and "Your Tag Keys". In "Your Tag Keys", there's a list with "aws:autoscaling:groupName" checked, which is applied to "ec2-50-17-33". In "EC2 Instance Attributes", there's a list of attributes like "Instance", "Platform", "Root Device", "State", "Type", etc. At the bottom of the modal, there's a table showing specific tags for an instance, such as "portal", "hr@example.com", "development", "web-server", and "love". The bottom right of the modal has "Cancel" and "Apply" buttons.

- From the console, you can change the settings to utilize any tag as a column for sorting.
- On the right are EC2 Attributes
- On the left are tag keys that have been applied to at least one EC2 instance
- At the bottom of the tag list, you can see a tag called "aws:autoscaling:groupName" – that's an example of an inherited tag that's created and applied when Auto Scaling creates an instance.

The screenshot shows the AWS EC2 Instances page. At the top, there's a navigation bar with tabs for 'Systems Operations on AWS' (highlighted), 'Tagging | Organize Instances', and 'Sorting By Tagging'. On the left, a sidebar lists categories like Instances, Images, and Elastic Block Store. The main area displays a table of EC2 instances with columns for Application, Environment, Name, Project, Instance, and State. The instances are sorted by Environment, with 'production' instances grouped together. A tooltip for one instance says 'EC2 Instance: web server (i-386bef53)'. Below the table, there's a detailed view of the selected instance, showing its AMI, Zone, and Security Groups.

Application	Environment	Name	Project	Instance	State
portal	production	web server	love	i-386bef53	running
portal	production	app server	love	i-69989700	running
EOM ETL	production	db	saver	i-ee536962	running
EOM ETL	production	web server	saver	i-c45369a8	running
portal	staging	web server	love	i-a4498ccb	running
portal	staging	web server	love	i-c24498a3	running
portal	staging	ann server	love	i-ea78f285	running

In this screenshot, all instances are sorted by environment, although this could obviously be any tag value; sorting by a tag in the console is handy for getting a holistic view of AWS resources, especially if you don't really know what you're looking for.

The screenshot shows the AWS EC2 Instances search results page. The search term 'saver' has been entered in the search bar. The results table shows four instances:

Application	Environment	Name	Project	Instance	State
EOM ETL	production	db	saver	i-d79d97b6	running
EOM ETL	production	web server	saver	i-199e947b	running
EOM ETL	development	web server	saver	i-4384ba23	running
EOM ETL	development	db	saver	i-bc986d0	running

Below the table, there is a 'Tags' section with the following data:

Key	Value	Actions
Application	EOM ETL	<input type="button" value="Remove Tag"/> <input type="checkbox" value="Hide Column"/>
Contact	financedev@example.com	<input type="button" value="Remove Tag"/> <input type="checkbox" value="Show Column"/>
Environment	development	<input type="button" value="Remove Tag"/> <input type="checkbox" value="Hide Column"/>
Name	web server	<input type="button" value="Remove Tag"/> <input type="checkbox" value="Hide Column"/>
Project	saver	<input type="button" value="Remove Tag"/> <input type="checkbox" value="Hide Column"/>

You can also search using the console. In this hypothetical example, you are working on the "saver" application, so you are doing a quick search in the console to find out which EC2 instances are associated with that application. Here you can do this by searching for all instances with "saver" in a tag or field. Of course, this will match "saver" in **any** tag or field, so this won't be the best method for every single search, but this can be an easy way to manually find what you're looking for.

The screenshot shows the AWS Systems Operations on AWS interface. At the top, there's a navigation bar with 'Systems Operations on AWS' and 'Tagging | Searching'. Below that is a section titled 'Searching Using the CLI' containing two examples of AWS CLI commands:

```
$ aws ec2 describe-tags --filters \
Name=key,Values=Project,Name=value,Values=saver --output text
instance    i-01234567 saver    Project
instance    i-01234568 saver    Project
instance    i-01234569 saver    Project
instance    i-01234570 saver    Project

$ aws ec2 describe-tags --filters Name=resource-id,Values=i-
9a6269f6
instance    i-9a6269f6 2000    Cost Center
instance    i-9a6269f6 development    Environment
instance    i-9a6269f6 db    Name
instance    i-9a6269f6 Finance Department
instance    i-9a6269f6 saver    Project
```

At the bottom of the screenshot, there's a small copyright notice: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

If you prefer to use command-line, the CLI can be used for these same purposes. This is a simple example using the EC2 CLI. The first example demonstrates a search for any EC2 resources tagged as "project=saver". The second example lists all the tags that have been applied to a single instance.

All of the SDKs also support querying and editing tags for applicable resources.

The text (for future cut/paste)

```
$ aws ec2 describe-tags --filters
name=key,values=Project,name=value,values=saver --output text
instance    i-01234567 saver Project
instance    i-01234568 saver Project
instance    i-01234569 saver Project
instance    i-01234570 saver Project
```

```
$ aws ec2 describe-tags --filters name=resource-id,values=i-9a6269f6
instance    i-9a6269f6 2000 Cost Center
instance    i-9a6269f6 development    Environment
instance    i-9a6269f6 db    Name
instance    i-9a6269f6 Finance Department
instance    i-9a6269f6 saver    Project
```

The screenshot shows a web-based interface for AWS Systems Operations. At the top, there's a navigation bar with 'Systems Operations on AWS' and 'Tagging | Searching'. Below that is a blue header bar with 'Other resource types'. A terminal window displays the following command and its output:

```
$ aws ec2 describe-tags --filters Name=resource-id,Values=vol-a6c2f0ec
volume vol-a6c2f0ec      /dev/sdg      Attachment
volume vol-a6c2f0ec      2 of 4      Raid Disk
volume vol-a6c2f0ec      test RAID volume      Name
```

At the bottom of the terminal window, there's a small copyright notice: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

Just to re-iterate, tags can and should be used for all supported types, not just EC2 instance. Here's an example of using tags to keep track of a disk in a RAID group.

The text (for future cut/paste)

```
$ aws ec2 describe-tags --filters Name=resource-id,Values=vol-a6c2f0ec
volume vol-a6c2f0ec      /dev/sdg      Attachment
volume vol-a6c2f0ec      2 of 4      Raid Disk
volume vol-a6c2f0ec      test RAID volume      Name
```

Systems Operations on AWS

Tagging | Tagging Conventions

Examples

- Name (app 12 web server - 2013-08-22)
- Primary Support Contact (app12devteam@example.com)
- Cost Center (1234)
- Environment (development)
- Stakeholder Team (marketing)
- Project Codename (trendsetters)

What tagging conventions do you use?

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

So, how do we tag resources? Well, like I said at the beginning, if you work for a large enterprise, then odds are you already use some system for organizing resources, and you already have a set of columns or fields that you populate for each server or device. If you are happy with that system, then you can implement it with tags. If you aren't currently managing resource metadata, then here are some sample ideas:

For EC2 / RDS instances:

Name – note that generally you want to have a name that is descriptive and easily understood.

Primary / Secondary / Tertiary Support Contact
Cost Center
Environment
Stakeholders
Project Name

Other ideas:

- the name of the application that it's running
- the email for the primary business contact
- Maybe somebody asked for you a server "just for a week or two" – you can put in an "expiration date" to remind your organization to turn it off after that week or two has passed
- You could have a tag for the application healthcheck URL or the project wiki URL

For EBS volumes, you might want to assign tags for things like the application, the device assignment (/dev/sdf), or the drive letter (D:) – this can be extremely helpful if you're trying to rebuild or migrate servers / EBS volumes – it's far easier to do when the required information is already associated with the EBS volume.

The screenshot shows a section titled 'Tagging Parameters' with the following bullet points:

- 10 tags / resource
- Max key length -> 127 Unicode characters
- Max value length -> 255 Unicode characters
- Prefix "aws:" is reserved
- Limited set of currently taggable resources (EC2, EBS, S3 Buckets, RDS, Auto Scaling)

At the bottom of the page, there is a small copyright notice: © 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

We do have some limitations on tagging that you'll probably want to be aware of.

We are currently limited to 10 tags per resource.

Each tag has a maximum key length of 127 Unicode characters, and a maximum value length of 255 Unicode characters.

When a taggable resource is created by another AWS service (in the case of Auto Scaling, Elastic Beanstalk, CloudFormation, etc), some tags will be assigned to that resource with information about the service that created it – for example, the CloudFormation stack name. Any tags that were applied to that service resource (for example, an "Environment" tag on the CloudFormation stack) will also be propagated in this way. All these tags will have the prefix "aws:" – you can't edit or delete those tags, and you can't create your own tags that start with "aws:". Note that these tags don't count against the 10-tag limit.

Not all resources are currently taggable; you should generally be able to find a "Tags" tab on the console for everything that's taggable. All EC2 resources except for EIPs and ELBs are taggable, along with RDS and S3 buckets, so the majority of services that you'll be using (at least in the beginning) will be taggable.

Systems Operations on AWS

Tagging | Creating Tags with AWS CloudFormation

Tagging Example – EC2

```
"SimpleInstance" : {
    "Type" : "AWS::EC2::Instance",
    "Properties" : {
        "InstanceType" : "m1.small",
        "KeyName" : "myKey",
        "ImageId" : "ami-01234567",
        "SubnetId" : "subnet-01234567",
        "SecurityGroupIds" : [ "sg-01234567" ],
        "Tags" : [
            { "Key" : "Name", "Value" : "web server" },
            { "Key" : "Environment", "Value" : "development" },
            { "Key" : "Application", "Value" : "HR portal" }
        ]
    }
}
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

As with pretty much everything else in AWS, tags can also be created through the use of CloudFormation templates; this is a very simple CloudFormation snippet that creates an EC2 instance with three tags.

Systems Operations on AWS

Tagging | More Uses

amazon
webservices

- Detailed billing
 - Include tags in billing reports
 - Allocate costs
- IAM policies
 - Example: Environment=development

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Before moving on to the lab, we'll touch on two more uses for tags. First off, tags, can be included in detailed billing reports – this means that tags can be included in itemized bills, which can be used for cost allocations and chargebacks.

Tags can also be used in setting IAM policies – users can be granted access only to resources where a particular tag matches. A common use for this would be to create a group with full access to instances with an Environment tag of "development", which would prevent that group from accessing production resources.

We will get into both of these uses in detail in later modules.

Systems Operations on AWS

Tagging | Summary



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- Tags are key/value metadata pairs that are associated with AWS Resources
- Tags can be modified and displayed using the Management Console, CLI, SDKs, or API
- Not all AWS resources currently support tags.
- Tags can be used to organize, sort, and search in AWS resources
- Key components of Cost Optimization / Allocation and Security

Recap about tags:

- Tags are key/value metadata pairs, associated with AWS Resources
- Tags can be modified by the console, CLI, SDKs, or API
- The core AWS currently support and list of taggable resources is growing
- What do we use tags for? We use them to organize, sort, and search in AWS resources

We'll be seeing a LOT more of tags in the future, especially in the cost and security modules – from now on we'll make it a practice to tag all of our resources, and later on in the course we'll show how we can actually use tags as separation criteria in AWS billing reports.



LAB

Tagging

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Tagging | Lab Exercise

Lab Objective

Use both Management Console and CLI

- Manipulate tags
- Organize and search using tags

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



The lab objectives are as follows:

Use both Management Console and CLI

- Manipulate tags
- Organize and search using tags

We are going to use the console and the CLI to do some basic tag management, and then use both the console and the CLI to search for specific tags.



Module 7: Monitoring

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Enterprise Monitoring I Overview

What We Will Cover

- Monitoring and Amazon CloudWatch
- Integrating CloudWatch with monitoring software
- Monitoring elastic resources
 - General guidelines
 - Patterns
 - Helpful tools
- Lab activity
 - Custom Metrics
 - Client Registration
 - Alarms and Notifications



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

In this module, you are going to be diving into monitoring, and discussing ways to integrate AWS with traditional monitoring solutions.

You will start with some notes and a brief discussion about monitoring and the overlaps and differences between CloudWatch and traditional monitoring solutions. You will look at some patterns for integrating CloudWatch with monitoring software.

You will go over general guidelines for monitoring AWS resources, patterns for implementing monitoring, and consider what tools you are going to need in order to effectively integrate AWS with a monitoring solution.

Finally, you will have a lab activity in which you will set up an Auto Scaling group and go through the monitoring scenarios.

Systems Operations on AWS

Enterprise Monitoring I Overview

Learning Objectives

By the end of this section you should be able to:

- Describe the differences between monitoring in AWS and traditional monitoring.
- Decide what to monitor with Amazon CloudWatch, and what to monitor with monitoring software.
- Describe optimal patterns for integrating AWS resources with traditional monitoring software.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



By the end of this section you should be able to:

- Describe the differences between monitoring in AWS and traditional monitoring
- Decide what to monitor with CloudWatch, and what to monitor with monitoring software
- Describe optimal patterns for integrating AWS resources with traditional monitoring software

Systems Operations on AWS
Enterprise Monitoring I Overview
Why Monitor?

- Information Technology Infrastructure Library (ITIL): "The act of observing, supervising, reporting or controlling the activities of another entity."
- Why do you monitor services / resources?
 - SLAs
 - Automation

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Before diving into the details, let's just level-set on how we define "monitoring". Information Technology Infrastructure Library (ITIL) defines monitoring as "The act of observing, supervising, reporting or controlling the activities of another entity." That's a decent enough definition of the "what", so let's quickly talk about the "why". Why **do** we monitor services or resources?

Obviously, we're generally checking availability:
--"Is the site up? How about that backend API service?"

We're generally also looking to get some performance metrics for that component or service as well.

"How fast is the API server responding? Is the backend DB under heavy load?"

There are lots of reasons why monitoring is a good thing: we want our users to have a good, predictable experience; we want to proactively spot performance trends; we want to establish baselines of what "normal" load / response times are, etc. In the Ops world, these all come down to SLAs; we want to make sure that we're meeting our availability and performance SLAs, and to be proactively warned if we're in danger of not meeting them.

Monitors are also the first step towards automation; if we can programmatically spot performance inefficiencies then we can programmatically adjust for them. A basic example of this is auto-scaling.

Systems Operations on AWS
Enterprise Monitoring | Metrics
Amazon CloudWatch

- Scalable, highly available metrics collection system
- Metrics relevant to resources “out of the box”
 - EC2: CPU Utilization
 - EBS: VolumeReadOps
- Ability to insert custom metrics
 - Memory, Swap, Disks
 - Application metrics
- Thresholds and alarms:
 - SNS message
 - Auto Scaling policy



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

We’re going to get much deeper into CloudWatch later on in the course. For now, here are the things you need to know about it:

CloudWatch is a scalable and highly available metrics collection system that can be utilized to monitor AWS resources. It’s able to quickly deliver both raw metric data and statistics about that metric data.

By default, many AWS resource types (like EC2 instances, EBS volumes, ELBs) are monitored with CloudWatch and produce some predefined metrics relevant to that resource. For example, one metric available for EC2 instances is the CPU utilization. One metric available for an EBS volume is the number of read operations.

You can also add custom metrics to CloudWatch – for example, if you wanted to check the number of concurrent connections to your web server, you could make that a custom metric.

CloudWatch also has the ability to set thresholds and to activate alarms when those thresholds are reached. Currently an alarm can do one (or both) of two things: publish a message to an SNS topic, or trigger an Auto Scaling policy.

A simple example might be to have an SNS topic for “application X Auto Scaling farm” that sends email to your NOC, and two Auto Scaling policies – one to add instances to an Auto Scaling group, and one to remove instances. You might set a CloudWatch alarm that says “when my instances are, as a group, at more than 70%

CPU utilization for 10 minutes, add instances to the Auto Scaling group. When they are at less than 20% utilization for 15 minutes, remove instance from the group. In both cases, also send an message to the SNS topic.”

We will get into much more detail about this later on in the course, but that's the overview.

For some organizations, CloudWatch monitoring might be enough without having to introduce anything else into the mix.

Systems Operations on AWS

Enterprise Monitoring | Defining custom Metrics

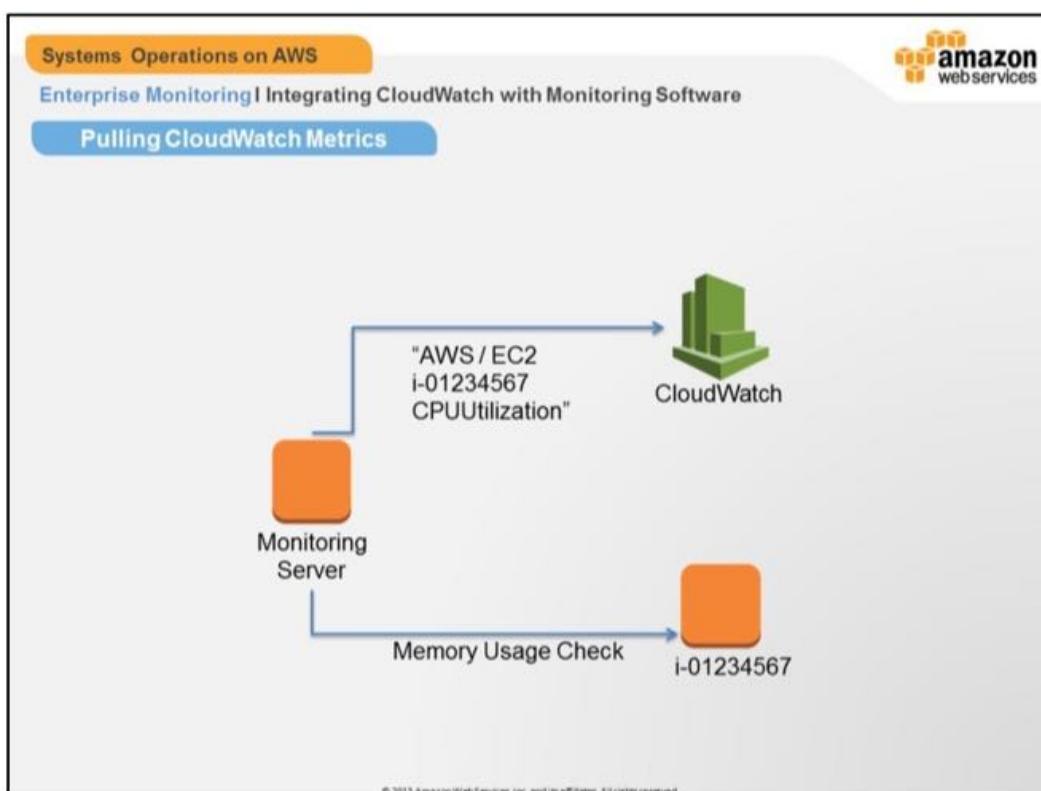
Custom Metrics

- Infrastructure Metrics
 - Memory, Disks, Swap and other OS metrics are available as an installable package ([Monitoring Scripts for Amazon EC2 Instances](#))
- Platform Metrics
 - JMX statistics (Tomcat Threads, JVM Heap and GC, ...)
 - Database connection pools and queries
 - Mail Session Activity
 - ESBs, MQs, Directory Services, etc...
- Application Metrics
 - Sales volume
 - Tickets opened
 - Events processed

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Custom metrics can have any meaning and range of values, but typically are related to infrastructure and scaling information, such as:

- Infrastructure metrics, measuring low level resources, such as memory and disks.
- Platform metrics, measuring middleware resources, such as threads and sessions.
- Application metrics, measuring business operation, such as sales or other application events.



Even if you use a 3rd-party monitoring solution, CloudWatch metrics can still be useful as a source of truth for metric data about a resource.

CloudWatch data can be retrieved via the CLI, SDK, or API and then imported into monitoring software.

Take the case of an EC2 instance – CloudWatch already provides metrics for CPU Utilization, so that metric can be taken straight out of CloudWatch. For any information that's not accessible by the hypervisor (such as memory utilization), you will still need to query the instance directly.

So why do this instead of just getting the CPU utilization from the box directly? Two good reasons:

- 1) Hypervisor is guaranteed to be most accurate metric – in the case of CPU, some OS's will mistakenly factor CPU throttling as “steal”, resulting in misleading metrics.
- 2) One CloudWatch call can get metrics for multiple AWS resources, which saves bandwidth and system resources for the server and for targets.

Systems Operations on AWS

Enterprise Monitoring | Sample CloudWatch Queries

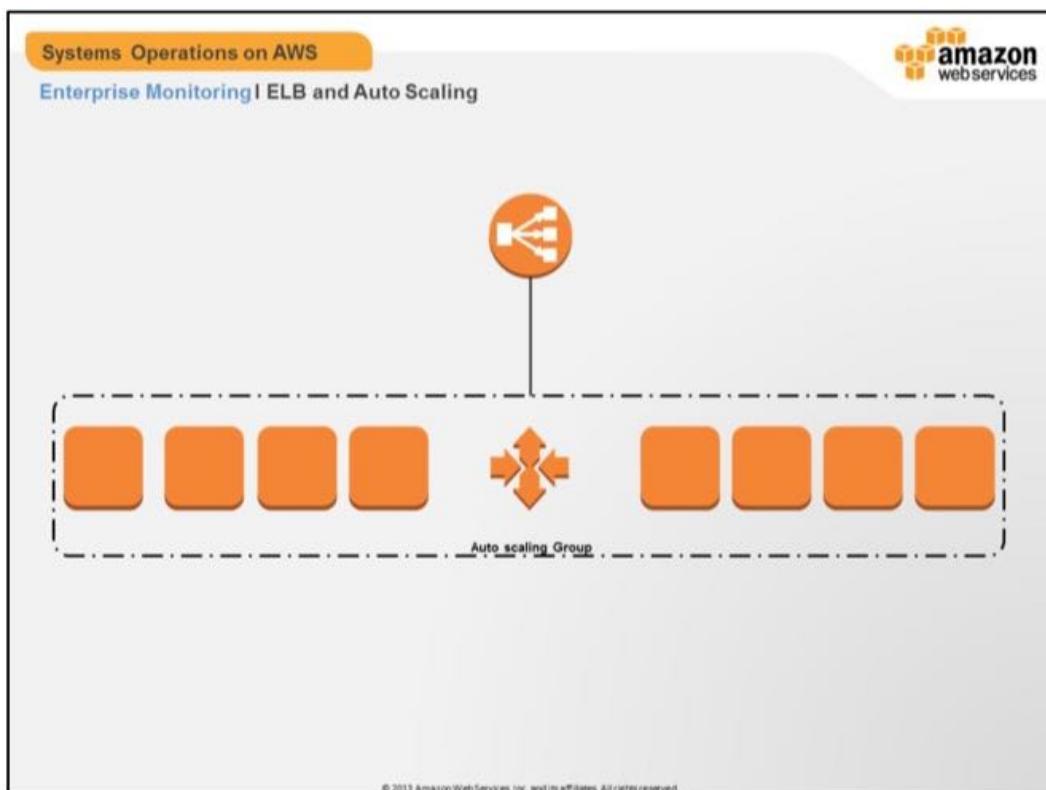
Sample CloudWatch Queries

```
$ aws cloudwatch get-metric-statistics \
--namespace "AWS/EC2" --metric-name CPUUtilization \
--statistics Average \
--start-time $(date -d '10 minutes ago' -u +%Y-%m-%dT%H:%M:%S.000Z) \
--end-time $(date -u +%Y-%m-%dT%H:%M:%S.000Z) \
--period 600 \
--dimensions Name=InstanceId,Value=i-01234567
```

```
$ aws cloudwatch get-metric-statistics \
--namespace "AWS/EBS" --metric-name VolumeQueueLength \
--statistics Maximum \
--start-time $(date -d '10 minutes ago' -u +%Y-%m-%dT%H:%M:%S.000Z) \
--end-time $(date -u +%Y-%m-%dT%H:%M:%S.000Z) \
--period 600 \
--dimensions Name=VolumeId,Value=vol-01234567
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Here are a couple of example CloudWatch queries using the CLI - the first is a query for the average CPU utilization of an instance and the second is a query for the maximum queue length of a volume. Note that both queries take place over the last 10 minutes, but that could be easily altered.

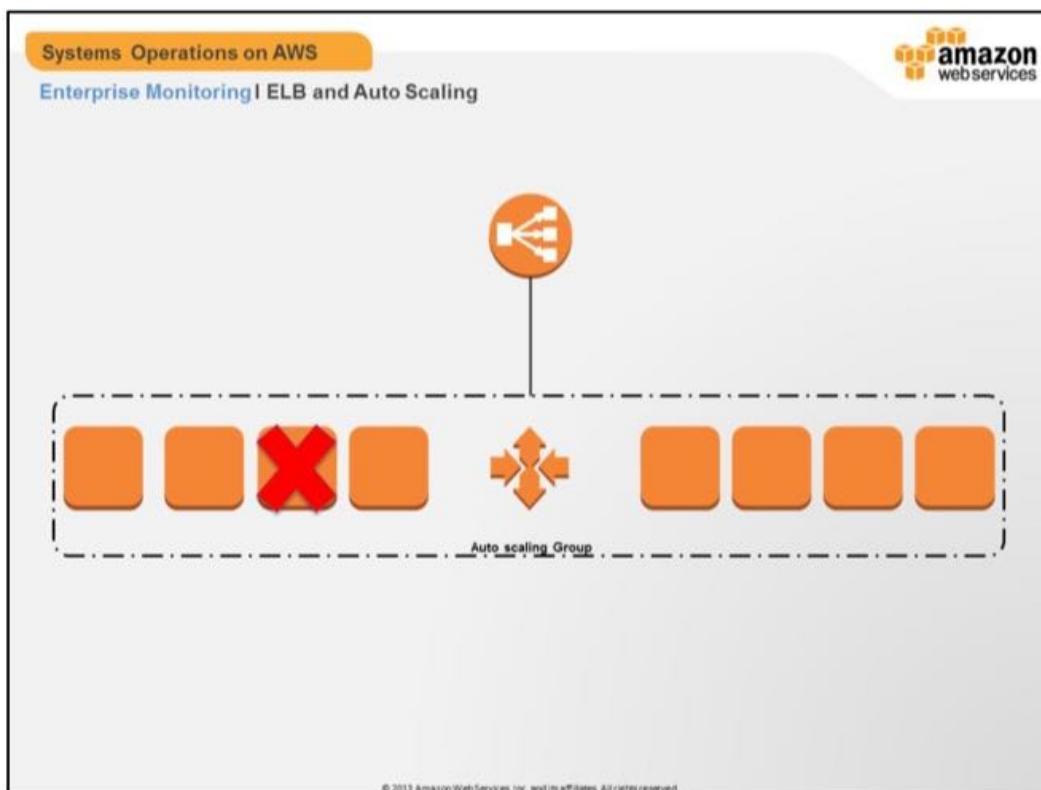


Here's a simplified example: ELBs have an inherent monitor (the health check URL), and Auto Scaling groups can base instance health on the ELB status. Suppose that we have an Auto Scaling group of 8 EC2 instances behind an ELB. In this hypothetical, there's a memory leak in our application, and at 3am the application service dies on one of the instances.

- Since the instance fails the health check, it gets pulled out of the ELB.
- Since the instance gets pulled out of the ELB, it gets removed from the Auto Scaling group and terminated.
- Since the Auto Scaling group specifies 8 EC2 instances, a new server will get put in its place, and we're back to where we started.

In a fixed infrastructure, this sort of outage would probably result in a incident ticket, and most likely be fixed by either a 24x7 Network Operations Center (NOC) or by waking up an admin. We would probably label this as "Sev 2".

In an elastic environment, we most likely want to know about this incident, but we probably don't need to take any action on it. Since everything's been automatically fixed, we might label this as "Sev 3"

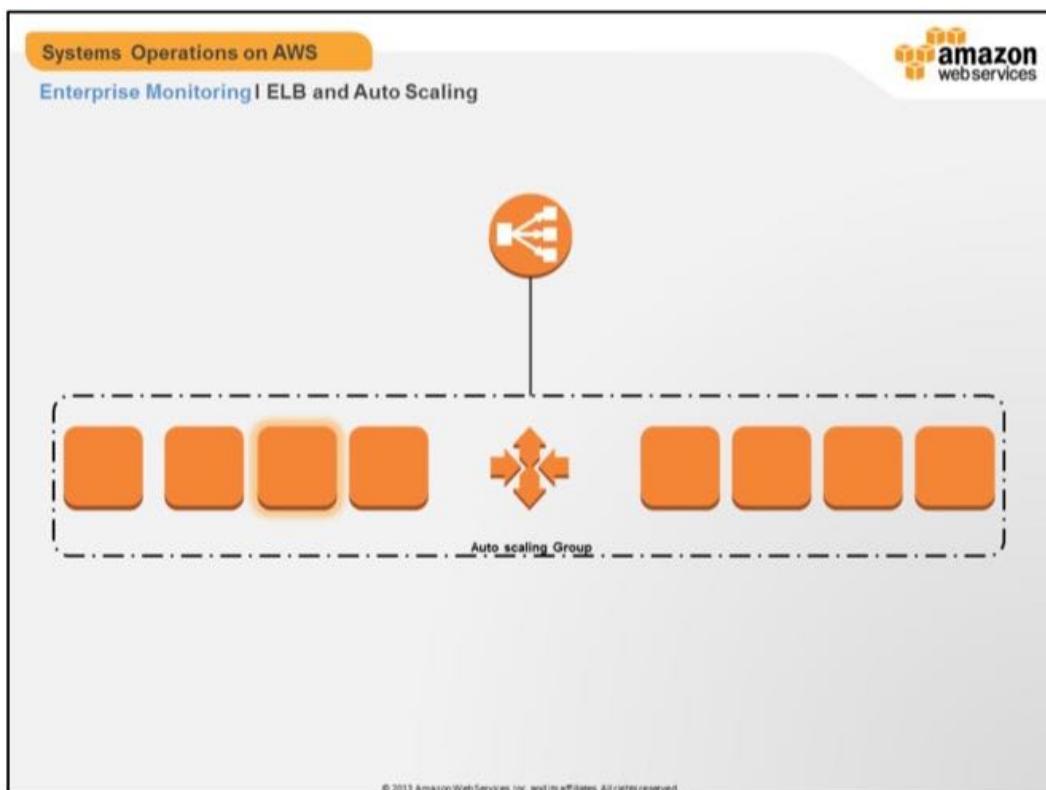


Here's a simplified example: ELBs have an inherent monitor (the health check URL), and Auto Scaling groups can base instance health on the ELB status. Suppose that we have an Auto Scaling group of 8 EC2 instances behind an ELB. In this hypothetical, there's a memory leak in our application, and at 3am the application service dies on one of the instances.

- Since the instance fails the health check, it gets pulled out of the ELB.
- Since the instance gets pulled out of the ELB, it gets removed from the Auto Scaling group and terminated.
- Since the Auto Scaling group specifies 8 EC2 instances, a new server will get put in its place, and we're back to where we started.

In a fixed infrastructure, this sort of outage would probably result in a incident ticket, and most likely be fixed by either a 24x7 NOC or by waking up an admin. We would probably label this as "Sev 2".

In an elastic environment, we most likely want to know about this incident, but we probably don't need to take any action on it. Since everything's been automatically fixed, we might label this as "Sev 3"



Here's a simplified example: ELBs have an inherent monitor (the health check URL), and Auto Scaling groups can base instance health on the ELB status. Suppose that we have an Auto Scaling group of 8 EC2 instances behind an ELB. In this hypothetical, there's a memory leak in our application, and at 3am the application service dies on one of the instances.

- Since the instance fails the health check, it gets pulled out of the ELB.
- Since the instance gets pulled out of the ELB, it gets removed from the Auto Scaling group and terminated.
- Since the Auto Scaling group specifies 8 EC2 instances, a new server will get put in its place, and we're back to where we started.

In a fixed infrastructure, this sort of outage would probably result in a incident ticket, and most likely be fixed by either a 24x7 NOC or by waking up an admin. We would probably label this as "Sev 2".

In an elastic environment, we most likely want to know about this incident, but we probably don't need to take any action on it. Since everything's been automatically fixed, we might label this as "Sev 3"

Systems Operations on AWS

Enterprise Monitoring | Monitoring Auto Scaling

Auto Scaling and Monitoring Services

Integrating with other monitoring services can provide additional features, but may require design considerations, such as:

- How to add and remove instances from the monitoring service?
- What is the difference between a removed and a failed instance?
- Can the monitoring service ingest and react to data as you scale?
- How long monitoring data must be retained?
- Is your monitoring data be confidential?

The answer to those question may be very different if you are integrating with a monitoring software on you infrastructure our with monitoring services available on the web.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

When developing monitoring solutions in auto scaling scenarios, specially with a large amount of clients, the same non-functional requirements of your service may apply to the monitoring service itself. The monitoring solution may also be subject to automation, performance, scalability and security requirements.

Systems Operations on AWS

Enterprise Monitoring | Integrating CloudWatch with Monitoring Software

Monitoring software or services?

Monitoring software and services can vary a lot in their requirements and features, such as:

- Cost
- Latency
- Throughput
- Features

- Convenience
- Integration
- Security
- Capacity

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

What should you consider when choosing between monitoring software or services? It may help to mention examples of the commercial products in these categories:

Monitoring software: Nagios, Cacti, Ganglia, Zabbix...

Monitoring services: Stackdriver, DataDog, NewRelic, CopperEgg, ...

Systems Operations on AWS

Enterprise Monitoring | Third-Party Monitoring Software

Using Your Own Monitoring System

- Proceed as normal
 - Use CloudWatch metrics where available
 - Create relevant monitors (e.g. don't ping RDS)
- What about Auto Scaling?
 - Bad: not monitoring new instances
 - Worse: monitoring terminated instances
- Three ways to manage Auto Scaling instances:
 - Client self-registration
 - Full scan / config re-write
 - SQS

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

For organizations that need to utilize their own monitoring solutions, there's good news: for the most part, you don't need to do anything different. When using VPCs, you can even use the same monitoring server and just run your checks over the VPN connection. Or, if it makes sense, you can also set up a monitoring server inside the VPC.

A couple of guidelines about monitoring AWS resources: first, use CloudWatch metrics whenever they're available. Second, make sure that your monitors match the service. For example, don't bother trying to ping RDS instances - it makes more sense to have a health check against the DB itself, such as a TNSping or a simple SELECT statement against a table.

So, lots of you are probably guessing the tricky part: Auto Scaling. If, for whatever reason, you need to monitor instances that are part of an Auto Scaling group, then you need to find a way to start monitoring new instances that come up, and even more importantly you need a way to stop monitoring instances after they have been removed. In the case of the first part, you run the risk of unmonitored production instances. In the case of the second part, you're going to start getting alerts for instances that have been intentionally removed.

We're going to cover, in some detail, three ways to manage monitoring Auto Scaling instances. We can have client register themselves, we can do a full scan and re-write of the monitoring configuration, or we can publish Auto Scaling events to an

SQS queue and have the monitoring server parse the queue.

First, let's talk about client self-registration.

Systems Operations on AWS

Enterprise Monitoring | Instance (De)registration

Client Self-Registration

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Pro	Con
Easy, but not always available	No de-registration

- Pass server information via CFN Init or User Data
- Use this for: registration

Some monitoring software supports client self-registration – in this model, the client agent registers itself with the monitoring server.

If your client supports this, then the only modification required for Auto Scaling is to somehow make the server information available to the client, which can be easily done by passing via cfn-init or user data.

What's good about this is that it's easy to do – you're just using the software as-is.

What's bad about this is that there's generally no built-in de-registration, so it only works one way.

Nonetheless, if your monitoring software has this feature, then go ahead and use it.

Systems Operations on AWS

Enterprise Monitoring | Instance (De)registration

Periodic Poll and Reconfigure

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Pro	Con
Guaranteed not to miss an event	Wasteful, difficult to scale

– Use this for: initial discovery, scheduled “true up”



A second pattern is to periodically poll your Auto Scaling configuration, compare that configuration to your monitoring configuration, and then true up any differences.

The good thing about this is that it's comprehensive, and you're not going to miss any events. If your monitoring server happened to be in the middle of a maintenance window when an instance comes online, then that new instance will get added at the next refresh cycle.

The bad thing about this is that it's wasteful and it's not going to scale terribly well – once you start getting into the hundreds or thousands of servers, the process of retrieving your entire Auto Scaling footprint and comparing all the pieces starts to get relatively expensive. That doesn't mean that you don't want to ever do this, but you don't necessarily want to do it all the time.

The best use case for this pattern is to build an initial configuration, such as when building a new monitoring server or adding an Auto Scaling group to your monitoring configuration. You could also run something like this at longer intervals just to be 100% sure that your monitoring configuration matches your blueprint.

The screenshot shows a web-based interface for AWS Systems Operations on AWS. At the top, there's a navigation bar with tabs: 'Systems Operations on AWS' (highlighted in orange), 'Enterprise Monitoring', 'Instance (De)registration', and 'Script' (highlighted in blue). On the right side of the header is the Amazon Web Services logo. The main content area contains a list of items:

- Tools:
 - EC2: describe-instances
 - RDS: describe-db-instances
 - Auto Scaling: describe-auto-scaling-groups
- Consider an SDK

At the bottom of the content area, there's a small note: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

A monolithic list of everything may not be a good fit for small deltas in infrastructure, but it can be very useful for an initial build, as well as occasionally comparing the current configuration against the current footprint.

In order to build a script like this, you're going to need various tools to describe the infrastructure. Specifically, you'll need the various "describe" calls for EC2, Auto Scaling, and RDS.

Given that this will be of moderate difficulty, you might also want to consider picking up one of the languages for which AWS has an SDK, although this could be done in bash or PowerShell.

Systems Operations on AWS

Enterprise Monitoring | Instance (De)registration

Publish Changes to SQS for Add/Remove Events



Pro	Con
Scalable, efficient, easy API integration	Doesn't work for discovery, potential missed add/remove events

– Use this for: processing add / remove events
– Still need periodic “true up”

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Finally, we have the pattern of publishing Auto Scaling events to an SQS queue and processing those events on the monitoring server.

What's good about this pattern is that it's frugal with resources and it's scalable. This pattern is also the easiest to integrate with monitoring services that have APIs to add or remove resources.

There are a couple of issues with this pattern: first of all, it only works for deltas – it can't be used for an initial discovery. Also, in the event that there's no SNS / SQS event (like if an instance is manually terminated from the console or API), you might miss an event.

So, while this is definitely the best pattern for processing additions or removals, you probably still want to periodically run a “true up” script.

Systems Operations on AWS

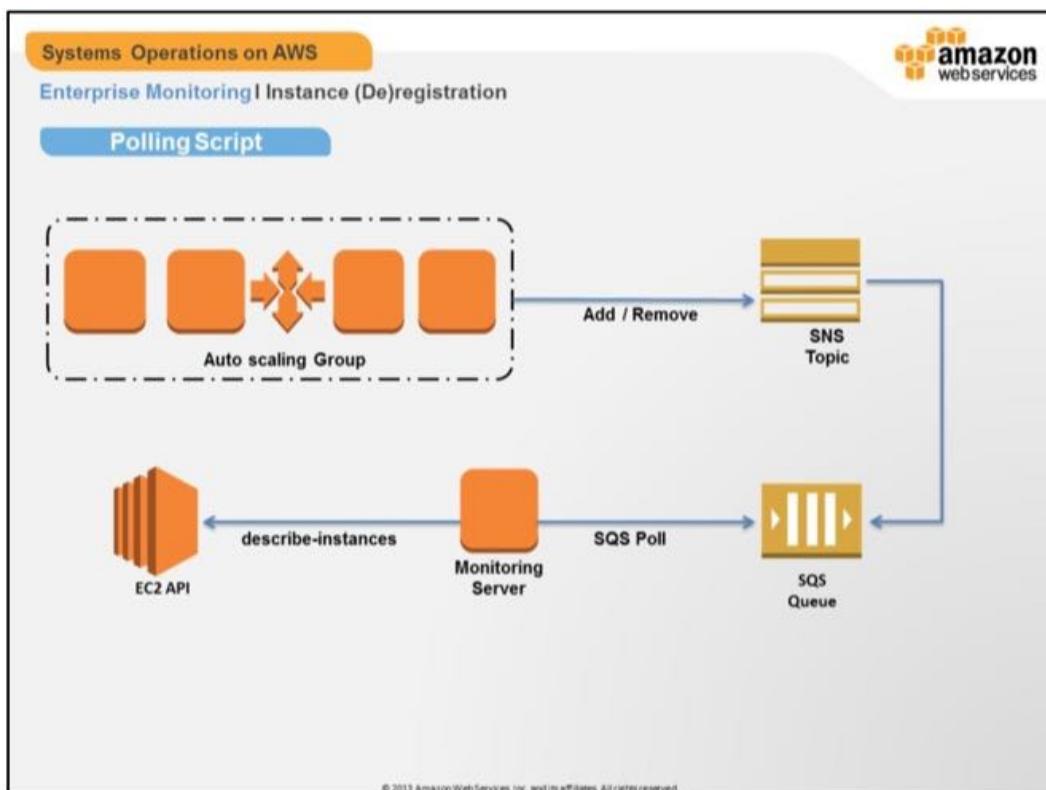
Enterprise Monitoring | Instance (De)registration

Polling Script

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- Utilize to process add / remove events
- Tools:
 - SQS: receive-message, delete-message
 - EC2: describe-instances

In order to process events, you will need to poll an SQS queue and delete messages after they're received. Then, describe the instances.



Here a visual representation of the polling script.

When an Auto Scaling group adds or removes an instance, it sends a message to an SNS Topic. The SNS Topic has an SQS Queue as one of its endpoints (it could also have, for example, an email address to alert admins or for record-keeping purposes). The monitoring server is continuously polling the queue, and when it receives a message from the queue it immediately takes action. If the message is a delete, then it deletes the instance record. If the message is an add, then it can query the EC2 API to get any required information about the instance in order to start monitoring it.

Systems Operations on AWS

Module Title I Summary

CloudWatch provides monitoring of key metrics of AWS resources.

Most AWS resources can be monitored by 3rd-party monitoring solutions with little or no customization.

Auto Scaling instances can be monitored as normal, but will need some helper utilities to be added to or removed from monitoring software.



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

In summary:

- CloudWatch provides monitoring of key metrics of AWS resources.
- Most AWS resources can be monitored by 3rd-party monitoring solutions with little or no customization
- Auto Scaling instances can be monitored as normal, but will need some helper utilities to be added to or removed from monitoring software



LAB

Monitoring Auto Scaling Hosts

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Enterprise Monitoring | Lab Exercise

Lab Objective

- Interact with CloudWatch
- Set up alarms and notifications
- Register instances from Auto Scaling groups



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

For the lab section, we'll be exploring a sample monitored environment in which all Auto Scaling hosts are monitored.

We'll manually activate a scaling policy to create more instances, and confirm that those instances are being monitored.



Module 8: Backup and Archive

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Backup and Archive I Overview

Learning Objectives

By the end of this section you should be able to:

- Describe how to perform backup and archive operations on AWS
- Identify how to choose the best option for backing up and archiving data on AWS
- Differentiate between implementing backups on AWS and traditional backup systems
- Describe how data is transferred from on-premises to AWS
- Describe how to back up data hosted on AWS

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Learning Objectives

- Trainees will be able to Describe the various services
- Trainees will be able to choose the best option for backing up or archiving a system. Different scenarios require different methodologies. Backup strategies would vary when using different services like a Database running on EC2 vs an RDS instance, or Redshift.
- Trainees will be able to differentiate backup and archiving on AWS from traditional backup infrastructure.
- Identify different ways of deploying backup solutions, tools, and methodology.

Systems Operations on AWS

Backup and Archive | Key Terms

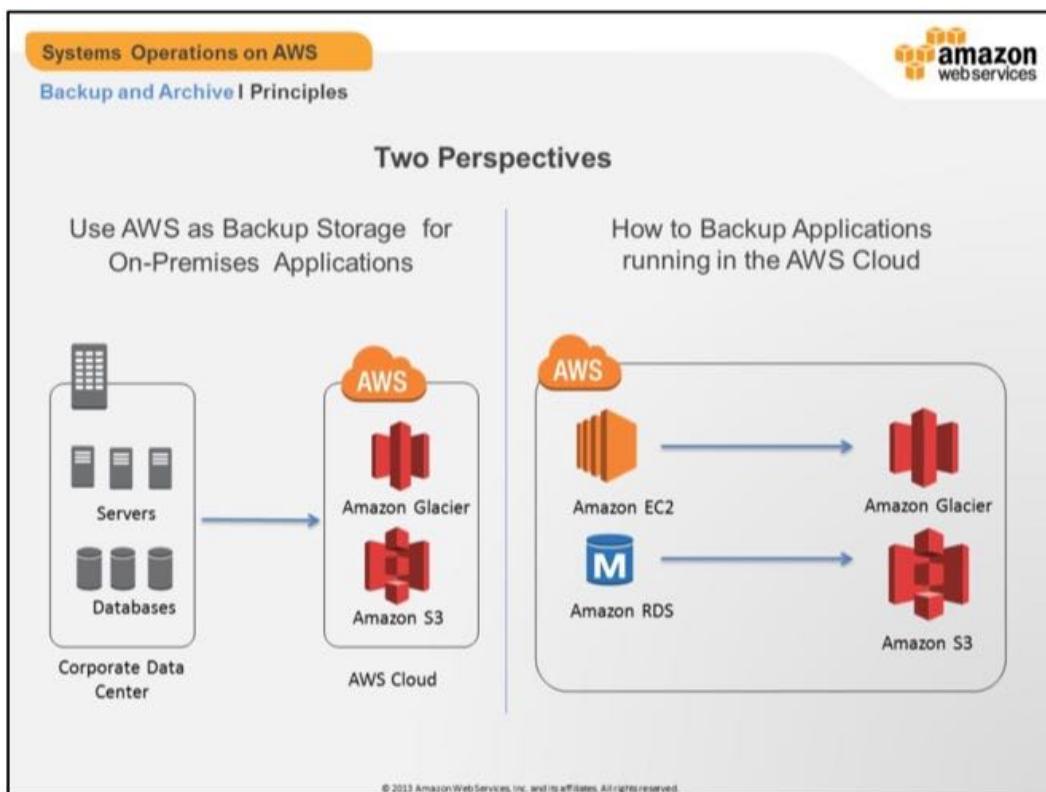
amazon
webservices

Backup	The process of backing up is making copies of stateful data or configuration which may be used to restore the original after a data loss event.
Archive	The process of moving data that is no longer actively used to a separate data storage device for long-term retention.
Restore	The process of restoring stateful data or configuration from a previously known point in time.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Backup - The process of backing up is making **copies of data** which may be used to **restore** the original after a data loss event. The primary purpose is to recover data after its loss, be it by data deletion or corruption. The secondary purpose of backups is to recover data from an earlier time. Several Services on AWS to support this including AWS S3, Storage Gateway and Third Party Gateways

Archive - The process of moving data that is no longer actively used to a separate data storage device for **long-term retention**.



Two Perspectives

There are two perspectives when discussing about backups on AWS. They are -

1. How to use AWS as a Backup Storage Infrastructure for existing applications running on-premises
2. How to Backups Applications hosted in AWS

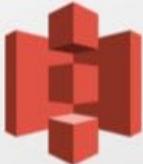
Both of these perspectives will be discussed during the course of the training

Systems Operations on AWS

Amazon EC2 | EC2 Storage Options

Amazon Simple Storage (S3)

Highly available object storage
99.99999999% durability
99.99% availability
Automatically replicated within region
Store point-in-time, incremental snapshots of EBS volumes
Restore snapshot to any Availability Zone within Region
Instance store-backed AMIs stored in S3

 Amazon S3

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Amazon Simple Storage Service, S3, is a highly available object store service on its own, with several other use-cases outside of EC2. Here are the the high level features of S3:

- S3 is extremely durable, providing eleven 9's of durability and four 9's of availability. It achieves this by automatically replicating objects within a specific region to ensure multiple copies of an object exist.
- In the case of EC2, S3 can be leveraged to:
 - Store point-in-time, incremental snapshots of EBS volumes which are then stored in S3. You can leverage the snapshot process in everything from regular, ongoing backups to providing a checkpoint snapshot before making configuration changes. Some customers will even use snapshots to hydrate 'clones' of EBS volumes to get this data to multiple instances, even in other regions by using the snapshot copy process.
 - S3 is also the location that stores the bits necessary for Instance store-backed AMI's

Systems Operations on AWS

Amazon EC2 | EC2 Storage Options

Amazon S3 Buckets and Objects

- Stores data as objects within buckets.
- An object is comprised of a file and optionally any meta data that describes that file.
- You can have up to 100 buckets in your account.
- You can manage access to the bucket.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- Amazon S3 stores data as objects within buckets. An object is comprised of a file, and optionally any metadata that describes that file. To store an object in Amazon S3, you upload the file you want to store to a bucket.
- When you upload a file, you can set permission on the object as well as any metadata.
- Buckets are a logical containers for objects. You can have one or more buckets in your account. For each bucket, you can control access, in other word, who can create, delete and list objects in the bucket.
- You can also view access logs for the bucket, and its objects, and choose the geographical region where Amazon S3 will store the bucket and its contents.

Systems Operations on AWS
Amazon EC2 | EC2 Storage Options
Amazon Glacier

● Lifecycle directly from Amazon S3
● 99.999999999% durability for an archive
● \$0.01 per GB per month
● Archives organized into vaults
● AWS Management Console to create and delete vaults.
Programming for other interactions.

Amazon Glacier

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Amazon Glacier allows you to offload the administrative burdens of operating and scaling archival storage to AWS, and makes retaining data for long periods, whether measured in years or decades, especially simple.

- Designed to provide average annual durability of 99.99999999% for an archive. The service redundantly stores data in multiple facilities and on multiple devices within each facility.
- In order to keep costs low, Amazon Glacier is optimized for data that is infrequently accessed and for which retrieval times of several hours are suitable. With Amazon Glacier, customers can reliably store large or small amounts of data for as little as \$0.01 per gigabyte per month, a significant savings compared to on-premises solutions, which makes this suitable for information archival.
- You store data in Amazon Glacier as archives. An archive can represent a single file or you may choose to combine several files to be uploaded as a single archive. Retrieving archives from Amazon Glacier requires the initiation of a job.
- You organize your archives in vaults. You can control access to your vaults using the [Amazon Identity and Access Management \(IAM\)](#) service.
- You can use the console to create and delete vaults. However, all other interactions with Amazon Glacier require programming.

- For example, to upload data, such as photos, videos, and other documents, you will need to write code and make requests using either the REST API directly or use AWS SDK wrappers.

Systems Operations on AWS

Amazon EC2 | EC2 Storage Options

Amazon Glacier - Vaults

A vault is a container for storing archives. When you create a vault, you specify a vault name and a region in which you want to create the vault.

- Unlimited number of archives in a vault.
- An AWS account can create up to 1,000 vaults per region
- Deleting a vault:
 - If there are no archives in the vault as of the last inventory that Amazon Glacier computed
 - No writes to the vault since the last inventory

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

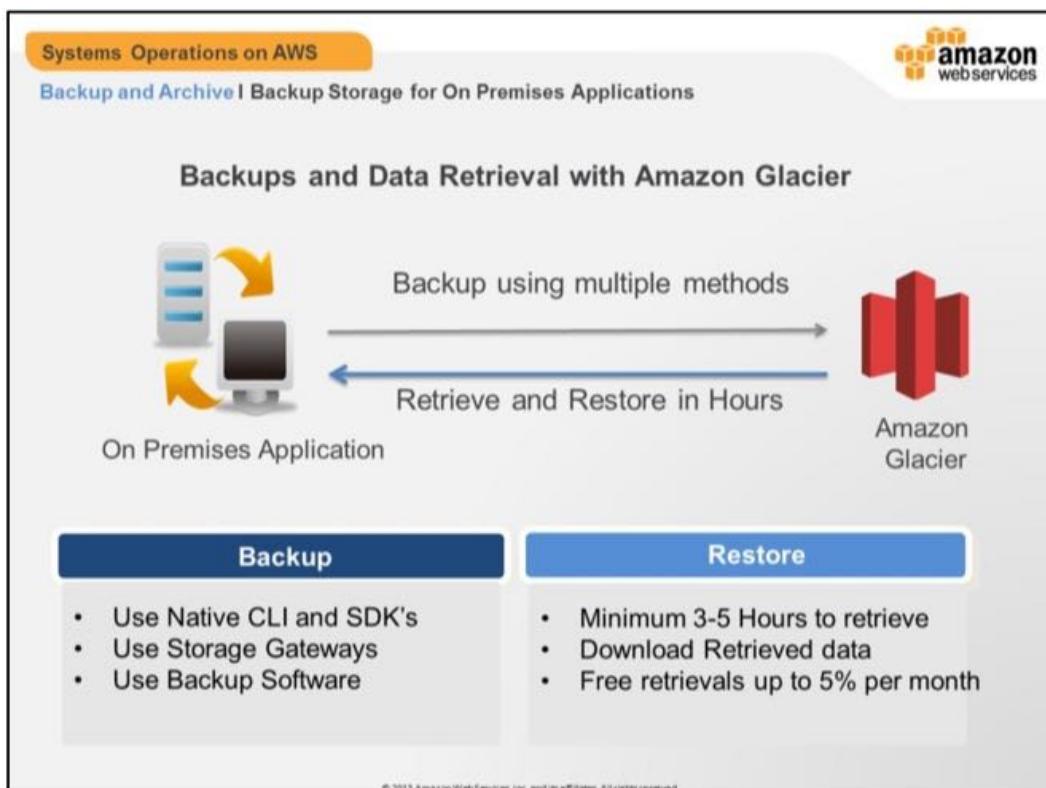
A vault is a container for storing archives. When you create a vault, you specify a vault name and a region in which you want to create the vault.

- When you create a vault, you specify a vault name and a region in which you want to create the vault.
- You can store an unlimited number of archives in a vault.
- An AWS account can create up to a thousand vaults per region.
- And you can delete a vault only if there are no archives in the vault as of the last inventory that Amazon Glacier computed and there have been no writes to the vault since the last inventory.
- When you create a vault, you specify a vault name and the region in which you want to create the vault.



AWS as Backup Storage for On-Premises Applications

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Backup and Archival to Glacier

Amazon Glacier is a low-cost storage service that provides secure, durable, and flexible storage for data backup and archival. Amazon Glacier enables customers to offload the administrative burdens of operating and scaling storage to AWS, so that they don't have to worry about capacity planning, hardware provisioning, data replication, hardware failure detection and repair, or time-consuming hardware migrations.

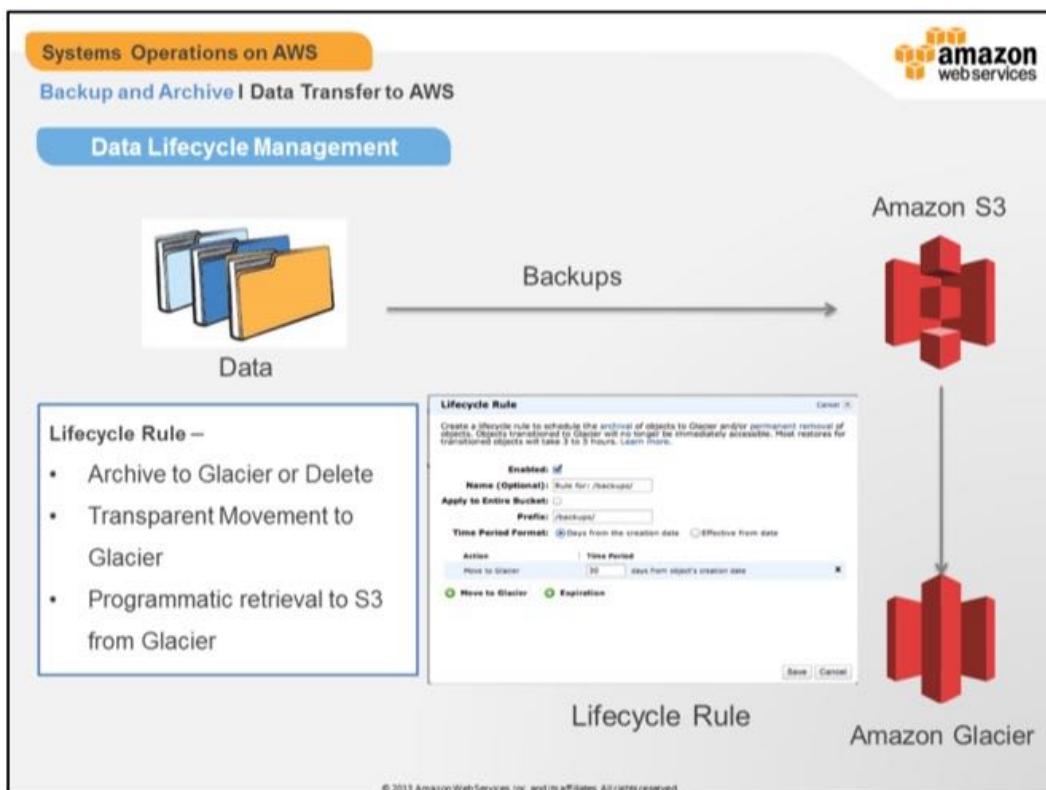
- Customers can use the native AWS CLI and SDK's to backup data in to S3.
- Customers can also use Third party Backup Management software and Storage Gateway appliances manage backups to Glacier.
- Low cost of storage
- Highly Durable Storage – 99.999999999 % durability.
- Data is copied to multiple datacenters automatically. All datacenters are engineered for high levels of redundancy.

- Cost effective and pay-as-you go which makes capacity planning easier
- No Technology refresh required

Restoring Data from Glacier

Restore of data from Glacier is a two step process. These include creating a retrieval request to retrieve data from Glacier to be made available for download and then downloading the data.

- A retrieval request can be programmatically initiated using the AWS SDK, or the Third-party Backup Management Software / Storage Gateways.
- Users can configure retrieval jobs to notify them when the retrieval completes at which point the data can be downloaded.
- It takes 3 -5 hours for the data to be retrieved. This is analogous to the time taken for data to be retrieved from tape archive and made available for the requestor to download the data. Once the retrieval job completes, the data is available for 24 hours
- After retrieval the data needs to be downloaded. For download to and on-premise datacenter, the data transfer out time depends on the available bandwidth between the datacenter and AWS and the method of data transfer used. More will be discussed on the method of data transfer later in the training.
- Up to 5 % of the total data archived can be retrieved free of charge every month at an average rate. Beyond 5% there is a charge depending on the size of data retrieved and the rate at which it is retrieved. Glacier ideally should be used only for infrequently accessed archive data.
- If data is downloaded back to on-premises Standard Data Transfer out charges for AWS apply
- Range retrievals allow for partial retrieval of data from Archives
- Archival metadata information can be stored and indexed in other AWS services like DynamoDB and CloudSearch for retrieval requests.



Lifecycle Management –

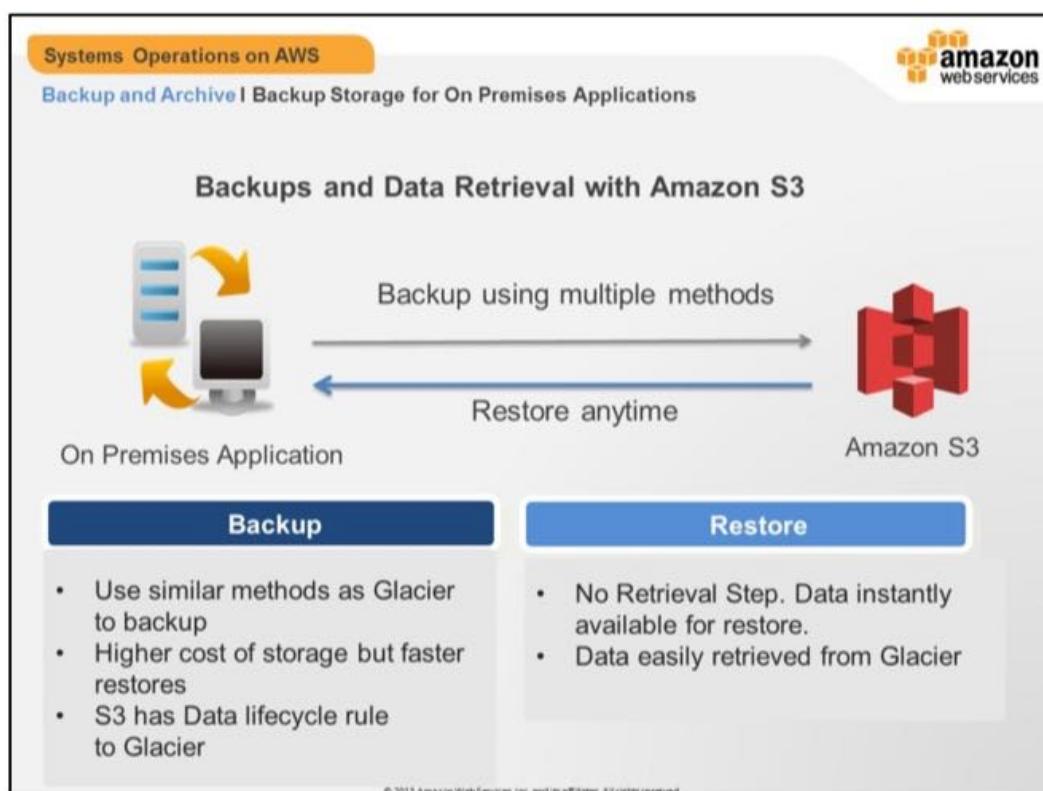
Data life cycle management is the process of classifying and moving data to different storage tiers based on various parameters like frequency of use, or criticality. This is usually done to optimize storage costs. With the amount of data being stored by organizations and people exponentially rising, data lifecycle management has become an important practice in every organization. AWS enables data lifecycle management between AWS S3 Storage and Glacier and deletion. Customers can define how Amazon S3 manages objects during their lifetime.

Some objects that you store in an Amazon S3 bucket might have a well-defined lifecycle

Common examples of data that are candidates for lifecycle management –

- Log files
- Archive documents
- Regulatory Compliance Data that needs to be archived
- Digital media archives

- Email
- Financial and healthcare records
- Raw experimental data
- Long-term database backups
- Data that must be retained for regulatory compliance



Amazon S3 is a simple storage service that offers software developers a highly-scalable, reliable, and low-latency data storage infrastructure at very low costs. S3 can also be used for Backing up your data to AWS, and used in conjunction with Glacier using data lifecycle management, customers can architect a highly effective backup management system that balances cost and performance.

Backup to S3

- Natively Customers can use AWS SDK's or CLI tools to backups data into S3.
- Similar to Glacier, Third party backup software and Storage Gateway appliances are available.
- Higher cost as compared to Glacier but faster restores as data is instantly available for restore.
- In-Built Data Lifecycle Management to Glacier and Deletion of data to lower overall storage costs.
- Highly Durable Storage – 99.999999999 % durability. Data stored in multiple datacenters.
- Cost effective and Pay as you go which makes capacity planning easier.
- No Tech refresh required ever.

Restoring Data from S3

With Amazon S3 you can retrieve data anytime

- These retrievals can be programmatically initiated
- Data is instantly available for retrieval
- With Data lifecycle management, Data archived in Glacier can be retrieved to AWS S3 Reduced Redundancy Storage (RRS) and then restored.
- Backup metadata information can be stored and indexed in other AWS services like DynamoDB and CloudSearch .

Trainer Notes

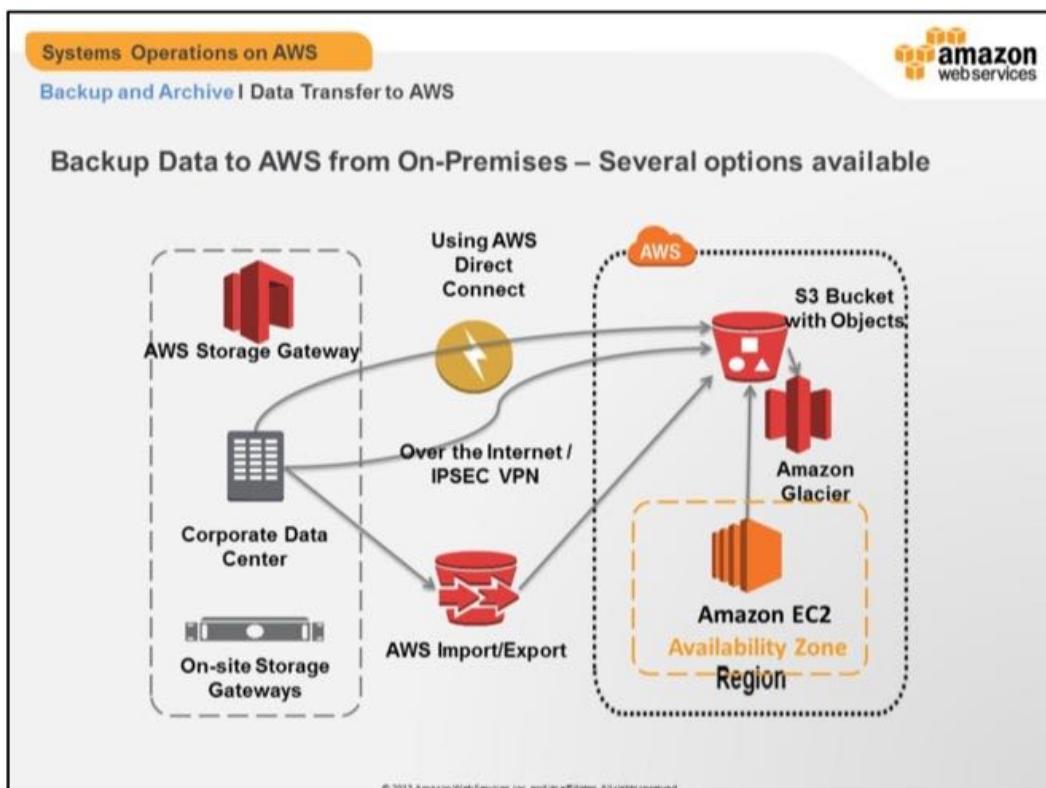
Examples - Riverbed Whitewater, CTera. Some appliances support de-duplication that reduce the overall cost of backups. There are other specialized appliances for specific use cases for example Sonian for email archives into AWS. Enterprise Backup software that support backups to S3 are Netbackup, Commvault, Oracle RMAN etc.

It will be helpful to draw an analogy of S3 and Glacier to the currently popular backup methodology of having disk based backups using technologies like VTL for recent backups and tape archives for long term backup and archival.



How is Data Transferred From On-Premises to AWS

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Transferring Data

For a Hybrid model where On-Premise systems need to use AWS as a backup infrastructure, usually an area of concern is how to transfer data to the AWS Cloud infrastructure. Depending on the Volume and Velocity of transfer required there are several options to transferring data backups into backup to Amazon and archival to Amazon S3/Glacier.

Methods to Transfer

1. Over the Internet
2. Over the AWS IPSEC VPN Gateway
3. AWS Direct Connect – Dedicated 1 GB or 10 GB connection via a partner colo datacenter
4. AWS Import/Export – Data transfer via Shipped media

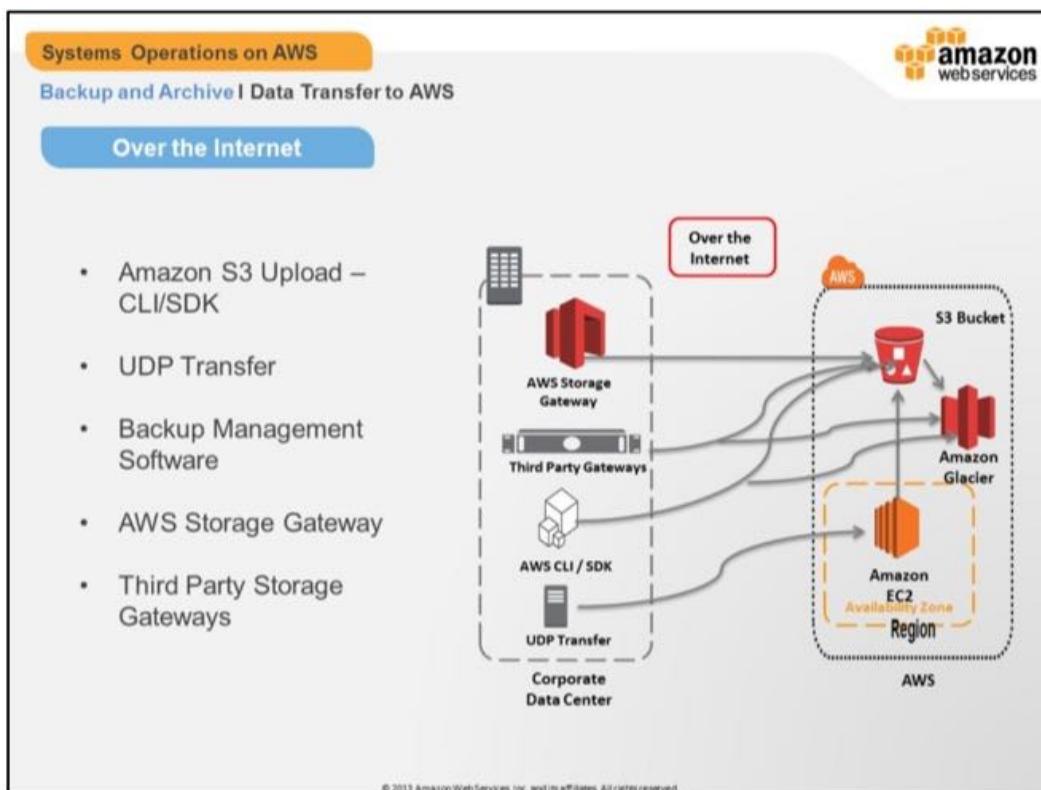
Tools used

1. AWS Storage Gateway – Snapshot on-premise volumes into the cloud

via Internet or Direct Connect

2. Third Party Gateways – Multiple Third party gateway support for snapshots, data transfer into the cloud via internet or Direct connect.
3. Amazon SDK's

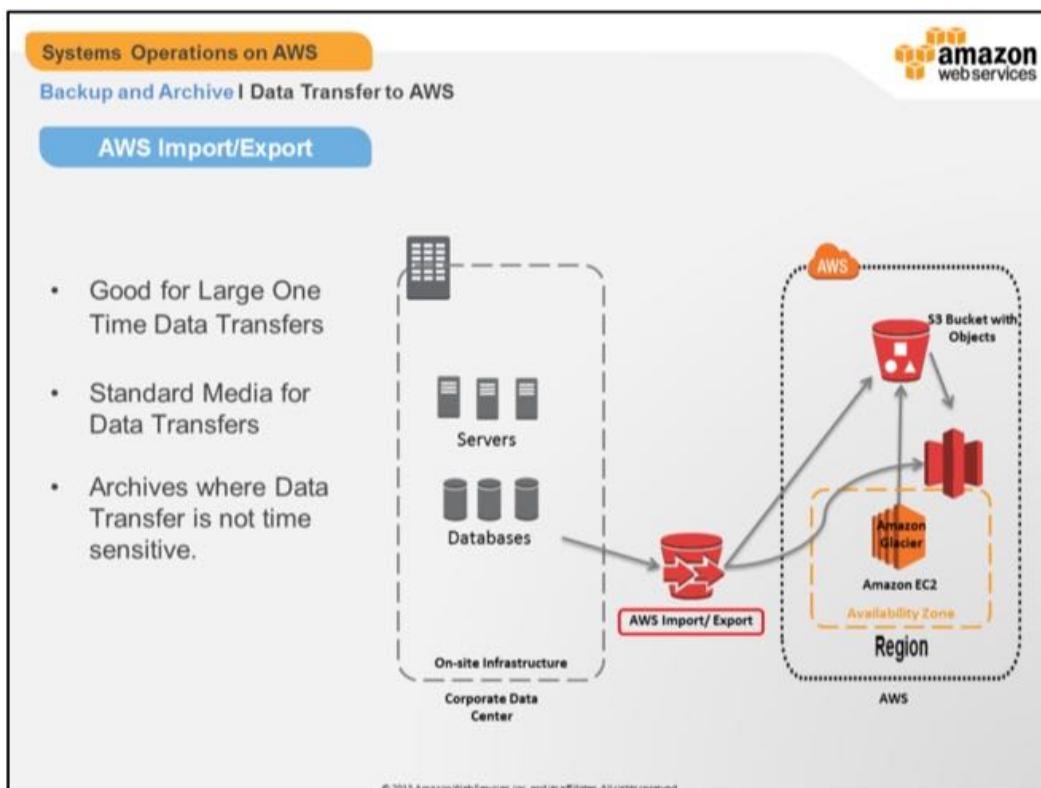
Details of these options will be discussed in the coming slides.



Transfer Data over the internet - Transferring data to and from Amazon S3 is quite done via the internet, and is therefore accessible from any location. Granular S3 bucket policies enable this data to be secure in AWS. There are many commercial and open source backup solutions which backup to Amazon S3. Commonly used ones are

- S3 Multipart Upload through AWS CLI and SDK
- UDP protocol transfers for faster throughput of data over the network. - Tsunami UDP , Aspera
- Backup Management software that can move data directly into S3.
- Commercial Storage gateways that also allow for de-duplicated transfers to S3 –

UDP Protocol Transfers to AWS – Using TCP as a transport protocol over higher latency networks can lead to lower data transfer rates and low utilization of the available network bandwidth. Software like Aspera and Tsunami UDP (Open Source) transfer data by dividing the source files into chunks and using the UDP protocol to transfer data and better utilize the network bandwidth. It is common to get 900 Mbps throughput on a 1 Gbps network. This usually involves setting up on-premise servers and EC2 instances as it is not possible to send this UDP traffic directly to S3.



Transfer Data over the AWS Import Export- The AWS Import/Export service enables transfers of very large data sets by shipping storage devices directly to AWS. AWS teams will import the data into your Amazon S3 and Amazon Glacier.

AWS Import/Export accelerates transferring large amounts of data between the AWS cloud and portable storage devices that you mail to us. AWS transfers data directly onto and off of your storage devices using Amazon's high-speed internal network. Your data load typically begins the next business day after your storage device arrives at AWS. After the data export or import completes, we return your storage device. For large data sets, AWS Import/Export can be significantly faster than Internet transfer and more cost effective than upgrading your connectivity.

AWS Import/Export supports:

- Import to Amazon S3
- Export from Amazon S3
- Import to Amazon EBS Snapshots
- Import to Amazon Glacier

How does it work ?

The AWS Import/Export command line tool can be used to create an AWS Import/Export Job

The AWS SDK or the Web Service API too can be used.

Each job corresponds to one storage device

Multiple storage devices can be shipped in one shipment

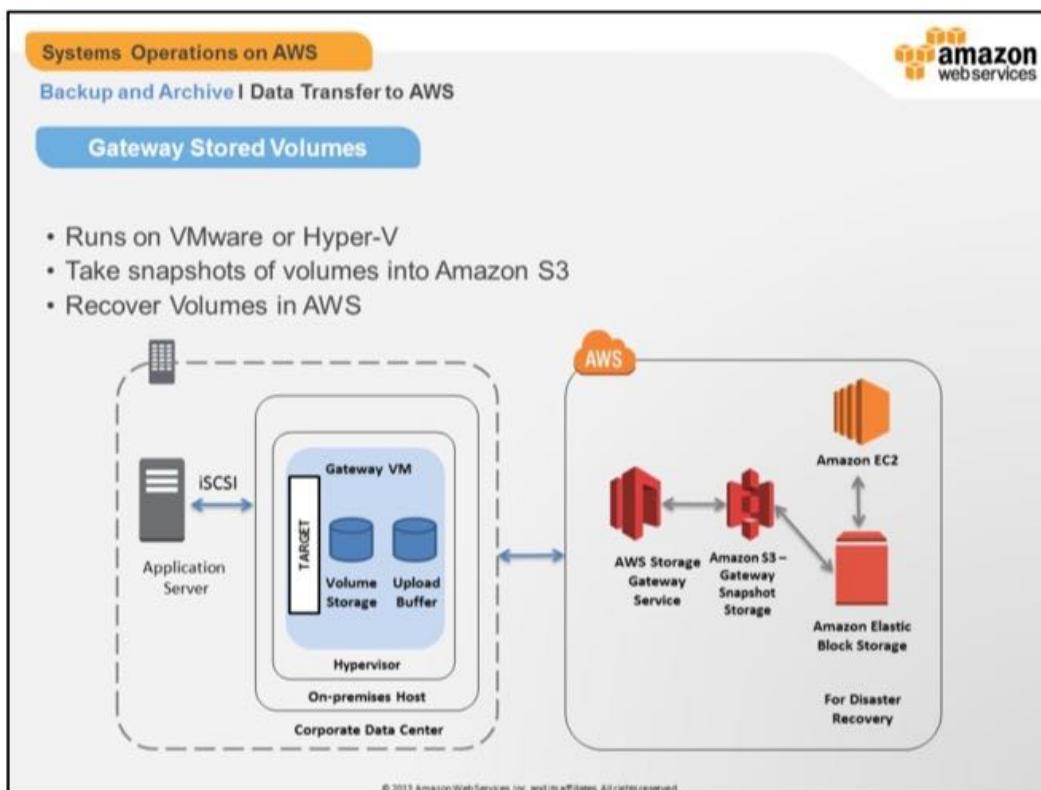
You can track the jobs using the command line tool or the API

The below formats are allowed

eSATA

USB 2.0 and 3.0 (including USB flash drives)

6.4-cm (2.5 inch) and 8.9-cm (3.5 inch) internal SATA hard drives

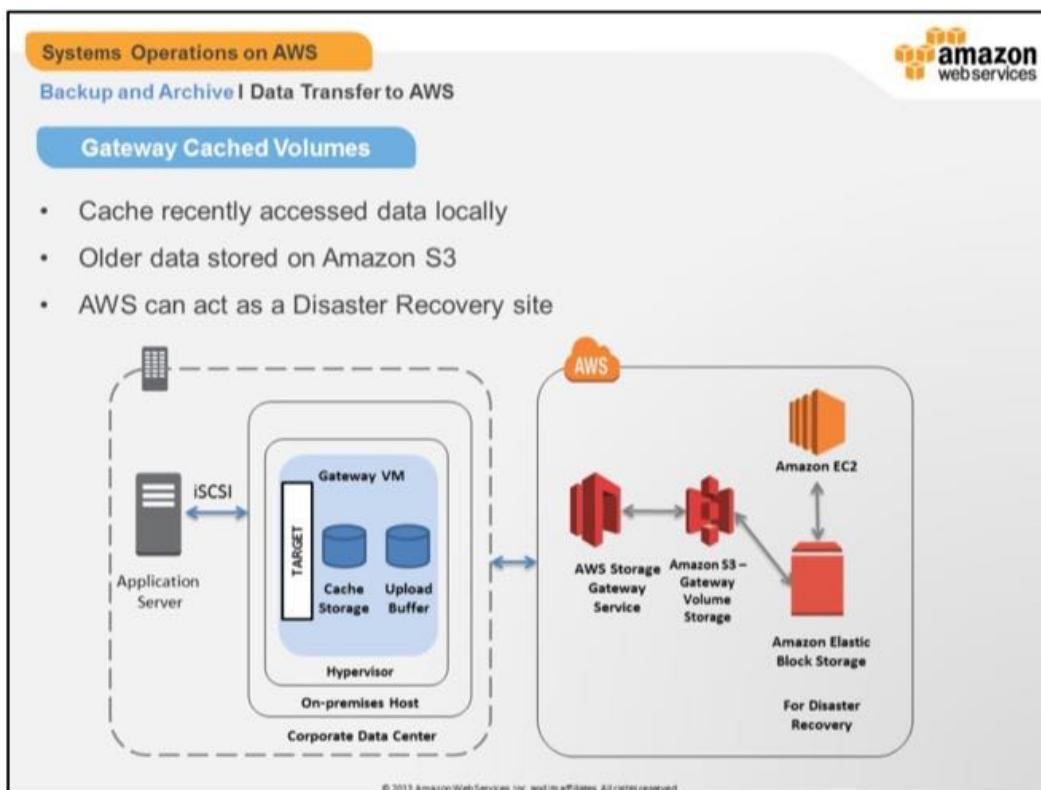


AWS Storage Gateway service architecture enables integration between your organization's on-premises IT environment and AWS's storage infrastructure. The Storage Gateway is a Virtual Machine appliance that runs on an on-premises virtual environment (currently VMware or Hyper – V). The Gateway exposes one or more iSCSI storage targets to other servers in your environment. In the background the Storage Gateway synchronizes the disks to AWS through two methods – Gateway Stored Volumes and Gateway Cached Volumes.

Gateway-stored volumes

Gateway-stored volumes enable you to store your primary data locally, while asynchronously backing up that data to AWS. Gateway-stored volumes provide your on-premises applications with low-latency access to their entire data sets, while providing durable, off-site backups. You can create storage volumes up to 1 TiB in size and mount them as iSCSI devices from your on-premises application servers. Data written to your gateway-stored volumes is stored on your on-premises storage hardware, and asynchronously backed up to Amazon S3 in the form of Amazon EBS snapshots.

Gateway-stored volumes can range from 1 GiB to 1 TiB in size and must be rounded to the nearest GiB. Each gateway configured for gateway-stored volumes can support up to 12 volumes and a total volume storage of 12 TiB.



Gateway-cached volumes, the iSCSI targets are actually in S3 and recent data is cached in a local volume. This enables you to utilize Amazon S3 as your primary data storage while retaining frequently accessed data local in your AWS Storage Gateway. Gateway-cached volumes minimize the need to scale your on-premises storage infrastructure, while still providing your applications with low-latency access to their frequently accessed data. You can create storage volumes up to 32 TiB in size and attach to them as iSCSI devices from your on-premises application servers. Data written to these volumes is stored in Amazon S3 and retained along with recently read data in your on-premises AWS Storage Gateway's cache and upload buffer storage.

Gateway-cached volumes can range from 1 GiB to 32 TiB in size and must be rounded to the nearest GiB. Each gateway configured for gateway-cached volumes can support up to 20 volumes and a total volume storage of 150 TiB.

Systems Operations on AWS

Backup and Archive | Data Transfer to AWS

Backup On-Premise Data to AWS - Considerations

- Distance between sites
- Amount of backup data to be transferred
- Available bandwidth
- Data Transfer - Technology used

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

AWS provides low cost storage options to backup data into the cloud. However, if the customers primary data is on-premise, then designing a solution to leverage the AWS cloud as a backup storage location needs careful consideration of the below points as they have significant impact on the overall solution and cost.

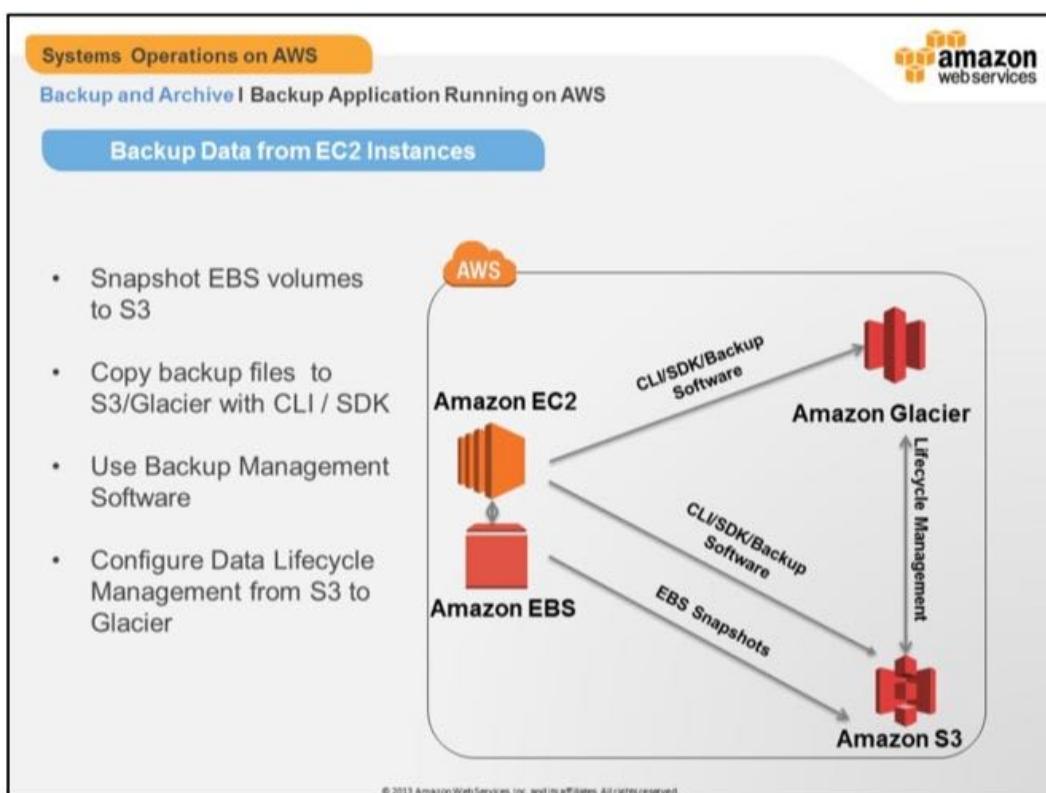
- Distance between the AWS and the Customer datacenter: larger distances typically are subject to more latency and/or jitter. While backing up to AWS, customers should choose a region closer to the Customer datacenter.
- Amount of Data to be transferred. Depending on this, your choice of tools and technologies that will be used will vary. Smaller volumes of data can be transferred via the AWS command line tools or SDK's, but as the volume grows, additional choices need to be considered like UDP Transfers, using third party storage gateways with support for compression and de-duplication. Larger bandwidth may need to be provisioned. Dedicated bandwidth options like Direct connect should be considered.
- Bandwidth availability and the timeframe it is required. Periodic

large data transfers can be done via AWS Import export and there is no need to invest in large bandwidth.



Backing Up Data Hosted in AWS

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

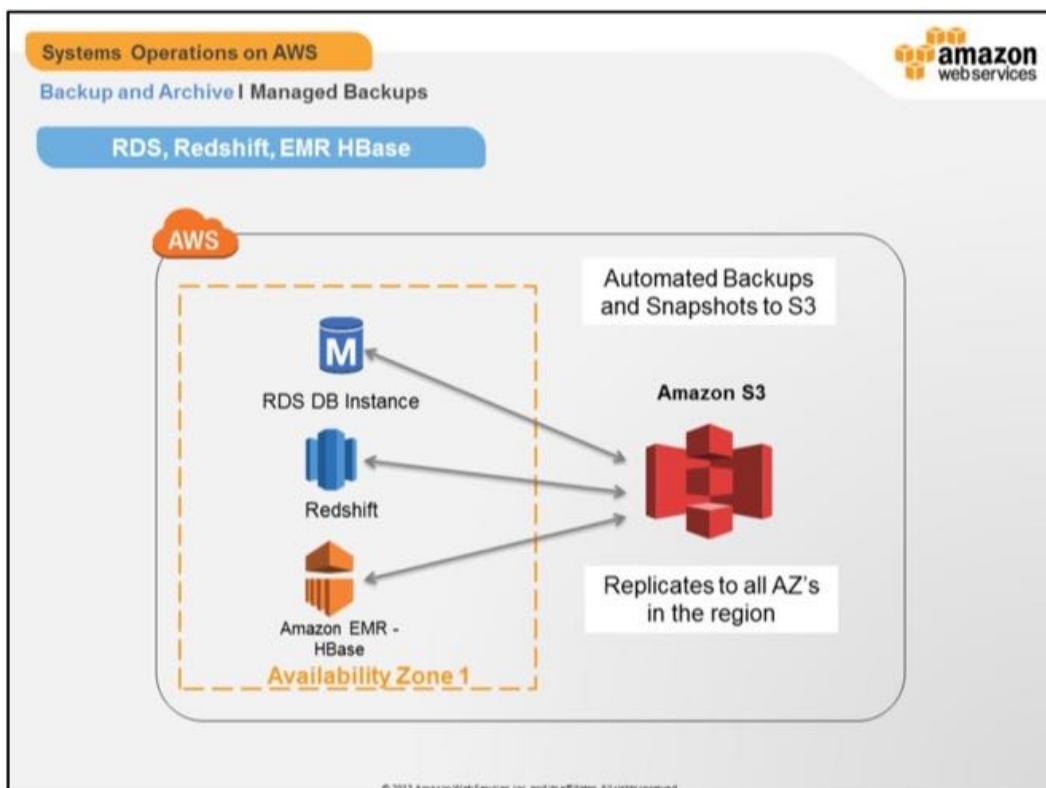


Snapshots: Snapshots taken of Amazon Elastic Block Store (EBS) volumes and backups of Amazon RDS are directly stored in Amazon S3. These snapshots can be restored as EBS volumes. S3 provides a highly redundant storage for backups. All objects stored in S3 are copied across multiple availability zones in the region.

Direct copy to S3 - Alternatively, you can copy files directly into Amazon S3, or you can choose to create backups of your files and databases to copy them to Amazon S3.

Third Party Backup Management Software - Several third party backup software support managed backups from EC2 to S3 or Glacier.

Data Lifecycle Management – Data Lifecycle Management can be configured on S3 to Archive to Glacier or Delete.



Some AWS services provide fully managed built-in Backup functionality that can be configured as per customer requirements. Some of these services are – AWS RDS (Relational Database Services), AWS Redshift, and AWS Hbase on EMR.

- Automated backups and
- Database snapshots (DB Snapshots)

The automated backup feature of Amazon RDS enables point-in-time recovery of your DB Instance. When automated backups are turned on for your DB Instance, Amazon RDS automatically performs a full daily snapshot of your data (during your preferred backup window) and captures transaction logs (as updates to your DB Instance are made). When you initiate a point-in-time recovery, transaction logs are applied to the most appropriate daily backup in order to restore your DB Instance to the specific time you requested.

Amazon RDS retains backups of a DB Instance for a limited, user-specified period of time called the retention period, which by default is one day but can be set to up to thirty five days. You can initiate a point-in-time restore and specify any second during your retention period, up to the Latest Restorable

Time.

DB Snapshots are user-initiated and enable you to back up your DB Instance in a known state as frequently as you wish, and then restore to that specific state at any time. DB Snapshots can be created with the AWS management Console or API's

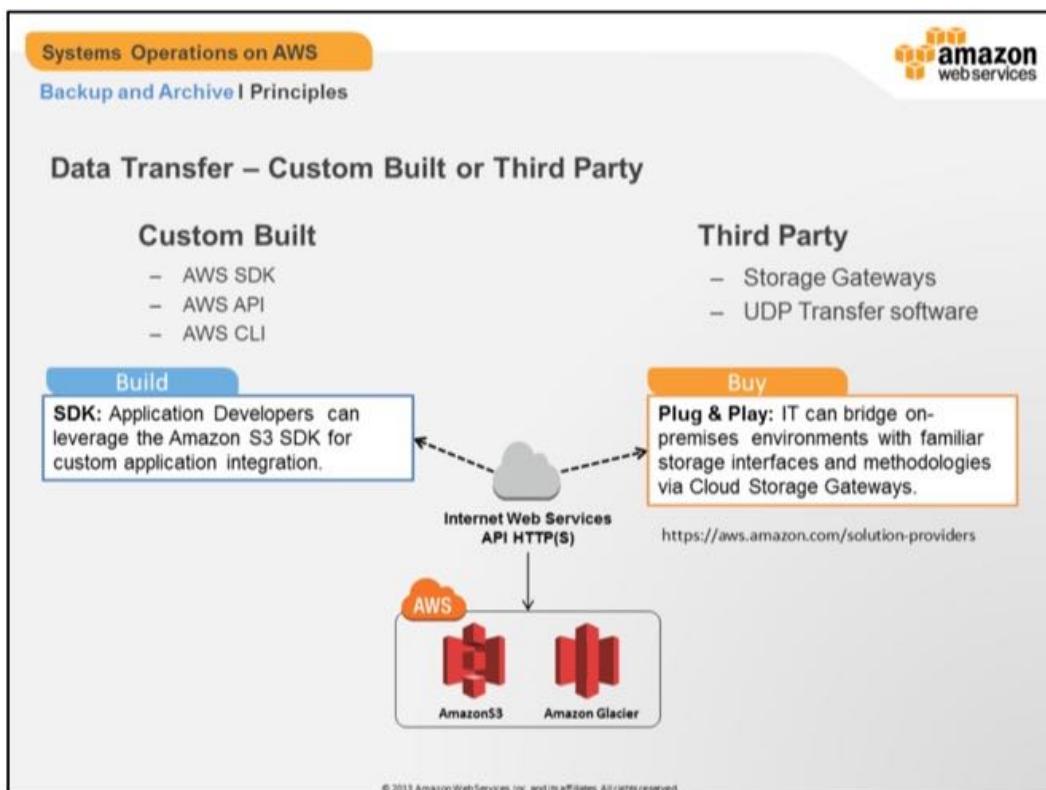
Similar to RDS, Redshift continuously backs up data to S3. Backups are retained for a limited, user-specified period of time called the retention period, which by default is one day but can be set to up to thirty five days. You can also take User Initiated point-in-time snapshots of Redshift.

The HBase offering from EMR too has an option of backing up data. You do a user-initiated one-time backup or a scheduled incremental backup to an S3 bucket. You can specify if the backup should be a consistent backup, which pauses the writes to the database during the backup.



How to Choose the Best Option for Backups

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.



There are two broad options for data transfer to AWS based on what an organization needs.

1. Custom Build – AWS Services are exposed as REST-ful API's and customers can custom build code to transfer data to AWS
2. Third Party – Customers can use third party software and appliances to transfer data to AWS

Pros and Cons		
Backup and Archive Data Transfer Options		
Option	Category	Pros and Cons
AWS SDK's and CLI tools	Backup Management	Pro - No Extra Cost Con – Increased management overhead. Backup policies management and Automation as to be done by the customer.
Third Party Backup software	Backup Management	Pro – Familiar backup management tools for Admins. Inbuilt Policy based backup management Con – Increased cost for Backup Management Software
AWS Storage Gateway	Backup Management	Pro – Native Block level backups to AWS , Cost Effective Con - Some features available in Third party gateways are not available – e.g. deduplication
Third Party Storage Gateways	Backup Management	Pro – Additional features that improve performance like De-Duplication Con – Increased cost of solution
UDP Transfers	Data Transfer	Pro – Effective Bandwidth Utilization, decreased transfer cost Con – Need to provision EC2 instances for transferring data, Increased cost for commercial

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

There are many options for taking backups to AWS and each one has its Pro's and Con's. Users should choose the best option that suits their needs.

AWS SDK's and CLI tools can be used by customers to write their own backup management scripts, by either uploading data to AWS S3/Glacier, or take snapshots of EBS Volumes.

Customers looking for a more managed approach with tools can opt for third party backup management software that manages backups to S3.

AWS Storage Gateway supports native Block level Snapshots to AWS S3.

For additional features than the AWS Storage gateway, customers can use third party storage gateway appliances.

UDP Transfers are a good options for data transfers, if you want to maximize utilization of the bandwidth. There are commercial and opensource UDP Transfer solutions that are available.

Systems Operations on AWS

Backup and Archive I Summary

amazon
webservices

- Two perspectives when talking about backups on AWS
 - Use AWS as backup and archival storage for on-premises Applications
 - Backup and Archive for applications running on AWS
- Many options for Backups on AWS with varying Data Velocity, Data Types and Cost
- Customers can use Native AWS provided tools or Third Party software and appliances
- Data Lifecycle management can be used for Archival management
- Customers can choose optimal mix of tools and methods per their backup and archival requirements



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- There are two perspectives when talking about backups on AWS
 1. Use AWS as backup and archival storage for on-premises Applications
 2. Backup and Archive for applications running on AWS
- There are many options for Backups on AWS with varying Data Velocity, Data Types and Cost
- Customers can use Native AWS provided tools or Third Party software and appliances
- Data Lifecycle management can be used for Archival management
- Customers can choose the optimal mix of tools and methods as per their backup and archival requirements



LAB

Backup

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Backup and Archive | Lab Exercise

Lab Objective

- Backing up a MySQL Database to Amazon S3 and restoring data to a required point in time.
- Configure backups in Amazon RDS MySQL instance and do a point in time restore.
- Leverage consistent EBS snapshots for backup and restoring.



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The lab for this module has three parts:

- Backing up a MySQL Database to S3 and restoring data to a required point in time.
- Configure backups in Amazon RDS MySQL instance and do a point in time restore.
- Leverage consistent EBS snapshots for backing and restoring.



Module 9: Operations Security

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Operations Security I Overview

Learning Objectives

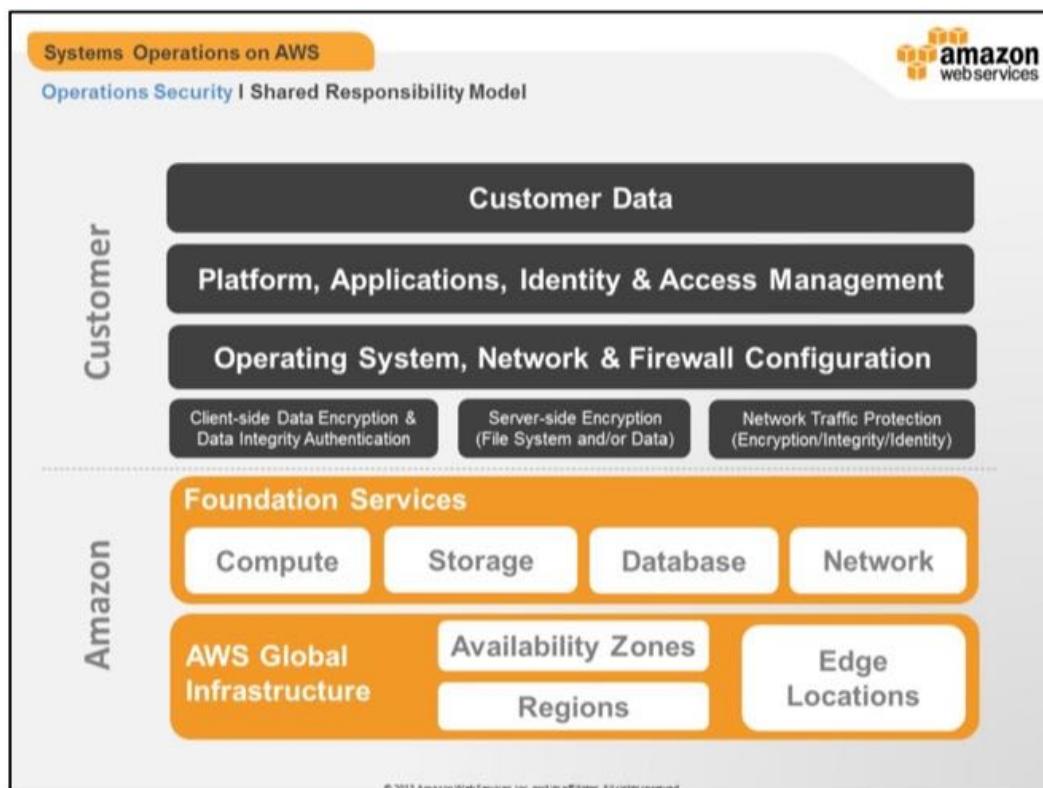
After completing this module you should be able to:

- Enforce your organization's policy using AWS features.
- Understand AWS access policy structure, components, and uses.
- Gain AWS native event logging capabilities.
- Follow best practices for configuring and managing AWS security controls.
- Understand encryption options in AWS.
- Follow the procedures to perform vulnerability scans and penetration testing.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

After completing this module you should be able to:

- Enforce your organization's policy using AWS features.
- Understand AWS access policy structure, components, and uses.
- Gain AWS native event logging capabilities.
- Follow best practices for configuring and managing AWS security controls.
- Understand encryption options in AWS.
- Follow the procedures to perform vulnerability scans and penetration testing.



A quick reminder of the shared responsibility model.

AWS is responsible for: Security **OF** the cloud

The customer is responsible for: Security **IN** the cloud

For this course, we'll be focusing on what you, the Customer, can do to secure your applications in the cloud. Because you're building systems on top of the AWS cloud infrastructure, the security responsibilities will be shared: AWS manages the underlying infrastructure but you must secure anything you put on the infrastructure. This includes your AWS EC2 instances and anything you install on them, any accounts that access your instances, the security group that allows outside access to your instances, the VPC subnet that the instances reside within if you've chosen this option, the external access to your S3 buckets, etc. This means that there are several security decisions you need to make and controls you must configure.

Customers

- Customers implement their own set of controls
- Everything from the hypervisor up
- We provide security tools that you must configure properly to meet your needs and maintain a strong security posture
- Can Amazon perform Windows Updates for your Instances? No. Because we have zero visibility into the Operating System layer
- In Federal, we have multiple customers with FISMA Low and Moderate Authority

to Operate (ATO)

Amazon / AWS

Not only are your applications and data protected by highly secure facilities and infrastructure, but they're also protected by extensive network and security monitoring systems. These systems provide basic but important security measures such as distributed denial of service (DDoS) protection and password brute-force detection on AWS Accounts. What we do:

- Physical and Environmental Security
 - Fire detection & suppression
 - Redundant power
 - Climate & temperature
 - Management of electrical, mechanical, life support systems
 - Storage device decommissioning (AWS uses the techniques detailed in DoD5220.22-M ("National Industrial Security Program Operating Manual") or NIST 800-88 ("Guidelines for Media Sanitization"))
- Business Continuity Management
 - Availability
 - Incident response
 - Company-wide executive review
 - Internal communication & training
 - External communication & alerting
- Network Security
 - Secure network architecture
 - Secure access points (HTTPS, FIPS 140-2 level 2 VPN endpoints and SSL-terminating load balancers in AWS GovCloud)
 - Transmission protection (every message to AWS API must be authenticated)
 - Amazon corporate segregation (separation of duties between logical/physical)
 - Fault-tolerant design
 - Network monitoring & protection (Protected against DDoS, MITM, IP spoofing, port scanning, packet sniffing by other tenants)
- AWS Access
 - Account review and audit (90 day explicit re-approval)
 - Background checks
 - Password policy
 - Multifactor Authentication (MFA)
- Secure Design Principles
 - Secure software development
 - Formal design reviews
 - Threat modeling
 - Risk assessments
 - Static code analysis

- Penetration testing
- Change Management
 - Systematic: review, test, approve, communicate
 - Phased deployment starting with lowest impact areas
 - Metrics for impact, health, thresholds, alarming
 - Rollback procedures in CM tickets
 - Change windows (when possible)
 - Root cause analysis

Systems Operations on AWS

Operations Security | Third Party Credentials & Attestations



- **AWS environment:**
 - Service Organization Controls 1, 2, and 3 Audits (formerly SAS70)
 - ISO 27001 Certification
 - Payment Card Industry Data Security Standard (PCI DSS) Level 1 Service Provider
 - FedRAMP
 - DIACAP and FISMA
 - International Traffic in Arms Compliance
 - FIPS 140-2
 - Cloud Security Alliance – Security, Trust & Assurance Registry
 - GovCloud
- **Customers have deployed various compliant applications:**
 - Sarbanes-Oxley (SOX)
 - HIPAA (healthcare)
 - FISMA (US Federal Government)
 - DIACAP MAC III Sensitive IATO
 - International Traffic in Arms Regulations (ITAR)

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

We know that it's important for you to understand the protection measures that are used to guard the AWS cloud infrastructure. But since you can't physically touch the servers or walk through the data centers, how can you be sure that the right security controls are in place?

The answer lies in the third-party certifications and evaluations that AWS has undergone. AWS has achieved ISO 27001 certification and has been validated as a Level 1 service provider under the Payment Card Industry (PCI) Data Security Standard (DSS). We undergo annual SOC 1 audits and have been successfully evaluated at the Moderate level for Federal government systems as well as DIACAP Level 2 for DoD systems.

Each certification means that an auditor has verified that specific security controls are in place and operating as intended. You can view the applicable compliance reports by contacting your AWS account representative.



Policy Enforcement

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Operations Security | Policy Enforcement

Access Policy Language

- Access Policies may be applied to AWS resources
- Written as a JSON document
- Contains one or more individual statements
- Can contain conditions to provide granular restrictions
 - Source IP address
 - Secure Transport (SSL)
 - User Agent
 - Current Time
 - And more...

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

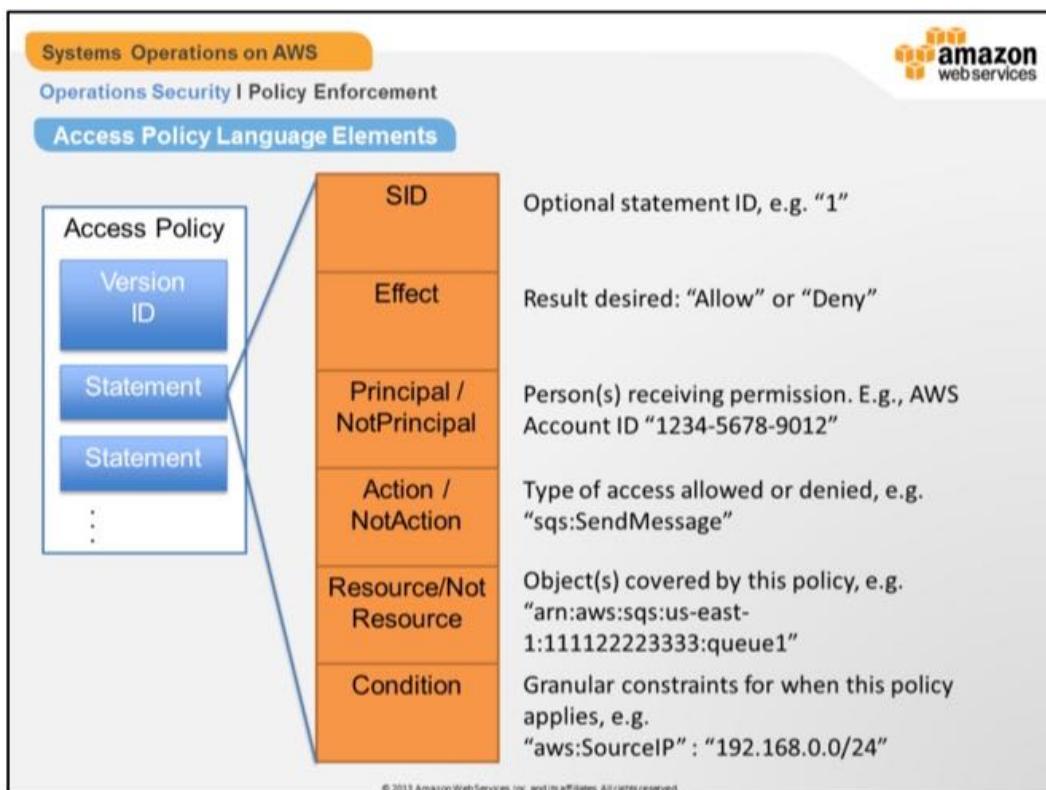
Let's dive into the access policy language so we can understand its structure, capabilities and applications. We'll cover syntax, descriptions, and examples of the elements, variables, and evaluation logic.

To assign permissions to a user, group, role, or resource, you create a policy, which is a document that explicitly lists permissions. In its most basic sense, a policy lets you specify the following:

- Actions: what actions you will allow. Each AWS service has its own set of actions. For example, you might allow a user to use the Amazon S3 ListBucket action, which returns information about the items in a bucket. Any actions that you don't explicitly allow are denied.
- Resources: which resources you allow the action on. For example, what specific Amazon S3 buckets will you allow the user to perform the ListBucket action on? Users cannot access any resources that you have not explicitly granted permissions to.
- Effect: what the effect will be when the user requests access—either allow or deny. Because the default is that resources are denied to users, you typically specify that you will allow users access to resource.

Policies are documents that are created using JSON. A policy consists of one or

more statements, each of which describes one set of permissions.



The structure of a policy is shown here. A policy includes:

- Optional policy-wide information (at the top of the document)
- One or more *statements*

Each statement includes the core information about a single permission. If a policy includes multiple statements, we apply a logical *OR* across the statements at evaluation time. If multiple policies are applicable to a request, we apply a logical *OR* across the policies at evaluation time.

The information in a statement is contained within a series of *elements*.

Sid

The *Sid* (statement ID) is an optional identifier that you provide for the policy statement. You can assign a *Sid* value to each statement in a statement array. In services that let you specify an *ID* element, such as SQS and SNS, the *Sid* value is just a sub-ID of the policy document's ID. In IAM, the *Sid* value must be unique within a policy.

In IAM, the *Sid* is not exposed in the IAM API. You can't retrieve a particular statement based on this ID.

Note: Some AWS services (for example, Amazon SQS or Amazon SNS) might

require this element and have uniqueness requirements for it. For service-specific information about writing policies, refer to the documentation for the service you're working with.

Effect

The *Effect* element is required and specifies whether you want the statement to result in an allow or an explicit deny. Valid values for *Effect* are *Allow* and *Deny*.

Principal

The *Principal* element specifies the user, account, service, or other entity that is allowed or denied access to a resource.

For IAM users and groups, you do not specify a principal. In those cases, the principal is the user whose credentials are used to make the request. (If the policy is attached to a group, the principal is the member of the group that's making the request.) For IAM roles, principals are used to specify who can assume the role. In other services, principals are used in policies that are attached to a resource, such as an Amazon S3 bucket or an Amazon SQS queue.

You specify a principal using the *Amazon Resource Name* (ARN) of the account, IAM user, or role. (Note that IAM groups cannot be specified as principals.)

In IAM, you can use wildcards (*) to indicate all possible users. As a shortcut, if you're specifying an account ID, you can use a shortened form that consists of the AWS: prefix followed by the ID.

NotPrincipal

The *NotPrincipal* element lets you specify an exception to a list of principals. For example, you can use this to prevent all AWS accounts *except* a specific account from accessing a resource. Conversely, you can deny access to all principals *except* the one named in the *NotPrincipal* element. As with *Principal*, you specify the user or account that should be allowed or denied permission; the difference is that *NotPrincipal* translates to everyone *except* that person or account.

Action

The *Action* element describes the type of access that should be allowed or denied (for example, read, write, list, delete, and so on). Statements must include either an *Action* or *NotAction* element. Each AWS service has its own set of actions that describe tasks that you can perform with that service.

You specify a value using a namespace that identifies a service (*iam*, *sqs*, *sns*, *s3*, etc.) followed by the name of the action to allow or deny. The name must match an action that is supported by the service. The prefix and the action name are case insensitive.

Notes about Action:

- You can specify multiple values for the *Action* element.
- You can use a wildcard (*) to give access to all the actions the specific AWS product offers. For example, the following *Action* element applies to all IAM actions.
- You can also use wildcards (*) or (?) as part of the action name. For example, the following *Action* element applies to all IAM actions that include the string *AccessKey*, including *CreateAccessKey*, *DeleteAccessKey*, *ListAccessKeys*, and *UpdateAccessKey*.
- Some services let you limit the actions that are available. For example, Amazon SQS lets you make available just a subset of all the possible Amazon SQS actions. In that case, the * wildcard doesn't allow complete control of the queue; it allows only the subset of actions that you've shared.

NotAction

The *NotAction* element lets you specify an exception to a list of actions. For example, you can use *NotAction* to let users use only the Amazon SQS *SendMessage* action, without having to list all the actions that the user is not allowed to perform. Using *NotAction* can sometimes result in shorter policies than using an *Action* element and listing many actions.

Resource

The *Resource* element specifies the object or objects that the statement covers. Statements must include either a *Resource* or a *NotResource* element. You specify a resource using an ARN. Each service has its own set of resources. Although you always use an ARN to specify a resource, the details of the ARN for a resource depend on the service and the resource. For information about how to specify a resource, refer to the documentation for the service whose resources you're writing a statement for.

Note: Some services do not let you specify actions for individual resources; instead, any actions that you list in the *Action* or *NotAction* element apply to all resources in that service. In these cases, you use the wildcard * in the *Resource* element.

NotResource

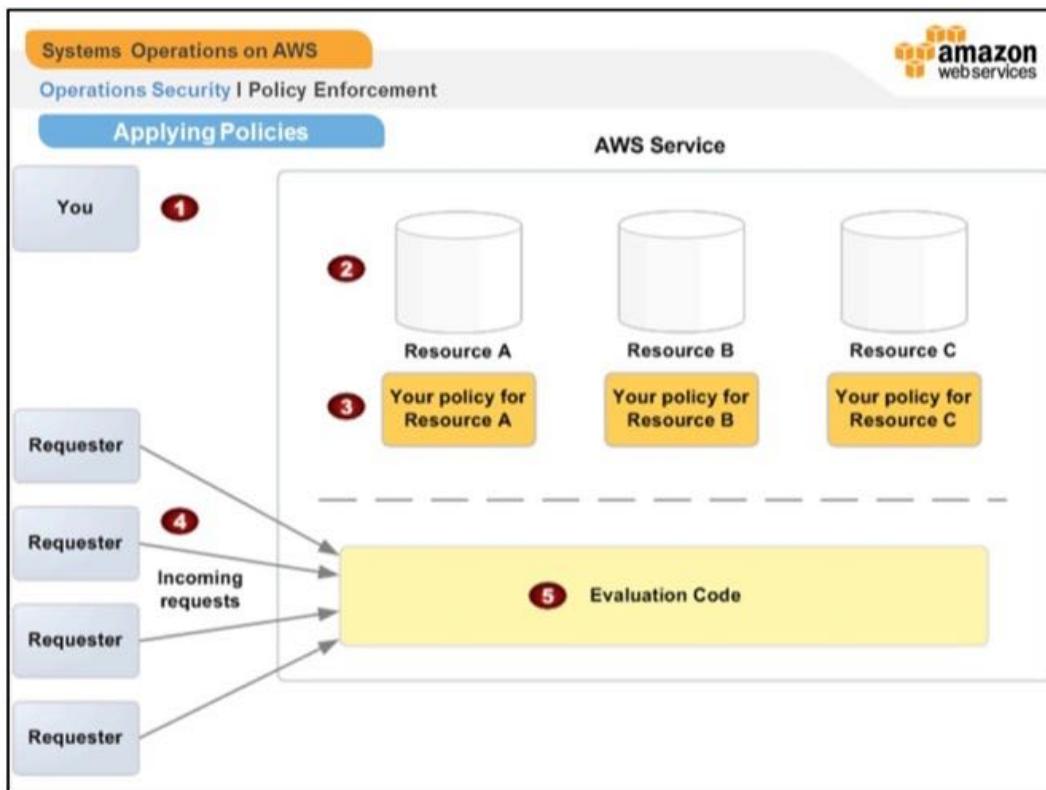
The *NotResource* element lets you grant or deny access to all but a few of your resources, by allowing you to specify only those resources to which your policy should not be applied.

Condition

The *Condition* element lets you specify conditions for when a policy is in effect. The *Condition* element is optional. In the *Condition* element, you build expressions in which you use Boolean operators (equal, less than, etc.) to match your condition

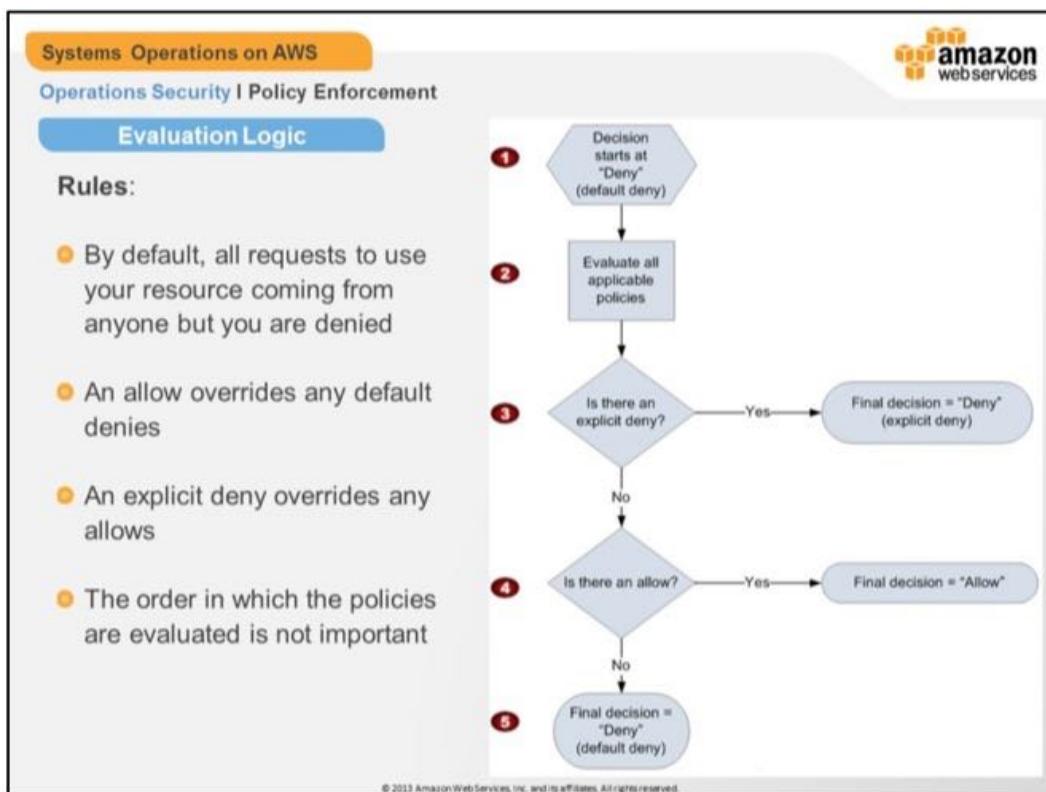
against values in the request. Condition values can include date, time, the IP address of the requester, the ARN of the request source, the user name, user ID, and the user agent of the requester. Some services also let you additional values in conditions; for example, Amazon S3 lets you write a condition using the `s3:VersionId` key, which is unique to that service.

A *Condition* element, or condition block, can contain multiple conditions, and each condition can contain multiple key-value pairs. If there are multiple conditions, or if there are multiple keys in a single condition, the conditions are evaluated using a logical AND. If a single condition includes multiple values for one key, the condition is evaluated using a logical OR. All conditions must be met for an allow or an explicit deny decision. If a condition isn't met, the result is a deny.



Policies are applied by you to each resource. This architectural overview illustrates the main components that interact to provide access control for your resources.

You (1) apply policies (3) to govern access to each resource (2). When requesters (4) request access, the evaluation code (5) is executed to determine access.



1. The decision starts with a default deny.
2. The enforcement code then evaluates those that are applicable to the request (based on the resource, principal, action, and conditions). The order in which the enforcement code evaluates the policies is not important.
3. In all those policies, the enforcement code looks for an explicit deny instruction that would apply to the request. If it finds even one, the enforcement code returns a decision of "deny" and the process is finished (this is an explicit deny; for more information, see Explicit Deny).
4. If no explicit deny is found, the enforcement code looks for any "allow" instructions that would apply to the request. If it finds even one, the enforcement code returns a decision of "allow" and the process is done (the service continues to process the request).
5. If no allow is found, then the final decision is "deny" (because there was no explicit deny or allow, this is considered a default deny (for more information, see Default Deny)).

Systems Operations on AWS

Operations Security | Policy Enforcement

Service Specific Policies

Service	Policy
AWS IAM Access Policies	Controls actions and resources for most services
Amazon SNS	Topic-specific access policies
Amazon SQS	Queue-specific access policies
Amazon S3	Can use ACLs or access policies Bucket-specific policies can allow cross-account access to specific resources

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



IAM Policy:

This was covered in more depth during the IAM learning module. What services are not supported? As of August 2013: Import/Export, EBS, CloudSearch, CloudHSM, DirectConnect, FPS

SNS Policies:

Each policy must cover only a single topic or queue (when writing a policy, don't include statements that cover different topics or queues)

Each policy must have a unique policy Id

Each statement in a policy must have a unique statement sid

SQS Queue Policy:

You don't need to write your own policies if you want to allow access based only on AWS account ID and basic permissions (e.g., SendMessage, ReceiveMessage). In that case, you can just use the SQS AddPermission action. If you want to explicitly deny access or allow it based on finer conditions (such as the time the request comes in or the IP address of the requester), you need to write your own policies and upload them to the AWS system using the SQS SetQueueAttributes action.

- SQS allows you to share only certain types of permissions
- Each policy must cover only a single queue (when writing a policy, don't include statements that cover different queues)
- Each policy must have a unique policy ID (Id)

- Each statement in a policy must have a unique statement ID (sid)
- SQS does not implement any special keys to use when you write conditions; the only keys available are the general AWS-wide keys.

S3 ACLs vs. Access Policies

We'll elaborate on the next slide

Systems Operations on AWS
Operations Security I Policy Enforcement
Amazon S3 Considerations

- Two models for managing permissions directly with S3
 - ACLs
 - Simple, coarse-grained control
 - S3 Bucket and object ACLs are completely independent; an object does not inherit the ACL from its bucket
 - Bucket Policies
 - Comprehensive, fine-grained control
 - Only a bucket owner can write bucket policies
- You can use ACLs and Access Policies together

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

S3 ACLs:

ACLs provide a coarse-grain permission model, where you simply grant access permissions to buckets or objects. Bucket policies, on the other hand, provide fine-grain control over the permissions you are granting. For example, you can write a policy granting users access to a bucket or an object, provided the user sends the request from a specific IP address, or the request arrives after a specific date and time. Depending on your needs, you can use one or both of these permission models.

S3 Bucket Policy:

Bucket policies define access rights for Amazon S3 resources. Only a bucket owner can write bucket policies. A bucket owner can write a bucket policy to:

- Allow/deny bucket-level permissions.
- Deny permission on any objects in the bucket. Because the bucket owner is fiscally responsible for the bucket, the owner can write a bucket policy to deny permissions on any objects in a bucket.
- Grant permission on objects in the bucket only if the bucket owner is the object owner. For objects owned by other accounts the object owner must manage permissions using ACLs.
- When you grant other AWS accounts access to your AWS resources, be aware that the AWS accounts can delegate their permissions to users under their accounts. This is known as cross-account access. For information about using cross-account access, go to

<http://docs.aws.amazon.com/IAM/latest/UserGuide/Delegation.html>

ACLs vs. Bucket Policy:

- ACLs provide permissions for only a few actions (next slide), whereas bucket policies provide permissions for dozens of actions (complete list below).

Amazon S3 Actions

The following list shows the format for the Amazon S3 actions that you can reference in a policy.

Actions Related to Objects

- s3:GetObject (covers REST GET Object, REST HEAD Object, REST GET Object torrent, SOAP GetObject, and SOAP GetObjectExtended)
- s3:GetObjectVersion (covers REST GET Object, REST HEAD Object, REST GET Object torrent, SOAP GetObject, and SOAP GetObjectExtended)
- s3:PutObject (covers the REST PUT Object, REST POST Object, REST Initiate Multipart Upload, REST Upload Part, REST Complete Multipart Upload, SOAP PutObject, and SOAP PutObjectInline)
- s3:GetObjectAcl
- s3:GetObjectVersionAcl
- s3:PutObjectAcl
- s3:PutObjectVersionAcl
- s3>DeleteObject
- s3>DeleteObjectVersion
- s3>ListMultipartUploadParts
- s3:AbortMultipartUpload
- s3:GetObjectTorrent
- s3:GetObjectVersionTorrent
- s3:RestoreObject

Actions Related to Buckets

- s3>CreateBucket
- s3>DeleteBucket
- s3>ListBucket
- s3>ListBucketVersions
- s3>ListAllMyBuckets (covers REST GET Service and SOAP ListAllMyBuckets)
- s3>ListBucketMultipartUploads

Actions Related to Bucket Sub-Resources

- s3:GetBucketAcl
- s3:PutBucketAcl
- s3:GetBucketCORS
- s3:PutBucketCORS
- s3:GetBucketVersioning

- s3:PutBucketVersioning
- s3:GetBucketRequesterPays
- s3:PutBucketRequesterPays
- s3:GetBucketLocation
- s3:PutBucketPolicy
- s3:GetBucketPolicy
- s3:PutBucketNotification
- s3:GetBucketNotification
- s3:GetBucketLogging
- s3:PutBucketLogging
- s3:GetLifecycleConfiguration
- s3:PutLifecycleConfiguration

You can delete objects by explicitly calling the DELETE Object API or configure its lifecycle (see Object Expiration) to enable Amazon S3 to remove them for you. If you want to block users or accounts from removing or deleting objects from your bucket you must deny them s3:DeleteObject, s3:DeleteObjectVersion and s3:PutLifecycleConfiguration actions.

Systems Operations on AWS

Operations Security I Policy Enforcement

Using Amazon S3 ACLs and Bucket Policies Together

- S3 ACLs have equivalent sets of policy actions
- Mappings:

Bucket ACL	Bucket Policy Actions
READ	s3>ListBucket, s3>ListBucketVersions, s3>ListBucketMultipartUploads
WRITE	s3>PutObject, s3>DeleteObject, s3>DeleteObjectVersion (owner only)
READ_ACP	s3>GetBucketAcl
WRITE_ACP	s3>PutBucketAcl
FULL_CONTROL	(READ + WRITE + READ_ACP + WRITE_ACP)

Object ACL	Object Policy Actions
READ	s3>GetObject, s3>GetObjectVersion, s3>GetObjectTorrent
READ_ACP	s3>GetObjectAcl, s3>GetObjectVersionAcl
WRITE_ACP	s3>PutObjectAcl, s3>PutObjectVersionAcl
FULL_CONTROL	(READ + READ_ACP + WRITE_ACP)

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

ACLs provide a coarse-grain permission model, where you simply grant access permissions to buckets or objects. Bucket policies, on the other hand, provide fine-grain control over the permissions you are granting. For example, you can write a policy granting users access to a bucket or an object, provided the user sends the request from a specific IP address, or the request arrives after a specific date and time. Depending on your needs, you can use one or both of these permission models.

A note about Bucket ACLs:

WRITE—Granting WRITE permission in a bucket ACL allows the `s3:PutObject` and `s3>DeleteObject` actions to be performed on any object in that bucket. In addition, when the grantee is the bucket owner, granting WRITE permission in a bucket ACL allows the `s3>DeleteObjectVersion` action to be performed on any version in that bucket.

The screenshot shows the AWS Policy Generator interface. At the top, there are tabs for "Systems Operations on AWS", "Operations Security Policy Enforcement", and "Policy Generator". The "Policy Generator" tab is active. Below the tabs, there are two main sections: "Effect" (set to "Allow") and "AWS Service" (set to "Amazon EC2"). Under "Actions", it says "1 Action(s) Selected" and shows a list of actions with checkboxes: CreateRoute, CreateRouteTable, **CreateSecurityGroup** (checkbox checked), CreateSnapshot, CreateSpotDatafeedSubscription, CreateSubnet, CreateTags, and CreateVolume. There is also a "All Actions (*)" checkbox. The "Source Name (ARN)" field is empty. The bottom right corner of the interface has the Amazon logo.

The AWS Policy Generator is a tool that enables you to create policies that control access to Amazon Web Services (AWS) products and resources. The different types of policies you can create are an IAM Policy, an S3 Bucket Policy, an SNS Topic Policy and an SQS Queue Policy.

Links to policy samples are also provided from this tool.

The screenshot shows the 'Lifecycle Rule' configuration page for an S3 bucket. At the top, there's a header with tabs: 'Systems Operations on AWS', 'Operations Security I Policy Enforcement', and 'Object Lifecycle Management'. Below the header, there are two main sections: 'Configured via Amazon S3' and 'Available actions'. Under 'Available actions', there are two options: 'Archive to Amazon Glacier' and 'Expire (delete forever)'. The main area is titled 'Lifecycle Rule' and contains fields for 'Enabled' (checked), 'Name (Optional)' (set to 'log-archiving'), 'Apply to Entire Bucket' (unchecked), 'Prefix' (set to 'log/'), and 'Time Period Format' (set to 'Days from the creation date'). A table lists actions and their corresponding time periods: 'Move to Glacier' (365 days from object's creation date) and 'Expiration (Delete Objects)' (3650 days from object's creation date). The 'Expiration' row is highlighted with a blue background.

Lifecycle management defines how Amazon S3 manages objects during their lifetime.

Some objects that you store in an Amazon S3 bucket might have a well-defined lifecycle:

- If you are uploading periodic logs to your bucket, your application might need these logs for a week or a month after creation, and after that you might want to delete them.
- Some documents are frequently accessed for a limited period of time. After that, you might not need real-time access to these objects, but your organization might require you to archive them for a longer period and then optionally delete them later. Digital media archives, financial and healthcare records, raw genomics sequence data, long-term database backups, and data that must be retained for regulatory compliance are some kinds of data that you might upload to Amazon S3 primarily for archival purposes.

For such objects, you can define rules that identify the affected objects, a timeline, and specific actions you want Amazon S3 to perform on the objects.

A rule can specify the following actions:

- Transition—When a specified date or time period in the object's lifetime is

reached, Amazon S3 sets the storage class to Glacier.

For example, you can set a rule for Amazon S3 to transition "MyArchiveObject.jpg" to the Glacier storage class 30 days after creation or to transition all objects with the key prefix "glacierobjects/" to the Glacier storage class a year after creation.

- **Expiration**—When the specified time period is reached in the object's lifetime, Amazon S3 deletes it.

For example, you can set a rule with an expiration action so that Amazon S3 will delete objects with the key prefix "log2012-01-01" 30 days after creation. Expiration applies to all Amazon S3 objects, including those that are archived in Amazon Glacier.

Systems Operations on AWS
Operations Security I Policy Enforcement
Object Lifecycle Management

- Can be applied to entire bucket
- Use prefix to allow more than one lifecycle rule per bucket
- Consider this sample S3 key space:

```
All Buckets
└ OurOrg-Private-Bucket
    └ Logs
        └ App Logs
            └ applog_2013-01-31-00.00.01.tgz
        └ System Logs
            └ syslog_2013-01-31-00.00.01.tgz
```

1. For a prefix of "Logs/", what objects are affected?
2. How about "Logs/System Logs/"?

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Amazon S3 manages object lifetimes with a lifecycle configuration, which is assigned to a bucket and defines rules for individual objects. Each rule in a lifecycle configuration consists of the following:

- A object key prefix that identifies one or more objects to which the rule applies.
- An action or actions that you want Amazon S3 to perform on the specified objects.
- A date or a time period, specified in days since object creation, when you want Amazon S3 to perform the specified action.

You can add these rules to your bucket using either the Amazon S3 console or programmatically.

Answers to Questions:

1. Logs/ = applog_2013-01-31-00.00.01.tgz and syslog_2013-01-31-00.00.01.tgz
2. Logs/System Logs/ = syslog_2013-01-31-00.00.01.tgz

Before You Decide to Archive Objects:

The Transition action in the lifecycle configuration rule enables you to transition objects to the Glacier storage class—that is, archive data to Amazon Glacier. Before you decide to archive objects, note the following:

- Objects in the Glacier storage class are not available in real time.
Archived objects are Amazon S3 objects, but before you can access an

archived object, you must first restore a temporary copy of it. The restored object copy is available only for the duration you specify in the restore request. After that, Amazon S3 deletes the temporary copy, and the object remains archived in Amazon Glacier.

Note that object restoration from an archive can take from three to five hours.

- The transition action allows only one-way transition to the Glacier storage class.

You cannot use a lifecycle configuration rule to convert a Glacier object to a Standard or Reduced Redundancy Storage (RRS) object. If you want to change the storage class of an already archived object to either Standard or RRS, you must use the restore operation to make a temporary copy first. Then use the copy operation to overwrite the object as the Standard or the RRS object.

- Glacier objects are visible and available only through Amazon S3, not through Amazon Glacier

Amazon S3 stores the archived objects in Amazon Glacier; however, these are Amazon S3 objects, and you can access them only by using the Amazon S3 console or the API. You cannot access the archived objects through the Amazon Glacier console or the API.

- A rule with an empty key prefix applies to all the objects in the bucket

If you specify an empty prefix, the rule applies to all objects in the bucket. So if the rule specifies a transition action with an empty prefix, you are requesting Amazon S3 to archive all the objects in the bucket at a specific period in the objects' lifetime.

Before You Decide to Expire Objects:

The Expiration action in the lifecycle configuration rule enables you to request Amazon S3 to delete objects. Before you decide to set this action in the lifecycle configuration, note the following:

- The Expiration action deletes objects

You might have objects in Amazon S3 or archived to Amazon Glacier. No matter where these objects are, Amazon S3 will delete them. You will no longer be able to access these objects.

- A rule with an empty key prefix applies to all the objects in the bucket

If you specify an empty prefix, the rule applies to all objects in the bucket. So if the rule specifies an expiration action with an empty prefix, you are requesting Amazon S3 to delete all the objects in the bucket at a specific period in the objects' lifetime.

Systems Operations on AWS

Operations Security I Policy Enforcement

Knowledge Check



1. What is the default action for access policies?
2. Of Amazon S3 ACLs and access policies, which provides finer grained control?
3. Which access policy statement wins: "explicit deny" or "allow"?

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

1. Deny
2. Access policies
3. Explicit deny

Systems Operations on AWS



Security Event Auditing

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Operations Security | Security Event Auditing

Native AWS Logging & Auditing

- Some AWS services provide native event logging
- Provides operators with security value and context
- Complementary to logs from OS, application, database
- Outputs include CLI, API, Console, S3 objects, others
- Example: RDS events

```
$ aws rds describe-events
[
    {
        "Date": "2013-07-03T20:21:33.200Z",
        "Message": "Applied change to security group",
        "EventCategories": [
            "configuration change"
        ],
        "SourceIdentifier": "stack2-65-1372882770-rds-5ltev6c3fb59-dbsecuritygroup-162iqi4i4eggi",
        "SourceType": "db-security-group"
    },
]
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Security and network operations, incident management, audit, compliance, and forensics teams rely on logs to understand what is happening and what has happened across their infrastructure. The AWS infrastructure provides logging capabilities, which complement events from operating systems, applications and databases to give additional context when researching potential issues.

Several example use cases pertinent to logging and auditing capabilities include:

- Tracking changes within your environment
- Monitoring the activities of privileged users
- Collecting data for behavioral/pattern analytics

In the example shown here, we see an RDS Security Group has been modified via the deployment of a CloudFormation template.

Systems Operations on AWS			
Operations Security Native AWS Logging & Auditing			
Category	Service	Data	Method
Compute	Auto Scaling	Events	CLI, API, SNS
Storage/Content	S3	Object access	Written to S3
Storage/Content	CloudFront	Access logs, cookies	Written to S3
Storage/Content	Glacier	Retrieval jobs only	SNS
Database	Redshift	Event logs/status	CLI, Console, API
Database	RDS	Events	CLI, Console, API, SNS
Database	ElastiCache	Events	SNS
Management	OpsWorks	Chef logs only	Console (download)
Management	CloudFormation	Event logs/status	CLI, Console, API
Management	Elastic Beanstalk	Event logs/status	CLI, Console, API
Management	Data Pipeline	Errors only	Written to S3
Management	CloudHSM	Appliance login, trust links	Syslog
App Services	SES	Bounces, complaints	SNS
App Services	SNS	Messages sent	SNS
App Services	EMR	Infer changes from Hadoop logs	Written to S3

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

This table highlights the AWS logging capabilities for purposes of security, change management and auditing. The method to consume these logs depends on the technologies used to collect, parse, process, and correlate the data. Links have been provided below for logging details from each supported service.

15 services support output of logs with security value. Refer to <http://docs.aws.amazon.com> for details and the latest updates.

Systems Operations on AWS

Operations Security | Third Party Logging & Auditing

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The diagram shows an orange cloud icon with the letters 'AWS' inside. A dashed line connects it to a central hub labeled 'AWS Admin Proxy'. From this hub, three arrows point to three groups of people: 'Security Analysts' (top right), 'Administration' (bottom right), and 'AWS Admins' (bottom left). Above the 'Security Analysts' group, the text 'Audit Queries & Notifications' is written.

- User auditing – administrative proxy
 - Supports deep event logging
 - Not all console functionality
- Environment logging
 - Uses API “describe” calls
 - Useful for auditing changes within your environment
 - Set up alerts for changes that decrease your security or push you out of compliance
- Security management
 - Point products available for Security Group management, application behavior, and more

Third party products can provide additional functionality including authentication control, permission, credentialing, federation, auditing, SSH key management, Security Group management, resource tracking and budgeting – even replays of RDP sessions and SSH command logging for privileged users.

Sample Vendors:

Administrative Proxy: Xceedium Xsuite, Octa

Environment Logging: Netflix Edda, CloudAware

Security Management:

- Dome9 (Provides security group management across regions and accounts, adding support for metadata, DNS name support, auditing & alerting, and more)
- Alert Logic (Intrusion detection, web application security, log management and vulnerability assessment coupled with 24x7 monitoring and expert guidance services)
- Boundary (change monitoring, insight into how applications behave and communicate)

Systems Operations on AWS

Operations Security | Security Event Auditing

Knowledge Check

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

1. How can AWS native event logs supplement OS, app, and DB logs?

2. Name three output methods for AWS native event logs.

1. Native logs provide information about infrastructure changes such as security groups which can be correlated with OS/app/DB logs. **(R14)**
2. Any two of: CLI, API, Console, syslog, SNS, S3



LAB

Security Event Collection

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Operations Security | Security Event Auditing Lab



Lab Overview

1. Use CloudFormation to launch an RDS instance and an EC2 instance
2. Query the API using "describe" call to view RDS instance logs (not database logs)
3. How can this output assist security teams?

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

This lab will automatically launch an RDS instance and an EC2 instance. We'll use the console as well as the CLI to capture events from the RDS service to see what changes have been made recently.

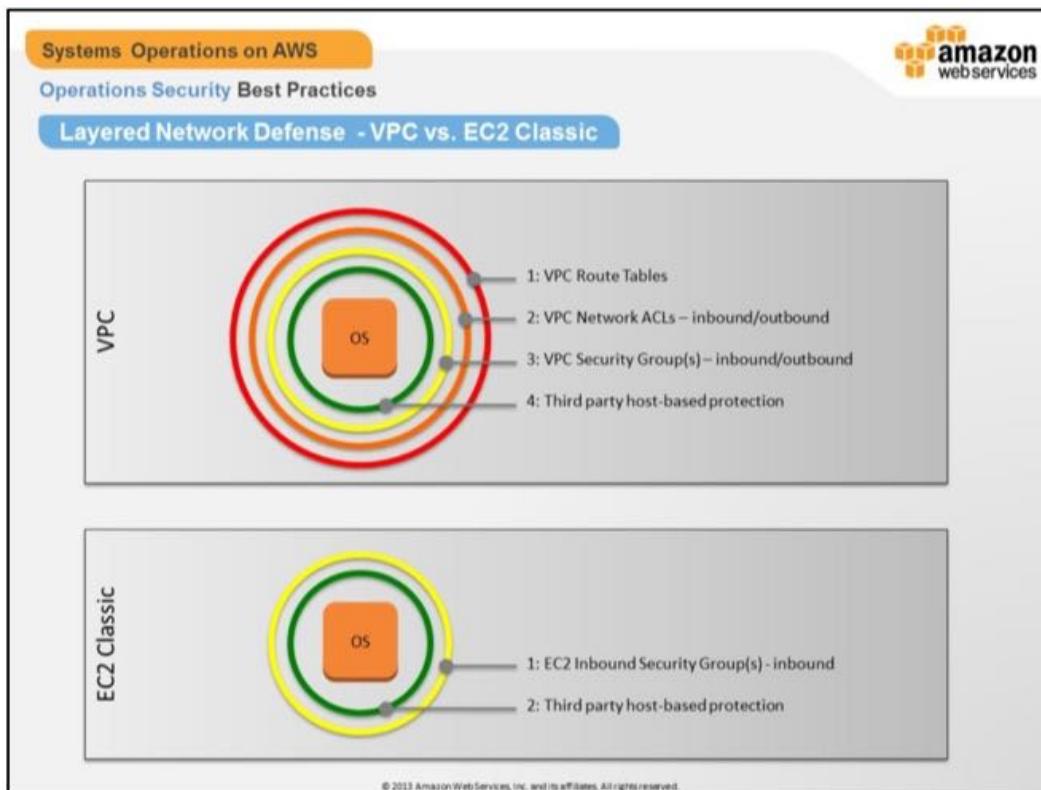
Do you see anything interesting? How might these logs support a security team who is investigating a possible system breach?

Systems Operations on AWS



Best Practices for AWS Security

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



A visual comparison of the network layer defenses available in the VPC and EC2 classic environments.

Systems Operations on AWS
Operations Security | Best Practices
Layered Network Defense - VPC

- VPC Route Tables
 - First security layer
 - Scope: subnet
 - Network segmentation
 - Wide reduction of attack surface area
- VPC Network Access Control List (NACL)
 - Stateless
 - Scope: subnet
 - Default allow
 - Ingress and egress rules
 - Blacklist: Use to block “known bad” ports – NetBIOS, Telnet, etc.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

VPC Route Tables

Route tables can provide you with the ability to reduce exposure to a large number of instances on a subnet. For example, consider a high performance computing (HPC) environment that uses a primary/master system interacting with multiple worker nodes. If the worker nodes do not require communications outside their subnet, traffic can be contained by either restricting the routing for that subnet (e.g. a route to SWF) or by not associating that subnet with a routing table at all. The master node could include an extra Elastic Network Interface (ENI) dedicated for interaction with the worker nodes.

Systems Operations on AWS

Operations Security | Best Practices

Layered Network Defense - VPC

- VPC Security Groups
 - Stateful
 - Scope: instance or elastic network interface (ENI)
 - Default deny ingress / Default allow egress
 - Ingress and egress rules
 - Whitelist: Use to permit "known good"
- Separation of duties
 - NACLs = network admins, "guard rails"
 - Security Groups = server admins, agile operation

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

When used correctly, NACLs and Security Groups can be combined to enforce separation of duties. For example, the network administrators can use NACLs as "guard rails", ensuring that sensitive ports like Telnet are not permitted within the environment. The sever administrators then have flexibility over configuring Security Groups for their instances, within the constraints of the NACLs. In practice, this would mean that server administrators could open Telnet on TPC/23 to their instances but would be effectively blocked by NACLs.

The screenshot shows the AWS Layered Network Defense landing page. At the top left, there are three tabs: "Systems Operations on AWS" (orange), "Operations Security I Best Practices" (blue), and "Layered Network Defense" (blue). At the top right is the Amazon Web Services logo. The main content area lists security options under three categories: "Third party host-based protection" (with sub-points: Firewall, Malware prevention, Encryption, Intrusion detection & prevention, File integrity checking, Data loss prevention, Traffic analysis, Reputation blocking), "AWS Partner Network (APN) Technology Partners" (with sub-point: Solutions that run on, or are complementary to AWS), and "AWS Marketplace" (with sub-point: Launch security solutions into AWS from the AWS Marketplace). At the bottom right is a yellow button labeled "Accept Terms & Launch with 1-Click". A small note at the bottom center states: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

A myriad security options exist through the AWS Partner Network (APN) and the AWS Marketplace.

APN Technology Partners

APN Technology Partners are commercial software and internet services companies that build solutions that run on, or are complementary to, AWS. Technology Partners include Independent Software Vendors (ISVs), SaaS, PaaS, developer tools, management and security vendors.

AWS Marketplace

AWS Marketplace is an online store that helps customers find, buy, and immediately start using the software and services they need to build products and run their businesses. AWS Marketplace complements programs like the Amazon Partner Network and is another example of AWS's commitment to growing a strong ecosystem of software and solution partners.

Visitors to the marketplace can use AWS Marketplace's 1-Click deployment to quickly launch pre-configured software and pay only for what they use, by the hour or month. AWS handles billing and payments, and software charges appear on customers' AWS bill.

AWS Marketplace features many categories including databases, application servers, testing tools, monitoring tools, content management, and business

intelligence software.

Systems Operations on AWS

Operations Security I Best Practices

Layered Network Defense – EC2 Classic

- What network defense is available in EC2?
- Security Groups
 - Can only control ingress traffic
- What's not available in EC2 Classic
 - Routing Tables
 - NACLs
- Third party host-based protection still recommended

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The network security features in EC2 Classic (non-VPC) environment lack the Routing Tables and NACLs features found in the VPC environment. Still, the EC2 Classic environment is useful for the enterprise for rapid application development and proof of concept testing.

Systems Operations on AWS

Operations Security I Best Practices

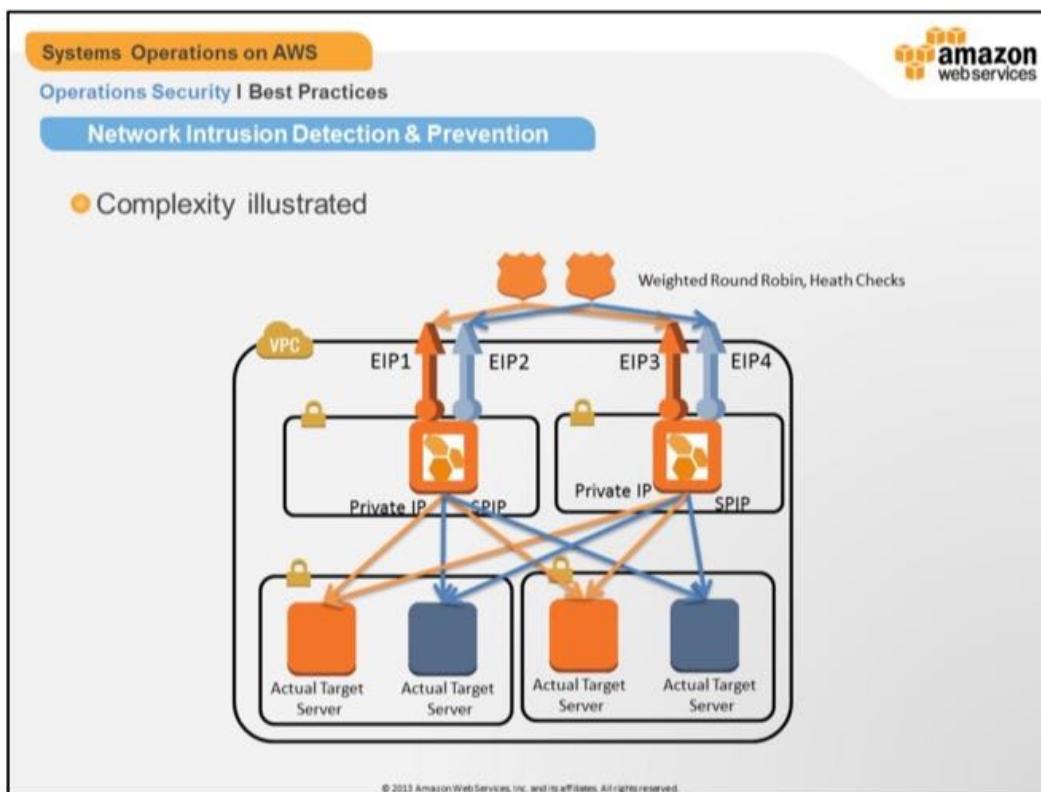
Network Intrusion Detection & Prevention

- Traditional method
 - Network appliance chokepoints
 - Cloud ready?
- Challenges for network chokepoints
 - High availability (VRRP, HSRP) - No multicast, broadcast, AnyCast, or gratuitous ARP
 - No concept of port mirror or port spanning
 - Sessions & state tables
 - Horizontal scaling (Auto Scaling)
 - Licensing
- Cloud: host-based recommended
 - Permits elasticity, Auto Scaling, load fluctuations
 - Similar Functionality, Different Layer

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

White Board Opportunity at “Cloud ready?” bullet:

- Draw a Multi-AZ, tiered web application with Auto Scaling
- Talk to audience about how one would attempt to insert network IDS or firewalls
- Attempt to draw the network-level controls into the existing diagram
- Explain considerations for HA, session/state tables
- Finally, discuss what happens when the Auto Scaling instances scale out
- Discuss impact at 10/100/1,000/10,000 instances



Does this diagram seem complex? How difficult would this be to build? How difficult would this be to maintain? Which COTS vendors are up to the task?

From our previous discussions, which major component is missing from the diagram that would add even more complexity and further illustrate the primary disadvantage to this architecture? (Answer: Auto Scaling)

This diagram should help drive the point home that in an elastic, virtualized environment, host-based security controls combined with AWS-provided layered network security controls is the preferred approach.

Systems Operations on AWS
Operations Security | Best Practices
Other Protection Components

- **Bastion host**
 - Limit administrative access, tight Security Groups
 - Further reduction in attack surface
 - Shutdown when not in use
- **Elastic Load Balancing (ELB)**
 - Remove direct traffic to instances from Internet
- **Amazon CloudFront**
 - Distribute content to thwart DoS attacks
- **Auto Scaling**
 - Absorb DoS attacks
 - Set limits to prevent Denial of Funding attacks
- **Web Application Firewall (WAF) SaaS providers**
 - Inspect ingress and egress traffic

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

AWS security monitoring tools help identify several types of denial of service (DoS) attacks, including distributed, flooding, and software/logic attacks. When DoS attacks are identified, the AWS incident response process is initiated. In addition to the DoS prevention tools, redundant telecommunication providers at each region as well as additional capacity protect against the possibility of DoS attacks. The AWS network provides significant protection against traditional network security issues, and you can implement further protection.

Auto Scaling & DDoS:

Your organization may determine that sometimes, it's less expensive to let a DDoS attack disable your workload. Simple example:

Determine the value of your Internet-facing workload in terms of currency per hour. Determine costs associated with Auto Scaling your infrastructure to absorb a DDoS attack. Set the max limit of your Auto Scaling group to ensure that your instance costs don't exceed your workload value.

Systems Operations on AWS



Trusted Advisor

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



The summary view allows you to quickly see the overall status of your AWS application across the four categories. Resources that have been suppressed will be ignored and provide a more accurate view of your application's health. You can drill into each individual check for details and a list of affected resources.

You have the ability to suppress any desired results, after you have inspected the results of a check and decide not to make any changes to the AWS resource or setting in question.

Systems Operations on AWS
Operations Security I Best Practices
Trusted Advisor

- Available with Business and Enterprise Support
- Multiple best practices checks
- Automated checks:
 - Cost optimization
 - Security
 - Fault tolerance
 - Performance
- Download report into XLS spreadsheet from site
- <https://aws.amazon.com/premiumsupport/trustedadvisor>

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

AWS Trusted Advisor draws upon best practices learned from AWS' aggregated operational history of serving hundreds of thousands of AWS customers. The AWS Trusted Advisor inspects your AWS environment and makes recommendations when opportunities exist to save money, improve system performance, or close security gaps. We are continually expanding the number of checks available through Trusted Advisor with over 100 additional best practice checks in our development pipeline. Trusted Advisor is available to customers with Business and Enterprise-level support.

The screenshot shows the AWS Trusted Advisor interface. At the top, there are three tabs: "Systems Operations on AWS" (highlighted in orange), "Operations Security | Best Practices" (highlighted in blue), and "Trusted Advisor". Below these tabs, the "Trusted Advisor" section is active, indicated by a blue bar with the text "Trusted Advisor". Under this section, there is a heading "Automated Security Checks" preceded by a yellow circle icon. A bulleted list follows, detailing various security checks performed by Trusted Advisor:

- Security Groups open ports granting global access
- Security Groups CIDR config granting global access
- IAM usage confirmation
- S3 bucket permissions
- AWS root account is MFA-enabled
- IAM password policy in use
- RDS security group issues
- And more...

At the bottom of the screenshot, a small copyright notice reads: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

Improve the security of your application by closing gaps, enabling various AWS security features, and examining your permissions.

Note that Trusted Advisor is also accessible via the AWS Support API. You can get the list of checks, access the latest results, and re-run the checks to refresh the results. This has been a much requested feature for customers who wish to integrate support case management into their in-house ticketing systems, and with the release of Support API's we have delivered on this request.

You can call `DescribeTrustedAdvisorChecks` to learn about the checks performed by the Trusted Advisor, `DescribeTrustedAdvisorCheckResult` to learn more about the results of a check, or `RefreshTrustedAdvisorCheck` to request a refresh of a particular check.

Imagine using this set of APIs to build a tuning and optimizing system for AWS! The system can get the results of a check, use the AWS APIs to tune and adjust, and then refresh the check to ensure that it has achieved the desired results.

We have created some code samples to show you how to use the APIs, published in our Github repo at <https://github.com/awslabs/aws-support-examples>.

Systems Operations on AWS

Operations Security | Best Practices

Knowledge Check

1. Why are traditional network-based security implementations not recommended for AWS?
2. What VPC network security layers are missing from the EC2 Classic environment?
3. True/False: Trusted Advisor checks to see if IAM and MFA are configured for each AWS account.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

1. Creates performance bottlenecks for Auto Scaling/elasticity. Challenges with HA and scaling because of No multicast, broadcast, gratuitous ARP. No port spanning for traffic sniffing.
2. Routing tables and NACLs
3. True (assuming you have appropriate support level)

Systems Operations on AWS



Encryption Deep Dive

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Operations Security | Encryption

Storage

- All AWS service endpoints support TLS/SSL
 - Some also allow unencrypted connections by default (S3)
- Enforcing Amazon S3 encryption by policy example:

```
{  
    "Statement": [  
        {  
            "Version": "2012-10-17",  
            "Principal": "*",  
            "Effect": "Allow",  
            "Action": "s3:GetObject",  
            "Resource": "arn:aws:s3:::mybucket/*",  
            "Condition":{  
                "Bool":{  
                    "aws:SecureTransport":"true"  
                }  
            }  
        }  
    ]  
}
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Even if you upload sensitive data using TLS/SSL and use server-side encryption, your intended recipient may download the object in clear text. How do you make sure only TLS/SSL is used to encrypt the data in transit? Use a S3 bucket policy.

Systems Operations on AWS
Operations Security | Storage Encryption

Amazon Web Services

- Amazon S3 and Amazon Glacier Server-Side Encryption (SSE)
- Durable – Amazon S3 key storage
- AWS manages the encryption keys
- Strong AES-256 encryption
- Simple Implementation – Additional S3 PUT header
- Automatic for Glacier

```
graph LR; Data[Data] -- "Per-object key" --> EncryptedObject[Encrypted object]; EncryptedObject --> Bucket1[bucket]; EncryptedObject -- "Master key" --> EncryptedKey[Encrypted per-object key]; EncryptedKey --> Bucket2[bucket]; Bucket3[bucket] --- KM[Key management (monthly rotation)];
```

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Encryption provides added security for your object data stored in your buckets in Amazon S3. You can encrypt data on your client-side and upload the encrypted data to Amazon S3. In this case, you manage encryption process, the encryption keys, and related tools. Optionally, you might want to use the server-side encryption feature in which Amazon S3 encrypts your object data before saving it on disks in its data centers and decrypts it when you download the objects, freeing you from the tasks of managing encryption, encryption keys, and related tools.

Amazon S3 Server Side Encryption employs strong multi-factor encryption. Amazon S3 encrypts each object with a unique key. As an additional safeguard, it encrypts the key itself with a master key that it regularly rotates. Amazon S3 Server Side Encryption uses one of the strongest block ciphers available, 256-bit Advanced Encryption Standard (AES-256), to encrypt your data.

You can specify data encryption at the object level. When you upload an object, you can explicitly specify in your request if you want Amazon S3 to save your object data encrypted. Server-side encryption is optional. Your bucket might contain both encrypted and unencrypted objects. Amazon S3 supports bucket policy that you can use if you require server-side encryption for all objects that are stored in your bucket. For example, the following bucket policy denies upload object (`s3:PutObject`) permission to everyone if the request does not include the `x-amz-server-side-encryption` header requesting server-side encryption.

Server-side encryption encrypts only the object data. Any object metadata is not encrypted. The object creation REST APIs provide a request header, x-amz-server-side-encryption that you can use to request server-side encryption.

Bucket Policy Sample

```
{  
"Version":"2008-10-17",  
"Id":"PutObjPolicy",  
"Statement":[{  
"Sid":"DenyUnEncryptedObjectUploads",  
"Effect":"Deny",  
"Principal":{ "AWS":"*" },  
"Action":"s3:PutObject",  
"Resource":"arn:aws:s3:::YourBucket/*",  
"Condition":{  
"StringNotEquals":{ "s3:x-amz-server-side-encryption":"AES256"  
} } ]}
```

Systems Operations on AWS
Operations Security | Encryption
Database

- **Amazon RDS Oracle**
 - Transparent Data Encryption for Oracle
 - Encrypt table spaces or specific table columns
 - AES or 3DES algorithms
 - Amazon RDS Oracle Native Network Encryption (NNE), configured at client and/or server
- **Amazon Redshift**
 - Cluster encryption for all user-created tables
 - Load data encryption using Amazon S3 client-side encryption
 - Supports server-side or client-side data encryption
- **Amazon RDS MySQL and SQL Server**
 - SSL encryption, client configured

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Oracle TDE

Amazon RDS supports Oracle Transparent Data Encryption (TDE), a feature of the Oracle Advanced Security option available in Oracle Enterprise Edition. This feature automatically encrypts data before it is written to storage and automatically decrypts data when the data is read from storage.

Note: The TDE option is a permanent option that cannot be removed from an option group, and that option group cannot be removed from a DB instance once it is associated with a DB instance. You cannot disable TDE from a DB instance once that instance is associated with an option group with the Oracle TDE option.

Oracle Transparent Data Encryption is used in scenarios where you need to encrypt sensitive data in case data files and backups are obtained by a third party or when you need to address security-related regulatory compliance issues.

Oracle Transparent Data Encryption supports two encryption modes: TDE tablespace encryption and TDE column encryption. TDE tablespace encryption is used to encrypt entire application tables. TDE column encryption is used to encrypt individual data elements that contain sensitive data. You can also apply a hybrid encryption solution that uses both TDE tablespace and column encryption.

Amazon RDS supports Oracle native network encryption, a feature of the Oracle Advanced Security option available in Oracle Enterprise Edition. With native network encryption, you can encrypt data as it moves to and from a DB instance.

Amazon Redshift

Cluster encryption:

When creating a cluster you can optionally choose the cluster encryption option for additional security. When you enable encryption in your cluster, Amazon Redshift stores all data in user-created tables in an encrypted format. Note that enabling encryption in your cluster will impact performance, even though it is hardware accelerated. On average, we expect you will see approximately a 20% degradation, with peak overheads of 40%. You should take this into account when deciding whether you should enable encryption when you create the cluster. Encryption is an immutable property of the cluster. The only way to go from encrypted to non encrypted or vice versa is to unload the data and reload it to a new cluster. Encryption also applies to backups. When restoring from an encrypted snapshot, the new cluster will be encrypted as well.

Load data encryption:

With Amazon Redshift, you can copy data from an Amazon S3 bucket to an Amazon Redshift database. The data that you copy can be encrypted on either the serverside or clientside. In server-side encryption, Amazon S3 handles encryption and decryption, transparently. In client-side encryption, you manage the encryption keys and the related encryption and decryption process.

You can upload data to an Amazon S3 bucket using client-side encryption, then securely load the data using the COPY command with the ENCRYPTED option and a private encryption key. You encrypt your data using envelope encryption. With envelope encryption, your application handles all encryption exclusively. Your private encryption keys and your unencrypted data are never sent to AWS, so it's very important that you safely manage your encryption keys. If you lose your encryption keys, you won't be able to unencrypt your data, and you can't recover your encryption keys from AWS. Envelope encryption combines the performance of fast symmetric encryption while maintaining the secure key management that asymmetric keys provide. A one-time-use symmetric key (the envelope symmetric key) is generated by your Amazon S3 encryption client to encrypt your data, then that key is encrypted by your master key and stored alongside your data in Amazon S3. When Amazon Redshift accesses your data during a load, the encrypted symmetric key is retrieved and decrypted with your real key, then the data is decrypted.

Oracle NNE

To use Oracle native network encryption with a DB instance, you add the NATIVE_NETWORK_ENCRYPTION option to an option group and associate that option group with the DB instance. You should first determine if the DB instance is associated with an option group that has the NATIVE_NETWORK_ENCRYPTION option. To view the option group that a DB instance is associated, you can use the

RDS console, the rds-describe-db-instance CLI command, or the API action `DescribeDBInstances`. Amazon RDS supports Oracle native network encryption for any DB instance class larger than db.t1.micro.

You can change the order or limit the algorithms that the DB instance will accept.

- RC4_256: RSA RC4 (256-bit key size)
- AES256: AES (256-bit key size)
- AES192: AES (192-bit key size)
- 3DES168: 3-key Triple-DES (168-bit effective key size)
- RC4_128: RSA RC4 (128-bit key size)
- AES128: AES (128-bit key size)
- 3DES112: 2-key Triple-DES (112-bit effective key size)
- RC4_56: RSA RC4 (56-bit key size)
- DES: Standard DES (56-bit key size)
- RC4_40: RSA RC4 (40-bit key size)
- DES40: DES40 (40-bit key size)

You can also specify the checksum algorithm. The default is sha-1, but md5 is also supported.

MySQL

Amazon RDS supports SSL connections with DB instances running the MySQL database engine.

Amazon RDS creates an SSL certificate and installs the certificate on the DB instance when Amazon RDS provisions the instance. These certificates are signed by a certificate authority. The public key is stored at <https://rds.amazonaws.com/doc/rds-ssl-ca-cert.pem>.

Important: The SSL support in Amazon RDS is strictly for encrypting the connection between your client and your DB instance; it should not be relied on for authenticating the server.

Microsoft SQL

Amazon RDS supports SSL encryption for SQL Server DB Instances. Using SSL, you can encrypt a SQL Server connection between your applications and your SQL Server DB Instances. SSL support is available in all AWS regions for all SQL Server editions including Express, Web, Standard and Enterprise.

Systems Operations on AWS
Operations Security | Encryption
Compute

Amazon

Amazon EC2 Instance Storage and EBS

- Windows
 - Windows Encrypting File System (EFS) for individual files (not the entire file system)
 - Open source or commercial volume encryption
- Linux
 - Open source or commercial file and volume encryption

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

If you are concerned about storing sensitive and confidential data in the cloud, you should encrypt the data (individual files) before uploading it to the cloud. For example, encrypt the data using any open source or commercial PGP-based tools before storing it as Amazon S3 objects and decrypt it after download. This is often a good practice when building HIPAA-Compliant applications that need to store Protected Health Information (PHI).

On Amazon EC2, file encryption depends on the operating system. Amazon EC2 instances running Windows can use the built-in Windows Encrypting File System (EFS) feature. This feature will handle the encryption and decryption of files and folders automatically and make the process transparent to the users [6]. However, despite its name, EFS doesn't encrypt the entire file system; instead, it encrypts individual files. If you need a full encrypted volume, consider using the opensource TrueCrypt product; this will integrate very well with NTFS-formatted EBS volumes.

Amazon EC2 instances running Linux can mount EBS volumes using encrypted file systems using variety of approaches (EncFS, Loop-AES, dmcrypt, TrueCrypt). Likewise, Amazon EC2 instances running OpenSolaris can take advantage of ZFS10 Encryption Support. Regardless of which approach you choose, encrypting files and volumes in Amazon EC2 helps protect files and log data so that only the users and processes on the server can see the data in clear text, but anything or anyone outside the server see only encrypted data.

Vendors: TrendMicro, SafeNet, PGP

Open Source Windows: TrueCrypt

Open Source Linux: EncFS, Loop-AES, dmcrypt, TrueCrypt

Systems Operations on AWS
Operations Security | Encryption

Key Ownership

Encryption	Services	Key Manager
Server-side Encryption	S3, Glacier	AWS
Client-side Encryption	S3 (Ruby SDK, Java SDK)	Customer

● Secure Key Storage: AWS CloudHSM

- AWS-managed, dedicated hardware appliance
- Managed SafeNet Luna SA
- Resides in AWS data centers, logically in VPC
- You manage the keys; AWS has no key access
- FIPS 140-2 and Common Criteria EAL4+ validated



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Instead of using Amazon S3's server-side encryption, you also have the option of encrypting your data before sending it to Amazon S3. You can build your own library that encrypts your objects data on the client side before uploading it to Amazon S3. Optionally, you can use the AWS SDK for Java, which you can use to automatically encrypt your data before uploading it to Amazon S3.

Important: When using client-side encryption, your private encryption keys and your unencrypted data are never sent to AWS; therefore, it is important that you safely manage your encryption keys. If you lose your encryption keys, you won't be able to unencrypt your data.

The AWS SDK for Java uses a process called envelope encryption. In envelope encryption, you provide your encryption key to the Amazon S3 encryption client and the client takes care of the rest of the process. Your private encryption key that the client uses in the envelope encryption process can be an asymmetric key pair (composed of public and private keys) or it can be a symmetric key. We recommend using asymmetric keys for the increased level of security that they provide in the envelope encryption process.

<http://docs.aws.amazon.com/AWSRubySDK/latest/AWS/S3/S3Object.html>

CloudHSM

A hardware security module (HSM) is a hardware appliance that provides secure

key storage and cryptographic operations within a tamper-resistant hardware module. HSMs are designed to securely store cryptographic key material and use the key material without exposing it outside the cryptographic boundary of the appliance.

AWS CloudHSM helps you meet corporate, contractual and regulatory compliance requirements for data security by using dedicated hardware security module (HSM) appliances within the AWS cloud. AWS and AWS Marketplace partners offer a variety of solutions for protecting sensitive data within the AWS platform, but additional protection is necessary for some applications and data that are subject to strict contractual or regulatory requirements for managing cryptographic keys.

Until now, your only options were to maintain the sensitive data or the encryption keys protecting the sensitive data in your on-premise data centers. However, those options either prevented you from migrating these applications to the cloud or significantly slowed application performance. CloudHSM allows you to protect your encryption keys within HSMs that are designed and validated to government standards for secure key management. You can securely generate, store, and manage the cryptographic keys used for data encryption in a way that ensures that only you have access to the keys. CloudHSM helps you comply with strict key management requirements within the AWS cloud without sacrificing application performance.

AWS CloudHSM works with Amazon Virtual Private Cloud (Amazon VPC). HSM appliances are provisioned inside your VPC with an IP address that you specify, providing simple and private network connectivity to your Amazon EC2 instances. Placing HSM appliances near your EC2 instances decreases network latency, which can improve application performance. Your HSM appliances are dedicated exclusively to you and are isolated from other AWS customers. Available in multiple regions and Availability Zones, CloudHSMs can be used to build highly available and durable applications.

Important: AWS strongly recommends that you use two or more HSM appliances in a high availability (HA) configuration. The failure of a single HSM appliance in a non-HA configuration can result in the permanent loss of keys and data.

Systems Operations on AWS

Operations Security I Encryption

Knowledge Check



1. Which AWS service endpoint does not support TLS/SSL?

2. True/False: S3 uses a unique encryption key for each **bucket** when using server-side encryption.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

1. None. All AWS service endpoints support TLS/SSL.
2. False. Each **object** is encrypted with a unique key.

Systems Operations on AWS



Vulnerability Scanning & Penetration Testing

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Operations Security | Vulnerability Scanning & Penetration Testing

Coordinate Testing

- Scans must be coordinated with AWS
- Scans must comply with the AWS Acceptable Use Policy:
<https://aws.amazon.com/aup>
- Only scanning of customer EC2 instances is permitted
- Review procedures and complete request form:
<https://aws.amazon.com/security/penetration-testing>
- Testing of m1.small and t1.micro instance types is not currently permitted
- Denial of service attacks and flooding not permitted

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Testing of m1.small and t1.micro instance types is not currently permitted to avoid potential adverse performance impact on shared customer resources

AWS's Policy Regarding the Use of Security Assessment Tools and Services

AWS's policy regarding the use of security assessment tools and services allows significant flexibility for performing security assessments of your AWS assets while protecting other AWS customers and ensuring quality-of-service across AWS. AWS understands there are a variety of public, private, commercial, and/or open-source tools and services to choose from for the purposes of performing a security assessment of your AWS assets.

The term "security assessment" refers to all activity engaged in for the purposes of determining the efficacy or existence of security controls amongst your AWS assets, eg. port-scanning, vulnerability scanning/checks, penetration testing, exploitation, web application scanning, as well as any injection, forgery, or fuzzing activity, either performed remotely against your AWS assets, amongst/between your AWS assets, or locally within the virtualized assets themselves.

You are NOT limited in your selection of tools or services to perform a security assessment of your AWS assets. However, you ARE prohibited from utilizing any tools or services in a manner that perform Denial-of-Service (DoS) attacks, or simulations of such, against ANY AWS asset, yours or otherwise. Prohibited activities include, but may not be limited to:

- Protocol flooding (eg. SYN flooding, IMCP flooding, UDP flooding)
- Resource request flooding (eg. HTTP request flooding, Login request flooding, API request flooding)

A security tool that solely performs a remote query of your AWS asset to determine a software name and version, such as "banner grabbing," for the purpose of comparison to a list of versions known to be vulnerable to DoS, is NOT in violation of this policy.

Additionally, a security tool or service that solely crashes a running process on your AWS asset, temporary or otherwise, as necessary for remote or local exploitation as part of the security assessment, is NOT in violation of this policy. However, this tool may NOT engage in protocol flooding or resource request flooding, as mentioned above.

A security tool or service that creates, determines the existence of, or demonstrates a DoS condition in ANY other manner, actual or simulated, is expressly forbidden. Some tools or services include actual DoS capabilities as described, either silently/inherently if used inappropriately or as an explicit test/check or feature of the tool or service. Any security tool or service that has such a DoS capability, must have the explicit ability to DISABLE, DISARM, or otherwise render HARMLESS, that DoS capability. Otherwise, that tool or service may NOT be employed for ANY facet of the security assessment.

It is the sole responsibility of the AWS customer to ensure the tools and services employed for performing a security assessment are properly configured and successfully operate in a manner that does not perform DoS attacks or simulations of such. It is the sole responsibility of the AWS customer to independently validate the tool or service employed does not perform DoS attacks, or simulations of such, PRIOR to security assessment of any AWS assets. This AWS customer responsibility includes ensuring contracted third-parties perform security assessments in a manner that does not violate this policy.

Furthermore, you are responsible for any damages to AWS or other AWS customers that are caused by your penetration testing activities.

Systems Operations on AWS

Operations Security | Vulnerability Scanning & Penetration Testing

Pre-Authorized Scanning

- Allows on-demand and scheduled scanning
- Available for scanning in EC2 Classic and VPC
- Provided by approved AWS Marketplace partners
- Options, capabilities and procedures vary by vendor

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Pre-Approved scanning is available from some vendors in the AWS Marketplace including Qualys, Core Security and nCircle.

Note: The pre-approved scans are intended for vulnerability scanning (version inspection) and less appropriate for penetration testing (attempting exploits). The parameters included in these tests are often restricted by the vendor to comply with AWS policy. In other words, if you launch a QualysGuard AMI with pre-approved scanning, that instance is only able to perform specific “workloads” or tests.

Systems Operations on AWS



AWS Security Resources

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Operations Security | AWS Security Resources



- AWS Site Links
 - <http://aws.amazon.com/security>
 - <http://aws.amazon.com/compliance>
- AWS Support & Trusted Advisor
 - <https://aws.amazon.com/premiumsupport/trustedadvisor>
- AWS Partner and Marketplace Solutions for Security
 - <http://aws.amazon.com/partners/overview/consulting-partner>

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Module 10: Logging

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Logging | Introduction

Amazon

- Amazon EC2 instances:
 - Elastic
 - Disposable
- What about log data?
 - Analysis
 - Compliance
- "Where should we store log files in AWS?"

Short answer: Amazon S3

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

In well-architected AWS infrastructures, Amazon EC2 instances are meant to be temporary, disposable resources, particularly in the case of application servers.

In contrast, the system log or application data should usually be kept, even after the instance is gone. We might want to keep the data around for analysis, and we may also need to keep it for compliance reasons.

So, how can we keep our log data after the instance is gone?

The short answer is "put it in S3", and that's what we're going to learn how to do.

Systems Operations on AWS

Logging | Overview

What We Will Cover



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- Log collection architectures
- Amazon S3 key optimization
- Log retention and Amazon S3 lifecycle policies
- Lab activity

In this module, we're going to cover the collection and retention of system and application logs.

- We'll review some log collection architectures and discuss the pros and cons of each.
- We'll explore how S3 key partitioning works and how to best optimize key naming conventions for logs, and finally
- we'll discuss log retention and S3 lifecycle policies

There will also be a lab in which you'll go through the process of setting up a repository for logs in S3 and configuring a log server to use that repository.

Systems Operations on AWS

Logging | Overview

What We Will Cover

By the end of this section you should be able to:

- Determine best logging architecture and retention policy for any given application
- Set up Amazon S3 destinations for log repositories
- Create scripts to upload logs to Amazon S3
- Determine naming convention for S3 keys to facilitate search and discovery of logs

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

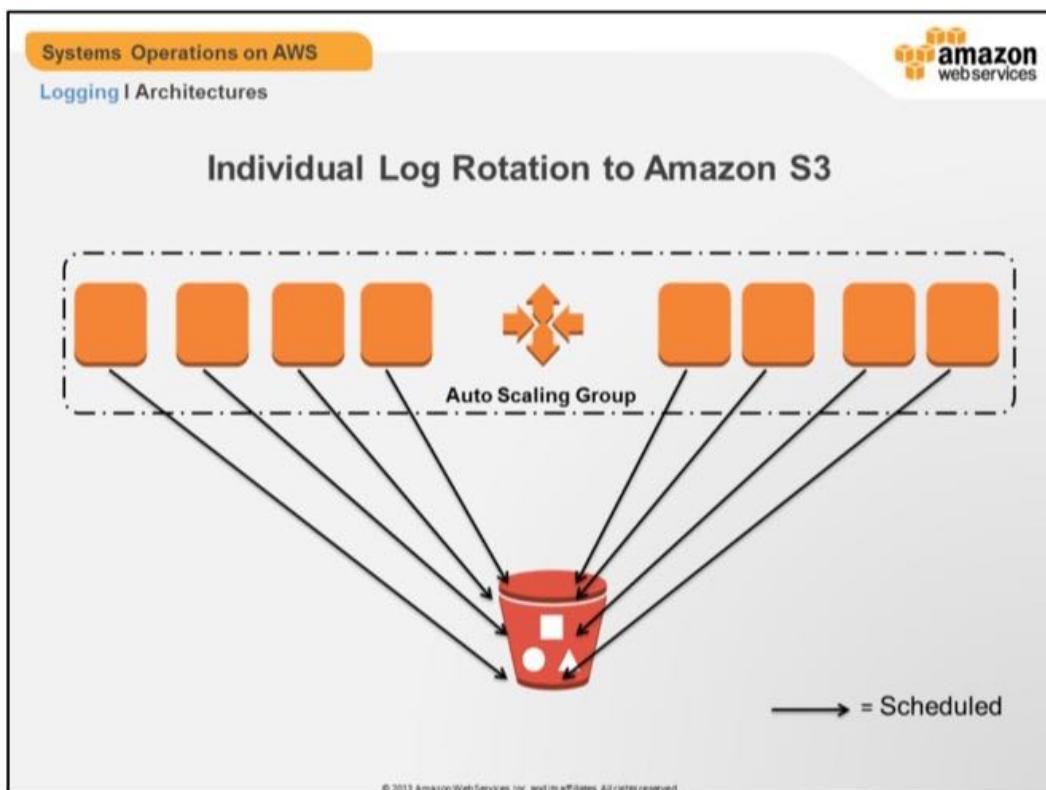
By the end of this section you should be able to:

- Determine best logging architecture and retention for any given application
- Set up S3 destinations for log repositories
- Create scripts to upload logs to S3
- Determine naming convention for S3 keys to facilitate search and discovery of logs.



Architectures

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.



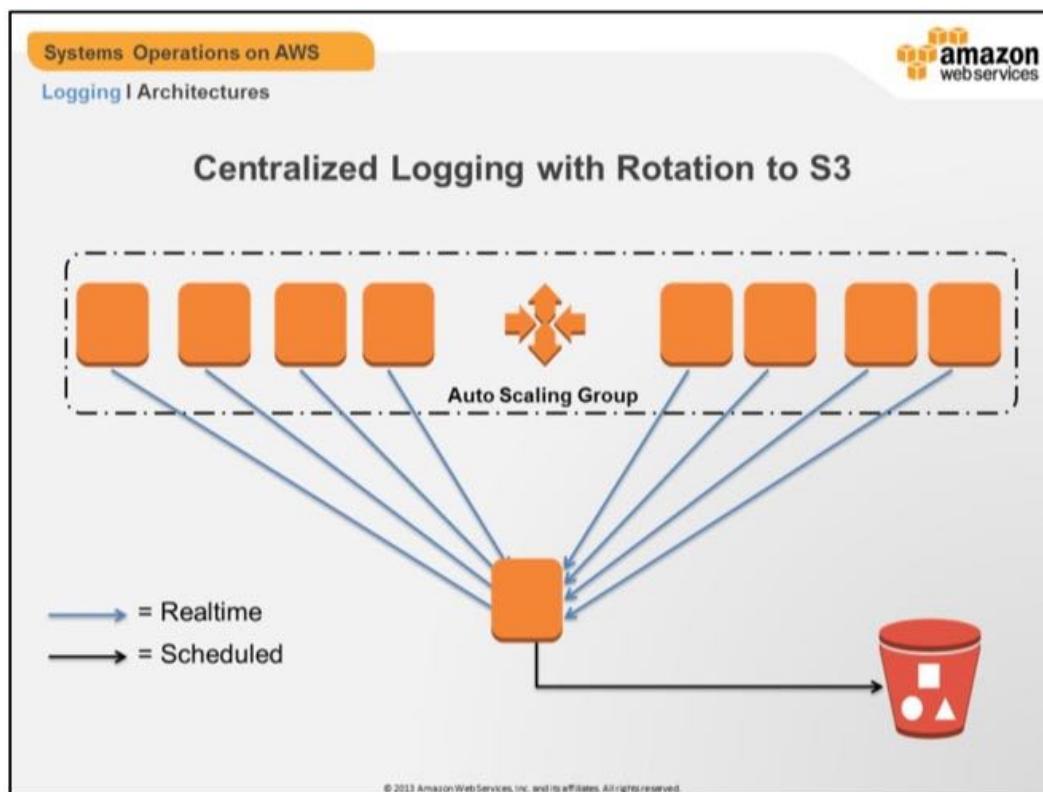
Here's a really simple setup: every single instance periodically rotates its logs to S3.

What's good about this?

- It's simple – the only thing to set up is adding S3 upload to the log rotation process
- It lends itself better to distributed loads from EMR or Redshift for analysis

What's not so good?

- Difficult to see what's going on in aggregate in real time – for example trying to track down a potential DOS
- If an instance unexpectedly terminates, we will lose any log data that wasn't rotated to S3. (How bad is that? It depends on the data and your organization – for most applications it can probably be deemed an acceptable risk)
- In the event of an Auto Scaling removal, the instance can ship its logs to S3 before terminating, so that potential loss only refers to an unplanned instance loss.



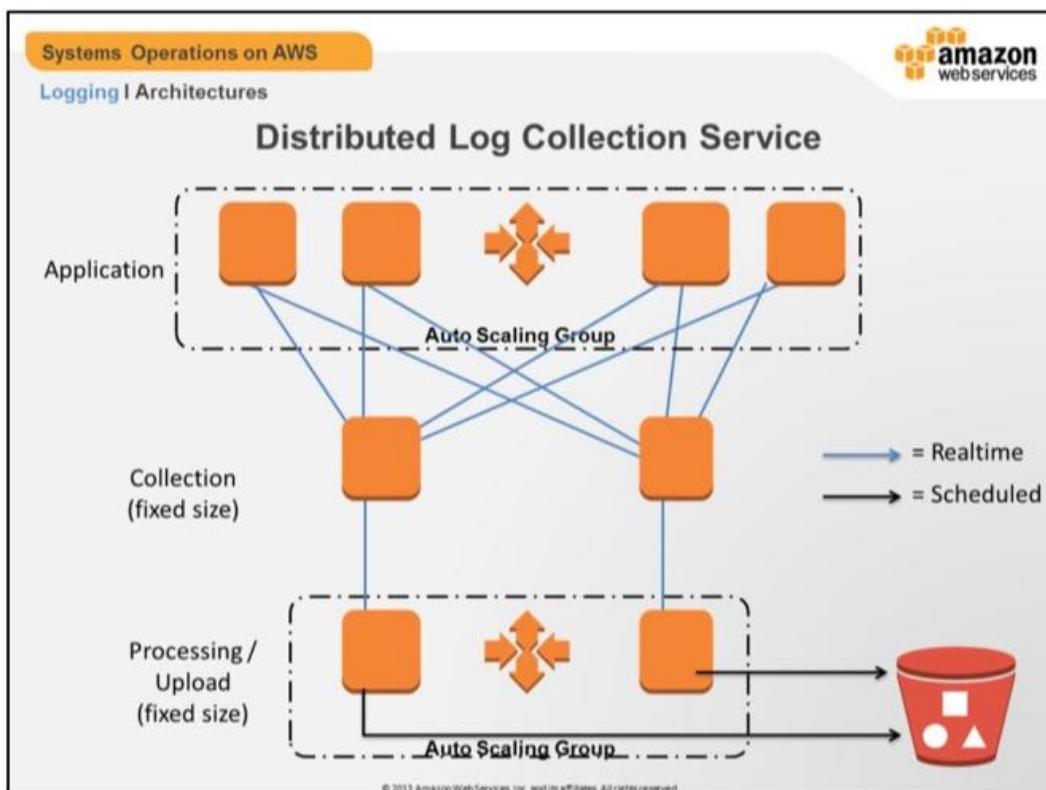
Here's a similar setup

What's good about this?

- We can manually search for log patterns in real time if necessary
- We will reduce the number of logfiles in S3. This is potentially worse for data loads but better if we want to manually search through logs for a specific event or pattern.

What's not so good?

- This specific model doesn't scale – if we start getting into very large numbers of instances then we'll be constrained by the network and disk of the centralized server. This could be ameliorated by spreading the load across multiple log servers.
- More eggs in one basket – if the log server unexpectedly dies then we'll lose all log data since the last rotation. When each instance is uploading its own logs, that may not be a big problem, but with multiple servers it's a bigger impact.



Getting a little more complex – sending logs to a replicated message transport such as Apache Flume. An HA collector and HA Processing engine ensure that no messages are lost.

What's good about this?

- No messages lost
- Still have centralized logs.

What's not so good?

- Increased complexity, difficulty of implementation, and cost



Finally, you can let someone else do the work for you.

What's good about this?

- Really easy! Generally just install an agent and you're done
- May be able to provide metrics or analytics that would be complex to self-implement
- No missed log messages

What's not so good?

- Cost
- May still need to self-implement some non-standard analysis, which takes away from part of the value-add
- The biggest drawback is you're trusting the security of your data to somebody else – for some organizations, this may be a deal-breaker.

The screenshot shows a navigation bar at the top with tabs: 'Systems Operations on AWS' (highlighted in orange), 'Logging | Logging Tools' (blue), and 'Analysis' (blue). The main content area contains a bulleted list of tools for log analysis:

- Commercial software / managed solutions
- Hadoop
 - Distributed, MapReduce, ideal for log analysis
 - EMR = managed Hadoop, optimized for Amazon S3 input / output.
- Amazon Redshift
 - Parallelized, S3 optimized, JDBC / SQL
- DIY (if you must)

At the bottom of the content area, there is a small copyright notice: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

There are multiple tools available for performing log analysis. First off, there are some great commercial and managed solutions for log management, and although we're not going to get into using any of them, they're a fine option for many organizations.

For those who want to perform their own analyses, one of the most popular (probably **the** most popular) frameworks is Hadoop. Hadoop works by breaking up jobs, distributing the pieces to worker nodes, then processing the results from all the nodes to produce a single output – well, that's a pretty simplified explanation, but that's the idea. Hadoop works really well for log analysis, because generally in logs you're looking for patterns (e.g. "add to cart") and then running some sort of analysis on those patterns (e.g. how many "add to cart" actions led to purchases?).

AWS Elastic Map Reduce (EMR) is a managed Hadoop product that's designed to be used with S3 for input and output. It allows developers to focus on the jobs rather than the administration / building of Hadoop, and also allows for very large (100's or 1000's of nodes) Hadoop clusters to be spun up on the fly, do their analysis, upload the results to S3, and shut themselves down. Hadoop has something of a learning curve for System Administrators / Operations Staff, but one key thing to know is that it doesn't necessarily require any Java programming ability. I would encourage everybody to try it out.

AWS also offers a product called Redshift, which is a large data store. Data is

internally parallelized and partitioned across multiple disks. Redshift is also optimized for data upload and download to/from S3. Although the functionality is, in this case, similar to Hadoop's, one key difference is that Redshift queries are SQL-based, and it can be accessed via a JDBC driver. In other words, from a client perspective, it looks like a SQL DB.

A thorough discussion of EMR and Redshift could literally be a full course by itself, so we'll leave it here. Of course, there's also the option of building your own log processing / analysis tools, although the time spent to develop it would probably be better spent learning EMR or Redshift. As a shameless plug, the Advanced Ops course contains an in-depth session and lab on analyzing log data with EMR.

The screenshot shows a slide from the AWS Systems Operations on AWS course. At the top left, there's a navigation bar with tabs: 'Systems Operations on AWS' (highlighted in orange), 'Logging | Amazon S3 Keyspace' (in blue), and 'Amazon S3 Keys' (highlighted in blue). At the top right is the Amazon Web Services logo. The main content area contains a bulleted list:

- Focus should be search / analysis
 - Don't use hostname as part of the keyspace
 - Optimize for grouping:
 - Application
 - Function
 - Date (YYYY/MM/DD)
 - Time (optional)
 - Example: storefront/web/2013/09/01/ip-10-0-0-10-access-201309011221.gz

At the bottom of the slide, there's a small copyright notice: "© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved."

When you have a highly scalable application, you'll find that logfiles get pretty big pretty fast. To that end, you want to make sure that your S3 keynames have an naming convention that can be easily organized and searched. The first, slightly counterintuitive lesson is to **not** put the instance hostname in the keyspace – of course, the logfile name should probably contain the hostname or instance id, but you don't want it to be part of the organization.

Later on when you want to analyze your logs, you're probably going to be doing it by application first, then by function, then by date range, so map your keys accordingly.

The screenshot shows a section titled "Lifecycle Rules" under "Logging | Amazon S3 Lifecycle Policies". It lists two main rules: "Expiration (delete)" and "Move to Amazon Glacier".

- Expiration (delete)
 - Compliance => minimum expiration
 - Legal => maximum expiration
- Move to Amazon Glacier
 - After last anticipated log analysis
 - System logs => 1 week
 - Application logs => longest analysis cycle (plus buffer)

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

As mentioned before, logs add up quickly. It's important to manage the lifecycle policies of logs appropriately.

It really comes down to this question:

What do my compliance and legal departments say? Frequently we end up with fixed minimum and maximum lifetimes for backups – compliance team says “we need to keep that data accessible for at least seven years.” Legal says “get rid of that data as soon as you’re allowed to do so”. So, that puts our required lifetime at seven years.

For migration to Glacier, best practice is to keep data in S3 until you run your last anticipated log analysis. In the case of system logs, this might mean migrating logs to Glacier after a relatively short time (1 week or 1 month). For web or application logs, this usually means three months (or probably four months so that you have some buffer).

Remember, there is a charge in Glacier for restoring more than 5% of your data per month, so plan accordingly.



LAB

Store Logs in S3

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Logging | Lab Exercise

Lab Objectives

- Create Amazon S3 bucket for log retention
- Set bucket lifecycle rule
- Configure IAM role for bucket access
- Implement S3 key naming convention
- Create log upload scripts



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

For the lab section, we'll be going through the full setup of an S3 bucket to store our logs, and creating scripts to use that bucket. We'll take into consideration everything in this module, including setting lifecycle policies on the bucket and determining the S3 key naming convention.



Module 11:

Creating an Elastic Infrastructure

(Amazon EC2 Auto Scaling)

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Elastic Infrastructure I Overview

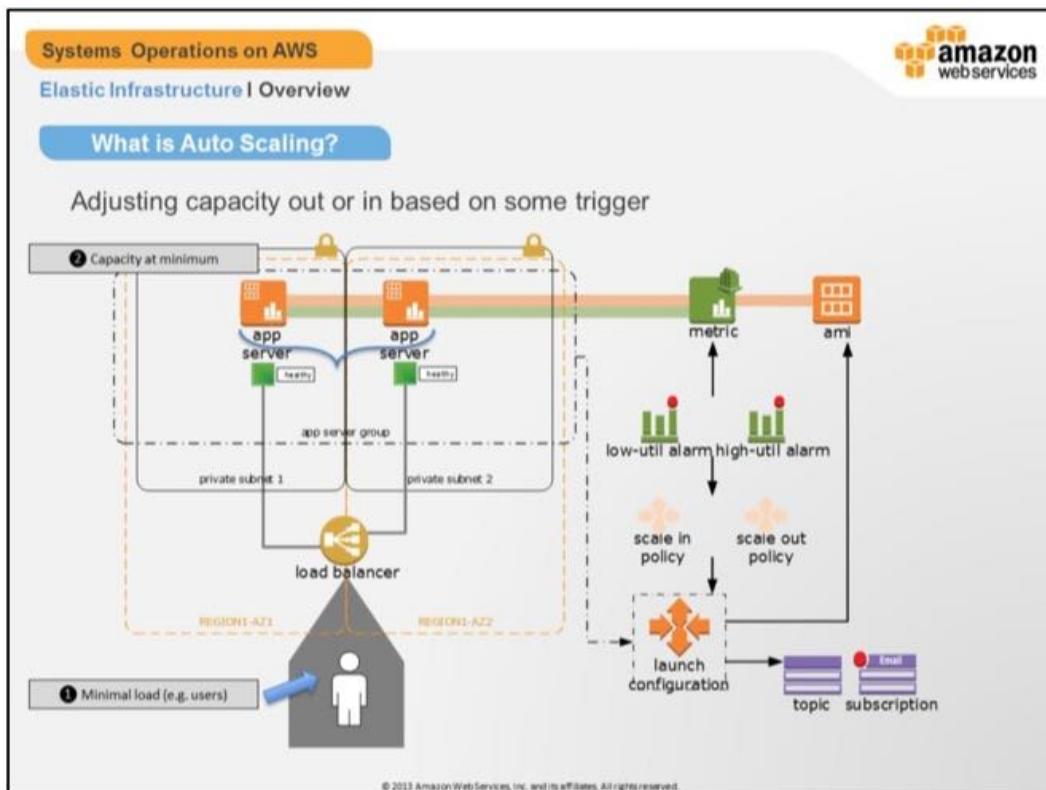
What We Will Cover

- What is Auto Scaling?
- Use cases for Auto Scaling
- Components of Auto Scaling
- Auto Scaling and Amazon CloudWatch
- Auto Scaling and Amazon SNS
- Auto Scaling and Elastic Load Balancers
- Scheduled Auto Scaling
- Auto Scaling with Spot Instances

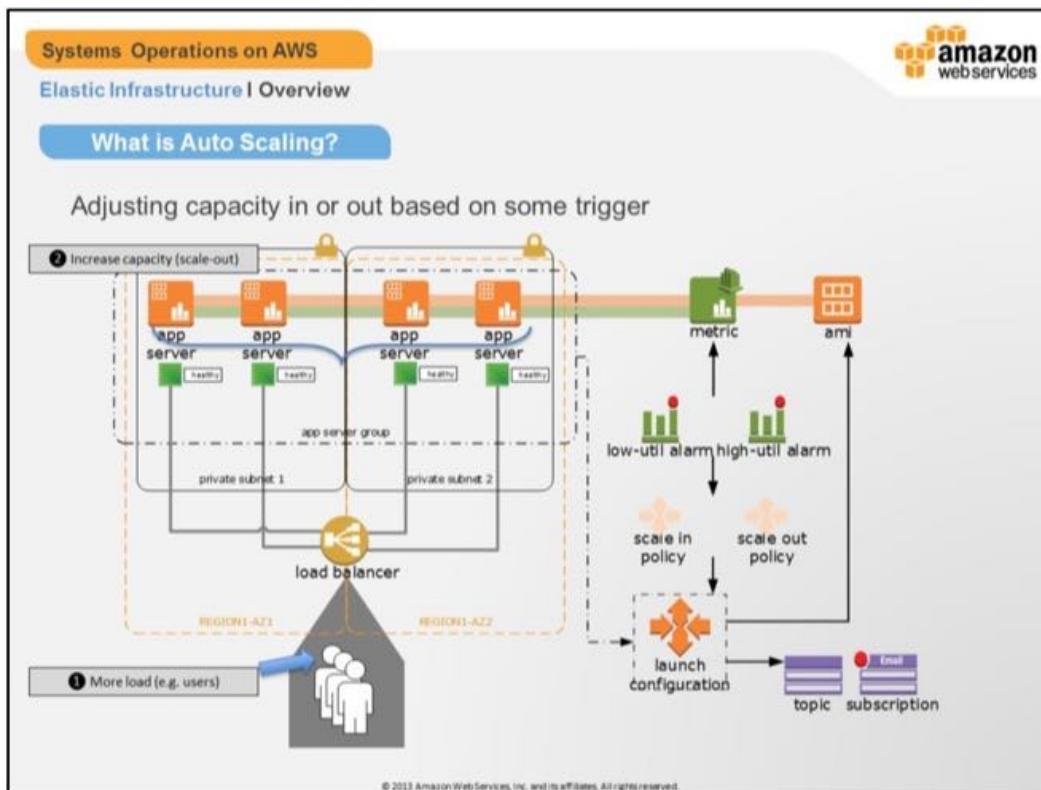
© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

What We Will Cover

- What is Auto Scaling?
- Use cases for Auto Scaling
- Components of Auto Scaling
- Auto Scaling and Amazon CloudWatch
- Auto Scaling and Amazon SNS
- Auto Scaling and Elastic Load Balancers
- Scheduled Auto Scaling
- Auto Scaling with Spot Instances



A trigger could be manually fired or as a result of some other external event, such as a CloudWatch alarm. Alternatively; the trigger may come from within the auto scaling Group itself; if for example the group requires 3 running instances and 1 of them goes down; auto scaling will trigger the creation of an another instance to maintain quorum.



Here we have the same environment, however the load on the application servers has increased (there are more users hitting the servers) – so we scale-out to accommodate the additional load on the system. Likewise, when the load is reduced we can reduce the capacity. This elasticity is great for optimizing costs; why overprovision architectures for the few cases where it may actually be needed? auto scaling allows costs to be minimized as you're only be charged for the increased capacity when it's actually needed.

Systems Operations on AWS

Elastic Infrastructure | Use Cases for Auto Scaling

amazon
webservices

- Web/application tier management
 - Resource based scaling (CPU, disk etc.)
 - Custom metrics (end user experience)
- Embarrassingly parallel workloads
 - Cookie cutter approach for many nodes
- Single instance high-availability
 - Within a single AZ
 - Across all AZs within a region

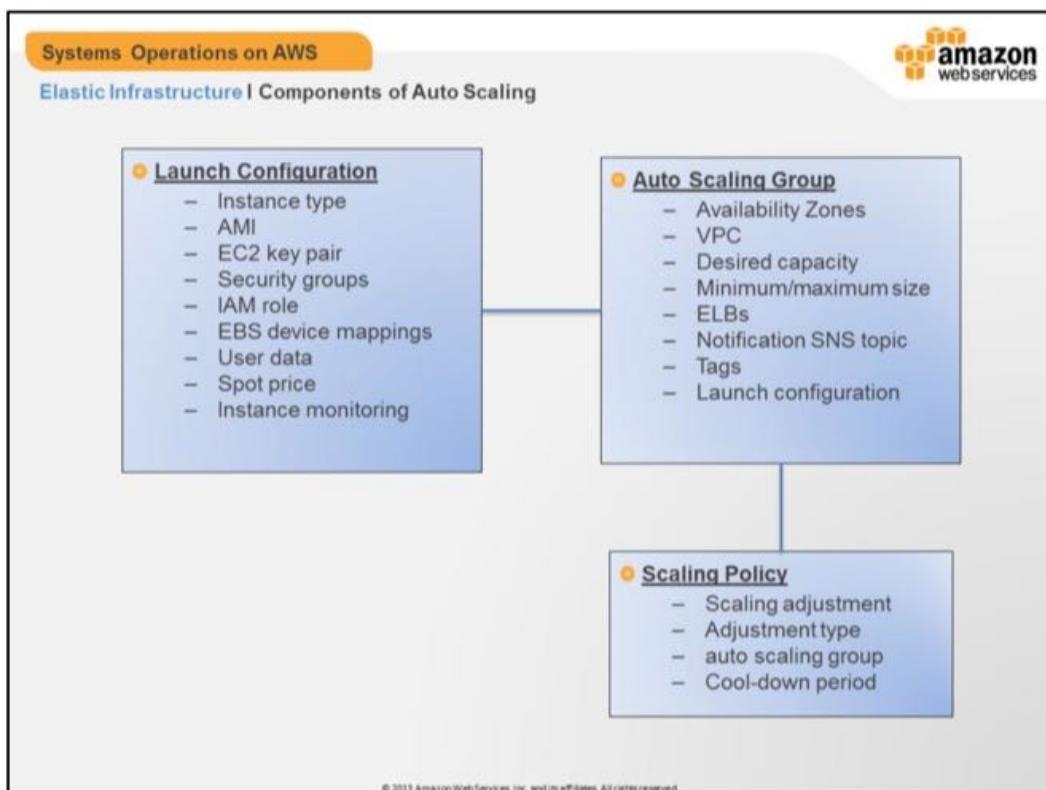


© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Web/application tier management - as we've seen in the previous diagrams, a web/application server architecture is a good fit for auto scaling; keep in mind the applications should typically be REST friendly. The auto scaling can be triggered on a CloudWatch metric such as CPU utilization, or could even be trigger by an external monitoring tool, such as a Business Process Monitor that is monitoring end-user response times (via a custom CloudWatch metric).

Embarrassingly parallel workloads – HPC environments and other large scale clusters are often good fits for auto scaling; each node is typically identical to the next and auto scaling allows you to provision a large number of nodes relatively easily.

Single instance high-availability – Often we just use auto scaling to maintain the availability of a single server. We'll look at auto scaling components later on, but essentially what we're doing is setting a minimum and maximum size of 1 instance; no more, no less; if the instance dies or disappears for whatever reason, auto scaling will kick in and launch another one to take it's place. The auto scaling group allows you to ensure that if we lose an entire AZ, we can always recover the instance in another AZ. Obviously use-case and system architecture would determine if this is the desired approach!



Components of auto scaling are:

Launch Configuration – this is essentially the EC2 instance configuration blueprint. It determines the type of instances and associated AMI that will be used when launching instances as part of an auto scaling activity. Spot price indicates that instances will only be launched if the current spot price is less than this value.

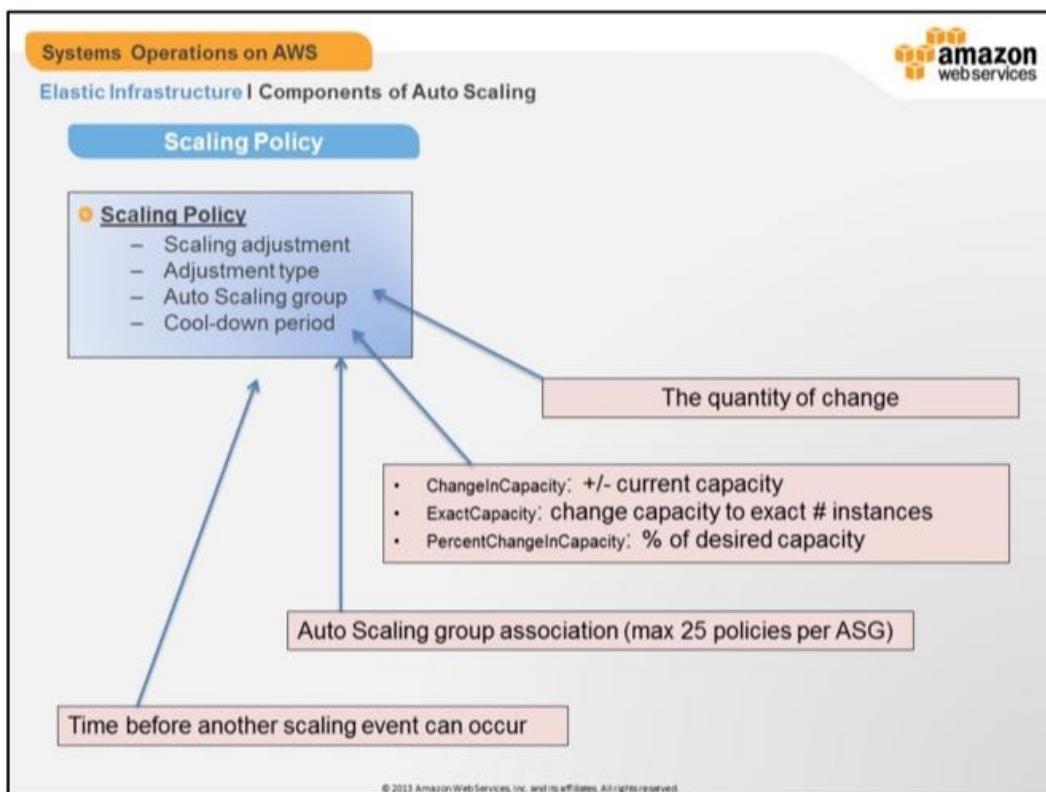
Auto Scaling Group – Defines *where* the instances will be launched, what the constraints are in terms of minimum, desired and maximum capacity, ELBs to register the instances against and SNS topic details which will be notified when any related auto scaling activities take place.

Scaling Policy – These are an optional entry point to auto scaling, a policy is executed according to the configuration (i.e. does capacity need to go up or down?) and it will use the related auto scaling group to identify the upper and lower bounds of instance numbers. Scaling policies are typically executed as an action of a CloudWatch alarm.

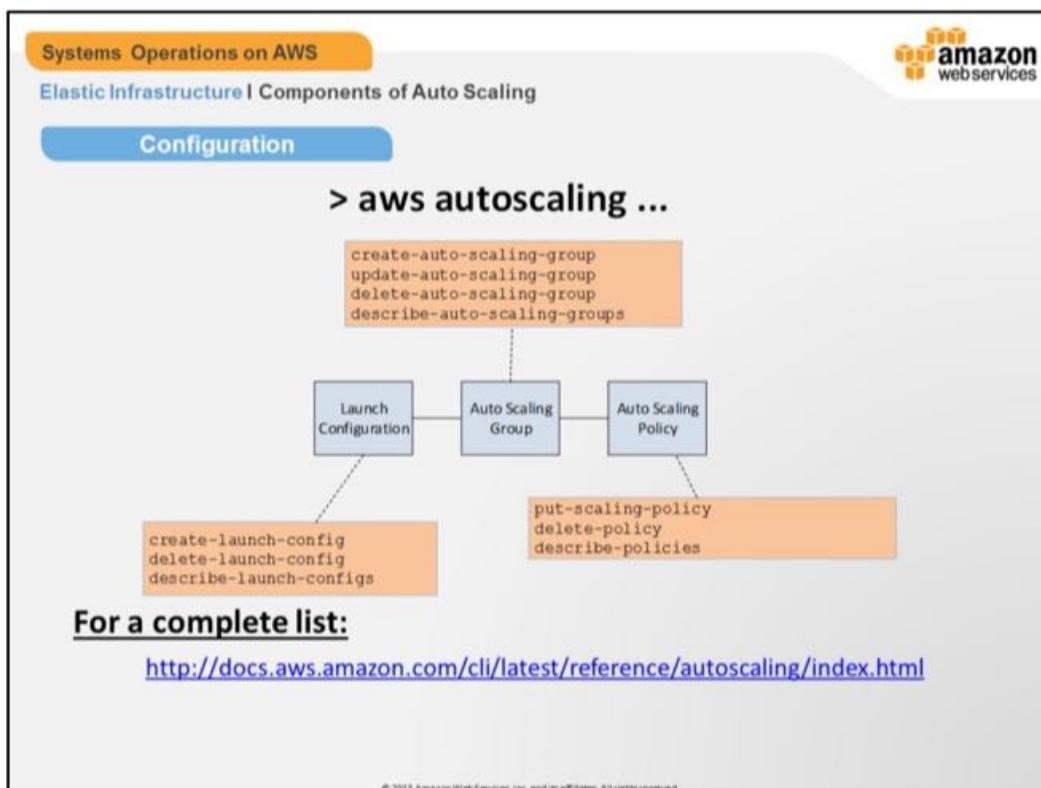
***** Note:** When auto scaling instances within a VPC, you need to specify that VPC subnets for the **vpc-zone-identifier** property when creating the auto scaling group.

We'll see in an upcoming slide how these configuration elements are constructed –

and that is from the Command Line Interface tools; there is no interface for auto scaling within the AWS Management Console!



Scaling Policies deserve a little more attention, because with policies we can determine how the scaling should occur. Policies are used in conjunction with Amazon CloudWatch which we'll see in later slides. For the time being, simply be aware what the policy consists of (the next slide will show the various CLI commands to work with auto scaling, including the ones used for creating and managing policies).



Here we can see the three components of auto scaling and the corresponding CLI commands that are run to *create/update/delete* and *list* each component. This is not the definitive list of commands, go to <http://docs.aws.amazon.com/cli/latest/reference/autoscaling/index.html> for a complete list. And yes; we'll mention it again, there is no auto scaling section in the AWS management console, you will need to learn these commands. The lab will take you through a scenario where you will get some familiarity with the CLI.

Systems Operations on AWS

Elastic Infrastructure | Components of Auto Scaling

Tags

- Tags propagated from group to instances
- Auto scaling group name provided by default
- Create tags with aws autoscaling:
 - create-auto-scaling-group
 - create-or-update-tags
- Update tags with aws autoscaling:
 - create-or-update-tags
- List tags aws autoscaling:
 - describe-tags
 - describe-auto-scaling-groups
 - describe-auto-scaling-instances

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Tags allow you to organize your resources efficiently; you can use tags to describe the application, version, cost center and so on. As we'll discuss at other points in this course, tags are also very useful when it comes to detailed billing, as we can filter on them to identify individual charges, e.g. for a specific application or project.

With auto scaling you create tags against the auto scaling group, then whenever any auto scaling activities in relation to that group occur, the defined tags will be propagated to the launched instances. By default the auto scaling group name tag (prefixed with aws) will be created for you, this allows you to easily identify which instances belong to which auto scaling groups.

Systems Operations on AWS

Elastic Infrastructure | Auto Scaling & Amazon CloudWatch

Amazon CloudWatch

- Provides monitoring for AWS resources, e.g. EC2
 - CPU utilization
 - Network In/Out
 - Disk Read/Write Ops
- Aggregate and per resource metrics
 - CPU utilization across Auto Scaling Group
- Supports custom metrics
- Alarm actions
 - Notify an SNS topic
 - Execute an auto scaling policy
- Optional for Auto Scaling

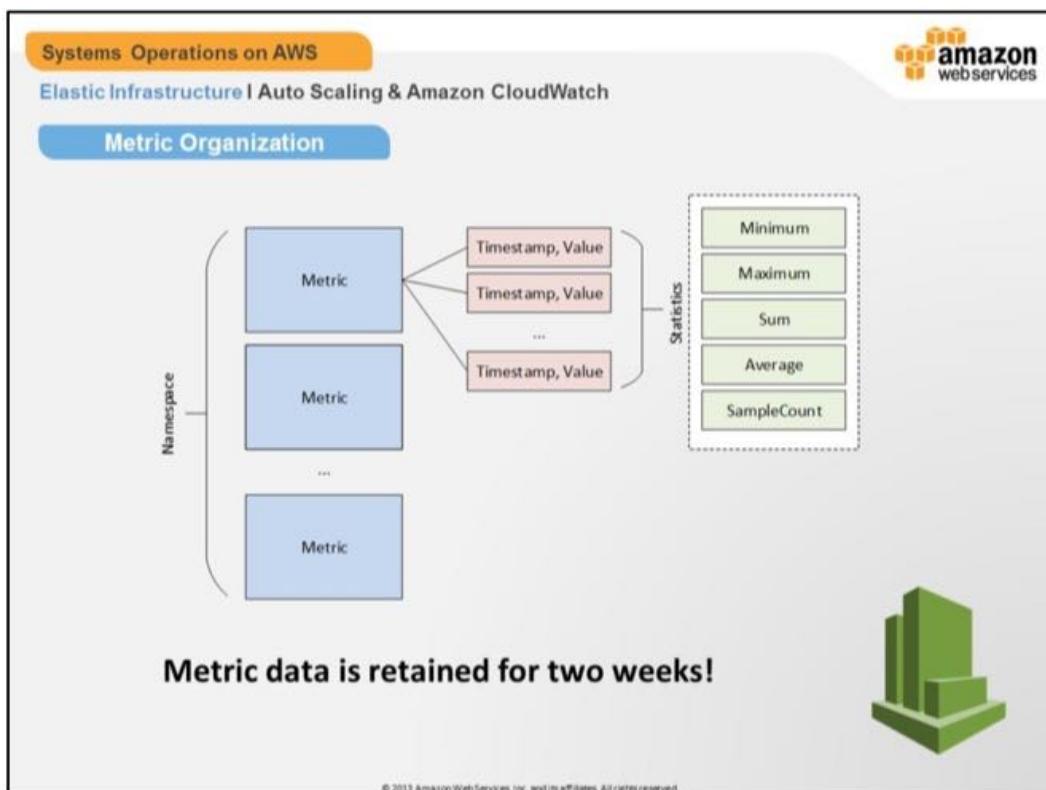


© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

We're now going to depart temporarily from auto scaling to discuss Amazon CloudWatch, an integral component of auto scaling (in terms of a trigger) and elastic infrastructure in general (ELB health and statistics etc.)

Amazon CloudWatch provides monitoring for AWS resources such as EC2, RDS, Elastic Load Balancing and so on. For this module we're primarily interested in CloudWatch for the purpose of elasticity; so our CloudWatch alarms will be executing auto scaling policies, this is known as "*scaling on demand*". We could use custom metrics to trigger auto scaling events too – the point being made is that we should consider the types of metrics that are reflections of application performance and use those when creating alarms to execute auto scaling policies. Recall we discussed scaling policies and the associated properties in a previous slide; the policy determines *how* the scaling event should happen.

Lastly; it's important to note that CloudWatch is completely optional when working with auto scaling, as we'll see later in this module we have other ways to trigger auto scaling events such as adhoc using the CLI, or via scheduling them.



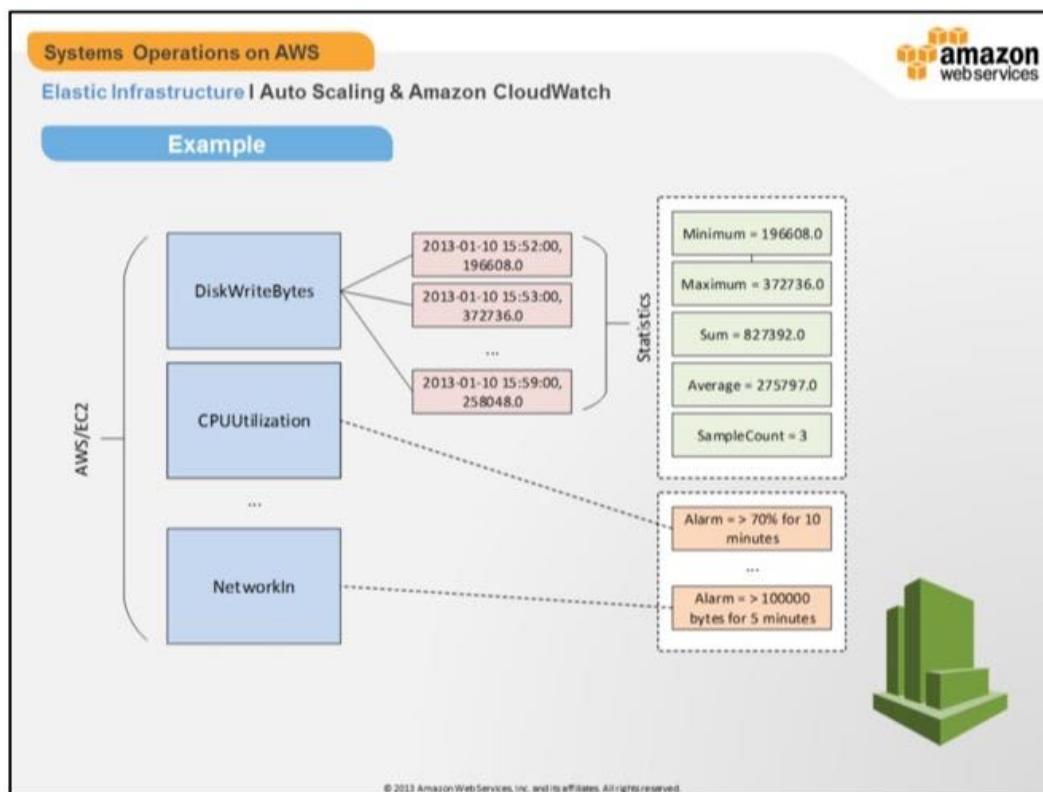
Amazon Cloudwatch is schema-less but there are relationships between the various components that infer some kind of structure. CloudWatch is organized at the highest-level by a namespace; a namespace provides a way of categorizing the metrics. All metrics that are published by AWS services are prefixed with *AWS*. For example, EC2 is *AWS/EC2* and auto scaling is *AWS/AutoScaling*. When publishing custom metrics you can choose whatever namespace you like. Each metric has a unit of measure, the *unit* denotes this – some common units are *percent*, *bytes* and *seconds*.

Data samples are the metric's measurements, and essentially consist of the time the measurement was taken and the actual value of the measurement. Over time these measurements or data samples can be aggregated into *Statistics*. Statistics are aggregates of data samples over a specific time period.

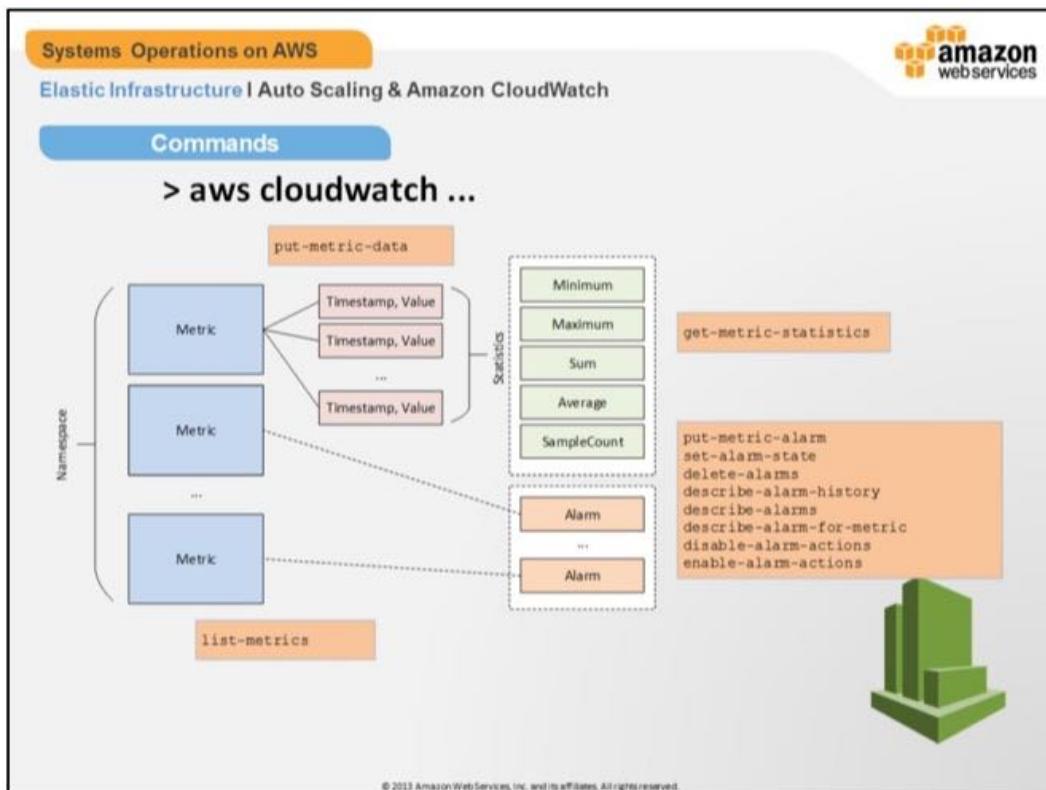
One other aspect of metrics is the *dimension*. A dimension is a name/value pair that uniquely identifies a metric. Example, *instance-id* allows us to retrieve metrics associated with a specific EC2 instance.

The next slide will show an example.

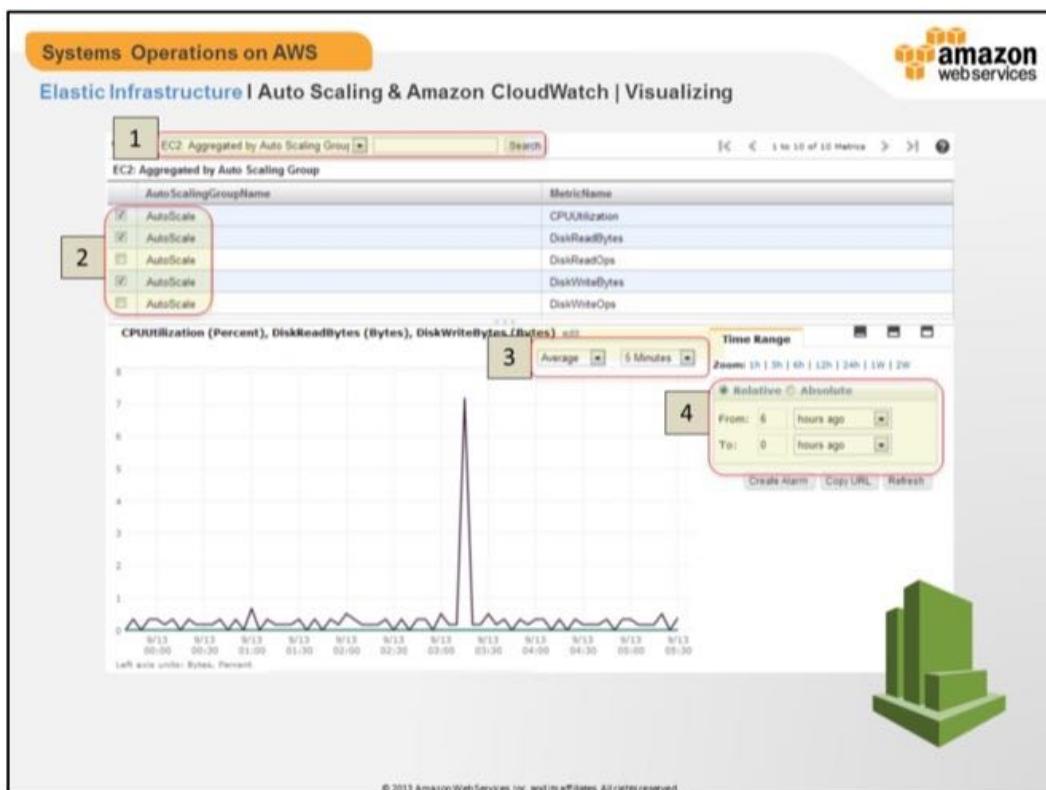
***** Note:** Metric data is only stored for two weeks. If you want to keep data longer you'll have to come up with your own strategy for scraping the metrics and storing them in some other data store (DynamoDB perhaps?)



Let's look at the previous metric organization slide decorated with something meaningful. Here we're looking at 3x metrics from the *AWS/EC2* namespace, *CPU Utilization*, *DiskWriteBytes* and *NetworkIn*. With *DiskWriteBytes* we can see three samples over period of time (realistically there would be 8 samples as our time period goes from 15:52 – 15:59; but for the sake of this example; we'll treat it as three samples). The *Statistics* are now calculated over the time period and the values shown in the graph. Also note the two other metrics that are showing the association with alarms. Likewise, the other two metrics (*CPUUtilization* and *NetworkIn*) would also have data samples.



Here is the same slide we just saw (i.e. components of Amazon CloudWatch) with the various CLI commands that can be run to interact with CloudWatch.



You can visualize CloudWatch metrics as graphs from the AWS Management Console.

1. You can select the namespace and optionally filter to find the metric you're interested in visualizing
2. Choose the resources you're interested in (*dimensions*) – multiple items can be selected and visualized in the same graph
3. Specify the *statistic* to show, remember we just talked about statistics as being an aggregate over time of our raw data samples
4. Determine the time range you wish to view the metric over

Also from this page, you optionally have the choice to create a CloudWatch alarm or *Copy URL*, which effectively gives you a URL that you can copy and reuse to generate the exact same graph in the future without having to re-apply all the parameters.

Systems Operations on AWS

Elastic Infrastructure | Auto Scaling & Amazon CloudWatch

Alarms

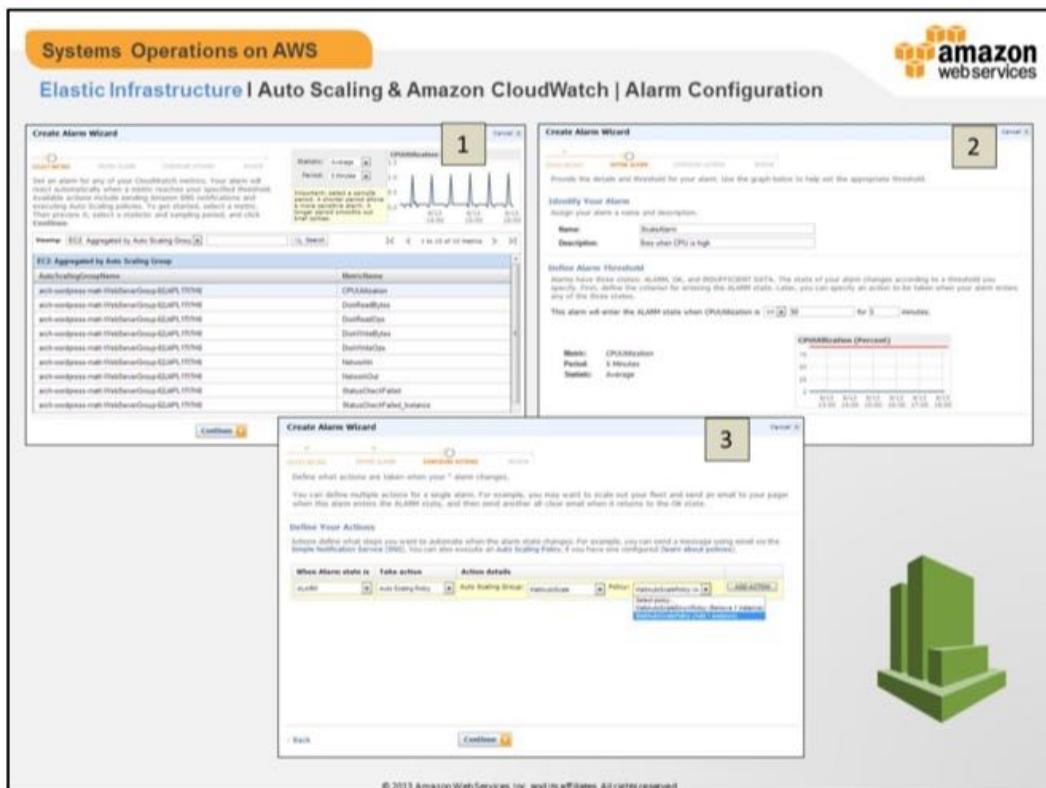
- CloudWatch alarms:
 - Send an SNS message
 - Execute an auto scaling policy
- An alarm has one of three possible states:
 - OK
 - ALARM
 - INSUFFICIENT_DATA
- Specific to a single metric over time
- Action invoked due to change in state

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Alarms allow you to send an SNS message or execute an auto scaling policy based on the characteristics of a single metric. For example, CPU utilization has been greater than 90% over a period of an hour. Depending on the metric condition the alarm will be in one of three states as shown here (OK, ALARM or INSUFFICIENT_DATA). While an SNS message is triggered by a change in the alarms state, auto-scaling is actually 'continually' triggered when an Alarm is in ALARM state.

***** Note:** The current default limit for the # alarms that can be configured per account is 5000.



If you recall in previous slides we discussed the various components of auto scaling. One of those components was the scaling policy. When using CloudWatch and auto scaling; you configure a threshold for a particular metric; when the threshold is breached or the condition is met a CloudWatch alarm will be raised and trigger an action to “execute” the scaling policy.

1. Decide which metric will be used to trigger the auto scaling event – the one we’re using here is *CPU Utilization %* across the auto scaling group.
2. Give the alarm a *name*, *description* and specify the *threshold* condition.
3. Lastly ensure that the alarm action is *Auto Scaling Policy* and then select the required auto scaling group and policy (in our example here we have two policies; one to scale out and one to scale in).

Systems Operations on AWS

Elastic Infrastructure | Auto Scaling & Amazon SNS

Overview

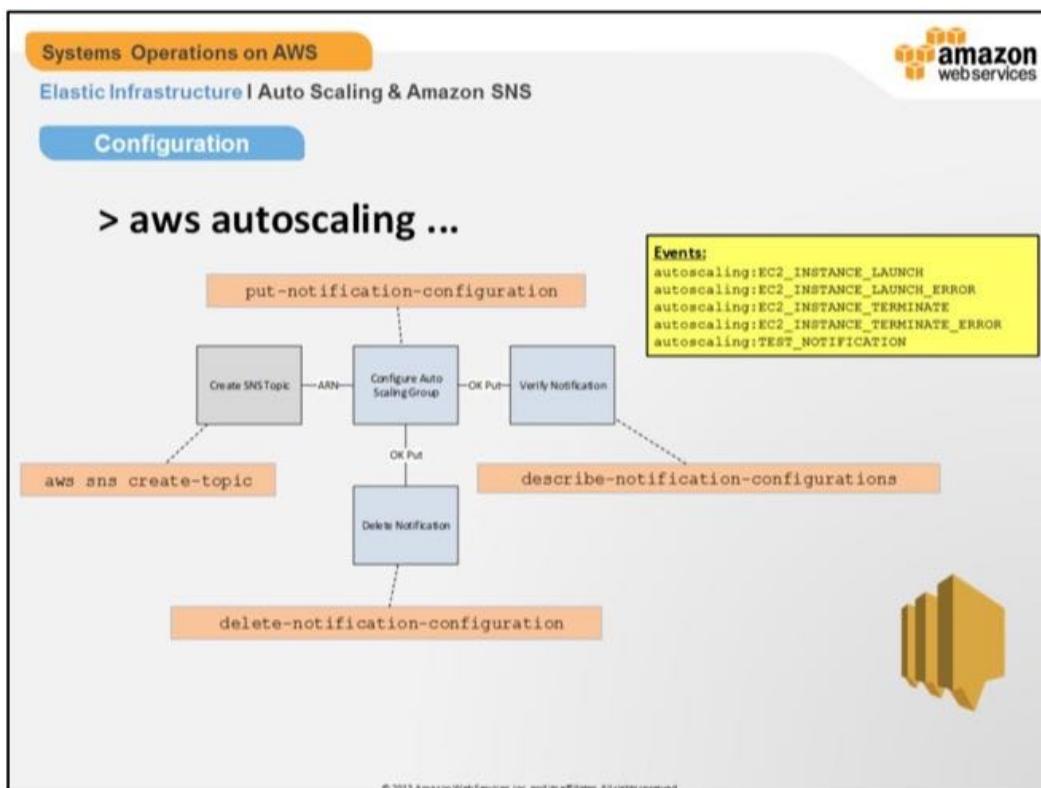
- Managed messaging service
- Messages are published to a topic
- Topic subscribers are notified
 - E.g. e-mail endpoint
- Events specific to auto scaling
- Command line tools for configuring

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Amazon SNS is a managed service utilizing the pub-sub pattern (publish-subscribe). SNS can be used whenever you wish to notify a large number of endpoints quickly and easily whilst providing a scalable, reliable messaging service. When you want to be *notified* when something happens in auto scaling, e.g. an instance has launched or terminated you use SNS to facilitate it. The next slide will go into more detail in regards to the commands and types of events that you can configure but in short:

You create (or leverage an existing) SNS topic and tell the auto scaling group (ASG) to *notify* that topic when an auto scaling event of interest occurs; subsequently notifying any of the topic subscribers.



The general process once the SNS topic has been created is to use the command line tools to configure the notification (remember: no console for auto scaling!), the following will configure the auto scaling group to send notifications:

```
aws autoscaling put-notification-configuration <AutoScalingGroup> --topic-arn
<ARN> --notification-types <Events>
```

Once the auto scaling group has been configured for sending notification, you can do the following to verify it by display the notification headers:

```
aws autoscaling describe-notification-configurations --auto scaling-groups
<AutoScalingGroup> -headers
```

When you wish to remove notifications run the following:

```
aws autoscaling delete-notification-configuration <AutoScalingGroup> --topic-arn
<ARN>
```

When the configured auto scaling groups generate any of the notification types that were chosen; anything subscribed (e.g. e-mail address) to the SNS topic will receive the notification.

***** NOTE:** The list of auto scaling notification types are listed in the yellow box.

Systems Operations on AWS
Elastic Infrastructure | Auto Scaling & Elastic Load Balancing

Overview

- Distribute traffic across multiple EC2 instances
 - Single AZ or Multi AZ
- Auto-scales based on incoming traffic load
- Implements health checks for EC2 instances
- Supports sticky sessions, IPv4, IPv6 and SSL
- ELB metrics in Amazon CloudWatch
- auto scaling groups register with an ELB
 - Load balances across dynamic ASG fleet
- Use ELB health-checks for ASG instances
 - `aws autoscaling update-auto-scaling-group <ASG> \--health-check-type ELB --grace-period <Period>`

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Elastic Load Balancers distribute incoming traffic across multiple registered EC2 instances and they automatically scale based on the incoming traffic load. ELBs are often used to distribute traffic to web servers to handle load; likewise we also use auto scaling to handle load, but in a different way – we scale the number of instances on demand.

We need to be able to scale at the load balancer tier AND the web server tier. The ELBs scale themselves and we can implement auto scaling to scale the web server tier, but how can we integrate the two if the elastic server fleet keeps changing in # of instances? Well, we simply register the auto scaling group with the ELB, that way when the capacity of the auto scaling group changes, instances will be automatically registered with the load balancer (and subsequently removed).

In terms of EC2 health checks, both ELBs and auto scaling implement their own. If the auto scaling group is registered with an ELB, you have the option of using the ELBs health checks to manage the instances in the auto scaling group. Note the command shown here, simply run the `aws autoscaling update-auto-scaling-group` with the `--health-check-type` attribute set to `ELB`.

***** Note:** from the documentation: “*If there are multiple load balancers associated with your Auto Scaling group, Auto Scaling will check the health state of your EC2 instances by making health check calls to each load balancer. For each call, if the Elastic Load Balancing action returns any state other than InService, the instance*

will be marked as unhealthy. After Auto Scaling marks an instance as unhealthy, it will remain in that state, even if subsequent calls from other load balancers return an InService state for the same instance.”

Systems Operations on AWS

Elastic Infrastructure | Auto Scaling & Elastic Load Balancing | ELBs & VPC

ELBs & VPC

- Elastic Load Balancers in VPCs can be:
 - Internet facing
 - DNS resolves to a public IP address
 - Internal
 - DNS resolve to a VPC IP address



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Elastic Load Balancing in VPCs present two options for your load balancers. Keep in mind that they're both available: i.e. load balancing the VPC doesn't force you to the internal option.

The key difference is the kind of interface the DNS record that's created for the ELB will resolve to: a public or private address. Note that the DNS record is publically resolvable in either case.

Another difference is specifying Availability Zones (AZs) versus a VPC subnet per AZ. Note that internal load balancer interfaces will only go into the same subnets where your instances reside, in keeping with the basic functionality ELB provides -- an interface that distributes traffic among some others.

Systems Operations on AWS

Elastic Infrastructure | Scheduled Auto Scaling | Overview

Overview

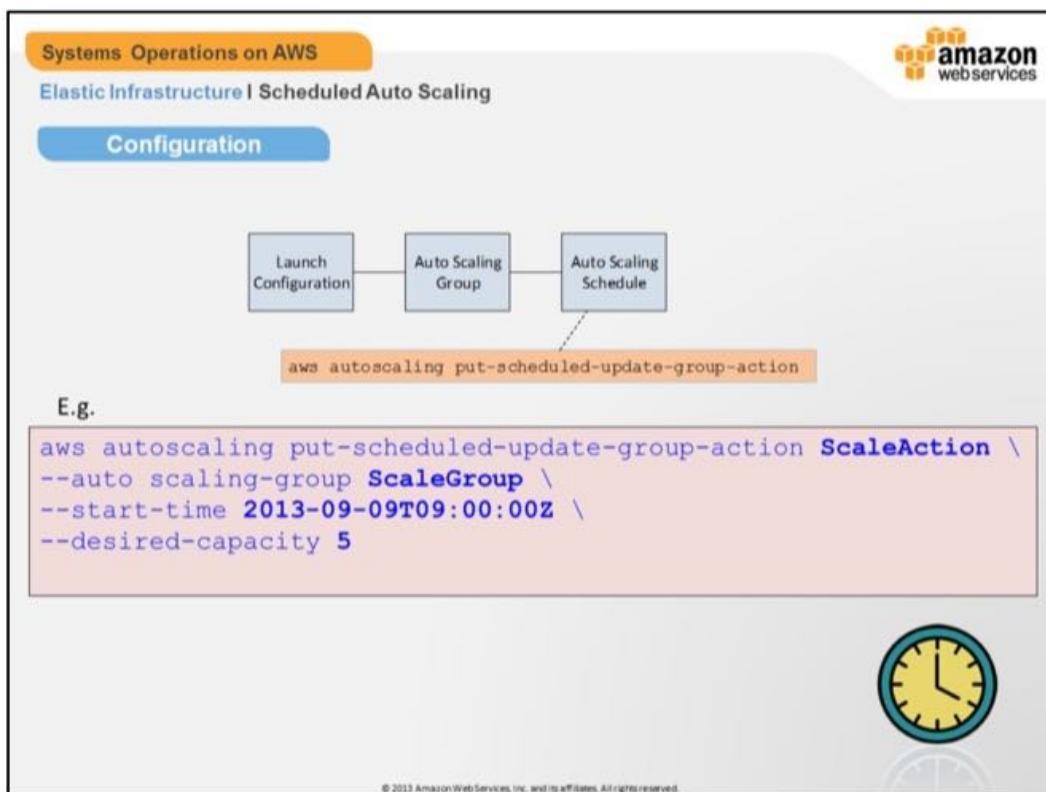
- Scale based on a schedule
- For predictable work-load variations
- Possible delay of ~1-2 minutes from schedule start
- 125 scheduled actions per month per ASG
- Scheduled actions must have unique time value
- Schedule up to a month in advance



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

If your environment has predictable work-load patterns, i.e. at certain times you know that load will increase, then you have the option of triggering auto scaling events based on a schedule. For example, at noon on a particular date you may wish to increase capacity to 5 instances, and at midnight (perhaps when the load will have been alleviated) you can schedule another auto scaling event to set the capacity back to a lower number.

Scheduled auto scaling actions typically start immediately upon the schedule time occurring, however there may be a ~1-2 minute delay on occasions. Also note that you can schedule up to 125 actions per auto scaling group per month; and you can schedule up to a month in advance.



Use the `as-put-scheduled-update-group-action` command to create auto scaling schedules, for example:

```
aws autoscaling put-scheduled-update-group-action <ScheduleName> --auto scaling-group <AutoScalingGroup> --start-time <YYYY-MM-DDThh:mm:ssZ> --desired-capacity <Capacity>
```

The example on the slide here is saying: “On September 9th 2013 at 9:00am, adjust the capacity of the auto scaling group ‘ScaleGroup’ to 5 instances”.

Systems Operations on AWS

Elastic Infrastructure | Auto Scaling with Spot Instances

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- Auto scaling makes spot bids on your behalf
 - Persistent bidding
- Bid price set in Launch Configuration
 - New bid price = new launch configuration
- Capacity is dependent on spot price
 - Instances terminated when bid price < spot price
- Track spot pricing history to assist in configuring AS
 - E.g. Do scheduled scaling actions look viable?



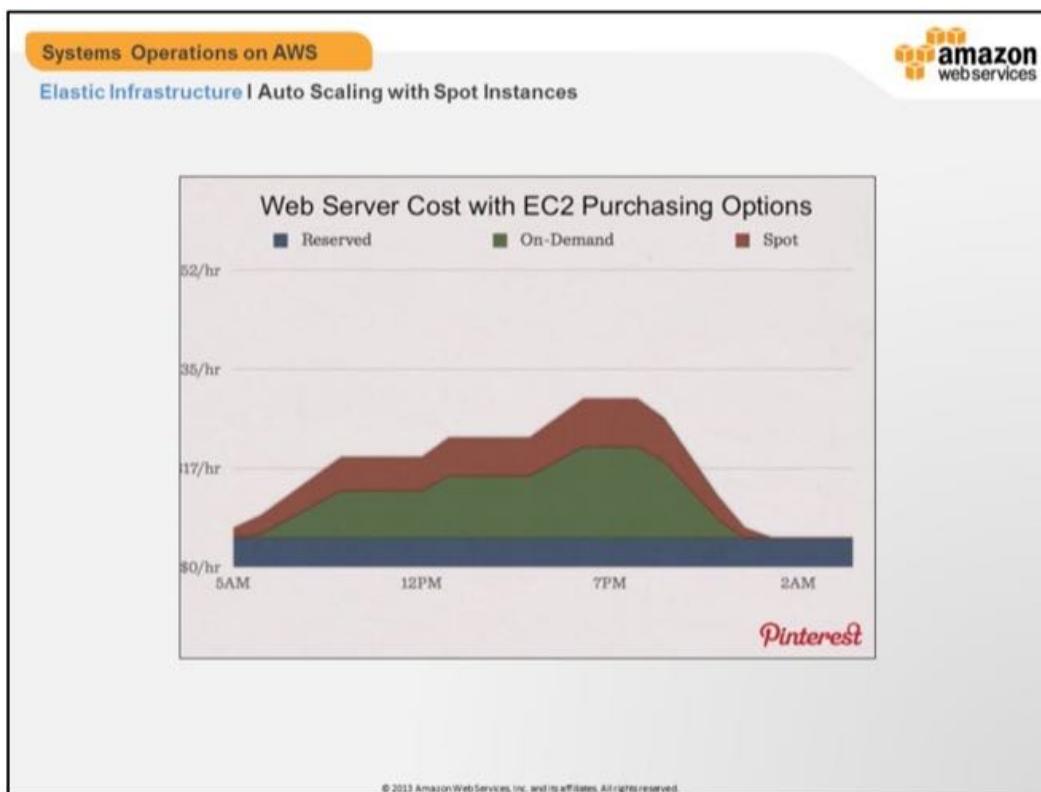
Auto scaling goes well with spot instances; especially if the scaling tier supports a RESTful like interface, spot instances can be terminated without notice if the bid price falls below the spot price; so applications need to be architected in such a way as to allow instances to be removed without negatively impacting the system; embarrassingly parallel systems typically support a RESTful like interface and removal of instances shouldn't pose problems to the application. Auto scaling with spot instances is as simple as specifying a bid price in the *launch configuration*. A thing to note is *launch configurations* cannot be modified, so if you want to change the bid price you'll have to create a new *launch configuration*.

As long as your bid price exceeds the spot price, auto scaling will treat the instances as if they were on-demand. I.e. if a new instance needs to be launched and the spot price is less than the bid, then the instance will launch. If it isn't then an instance will not be launched. Likewise if the auto scaling group consists of spot instances and the bid price is less than the spot price then the instances will be terminated.

When working with scheduled scaling actions it's a good idea to observe historical spot pricing to see if your schedule will be a good fit for spot instances in terms of acceptable bid prices.

Lastly, it's good practice to leverage reserved instances with spot and on-demand scaling groups. I.e. have a fixed number of reserved instances to handle the *known*

load and to be always available. Then, have two auto scaling groups, one for spot and the other for on-demand instances. Whilst the spot price is less than the bid you can leverage the price advantages of the spot instances and scale the on-demand group down; when the price moves in favor of on-demand (i.e. the spot price exceeds the bid price), scale the on-demand group. This would require some kind of external worker monitoring the groups and making adjustments as necessary.



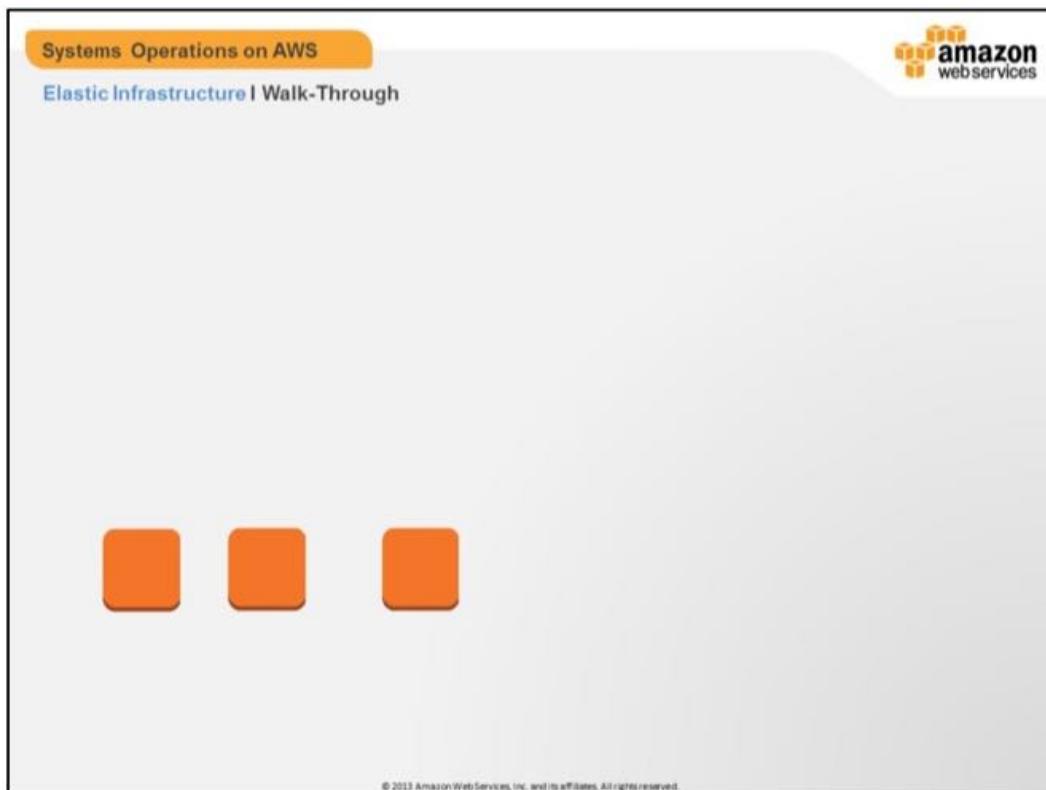
This is what we were just talking about. The graph here is taken from Pinterest (this is public and can be acknowledged!); note that there is a fixed tier of reserved instances (blue); and then there are two auto scaling groups one for spot (red) and one for on-demand (green). You'll see that they've lowered their costs by adopting spot instances and not just on-demand. Consider the costs had spot instances not been employed!



Amazon EC2 Auto Scaling Walk-Through

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The following slides step through the various phases of auto scaling; everything we've covered up to now is present here in some form. Feel free to ask questions or provide feedback during the walk-thru.



Here we have three EC2 instances; not much going on here. No auto scaling is at play. This is not elastic. We can not convert these instances to auto scale instances. We have to launch it through an auto scaling policy that specifies an AMI

If we want to work with auto scaling, we need to do some prerequisite configuration.

Systems Operations on AWS

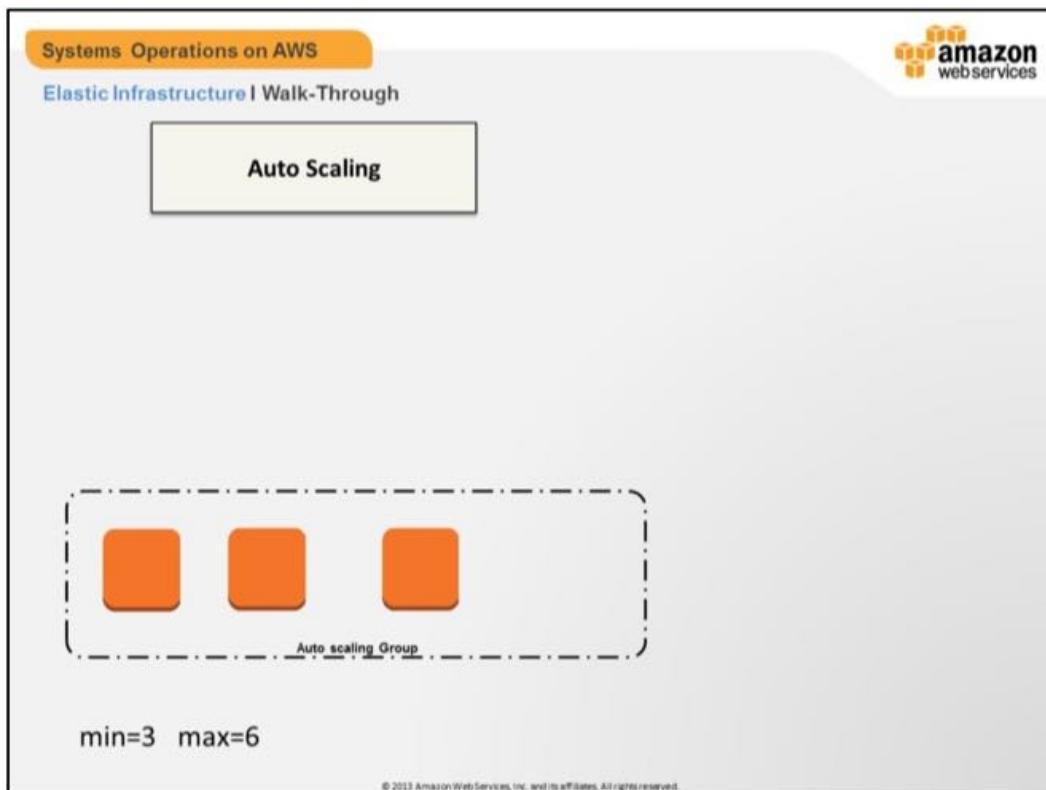
Elastic Infrastructure I Walk-Through



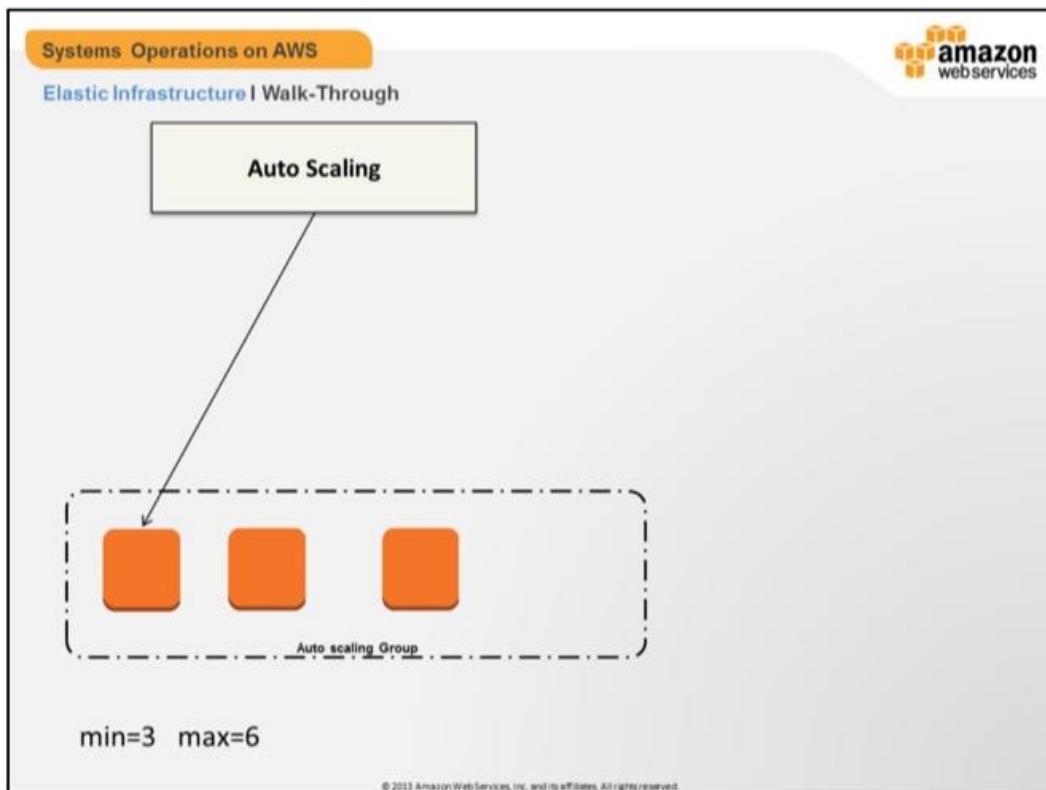
min=3 max=6

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

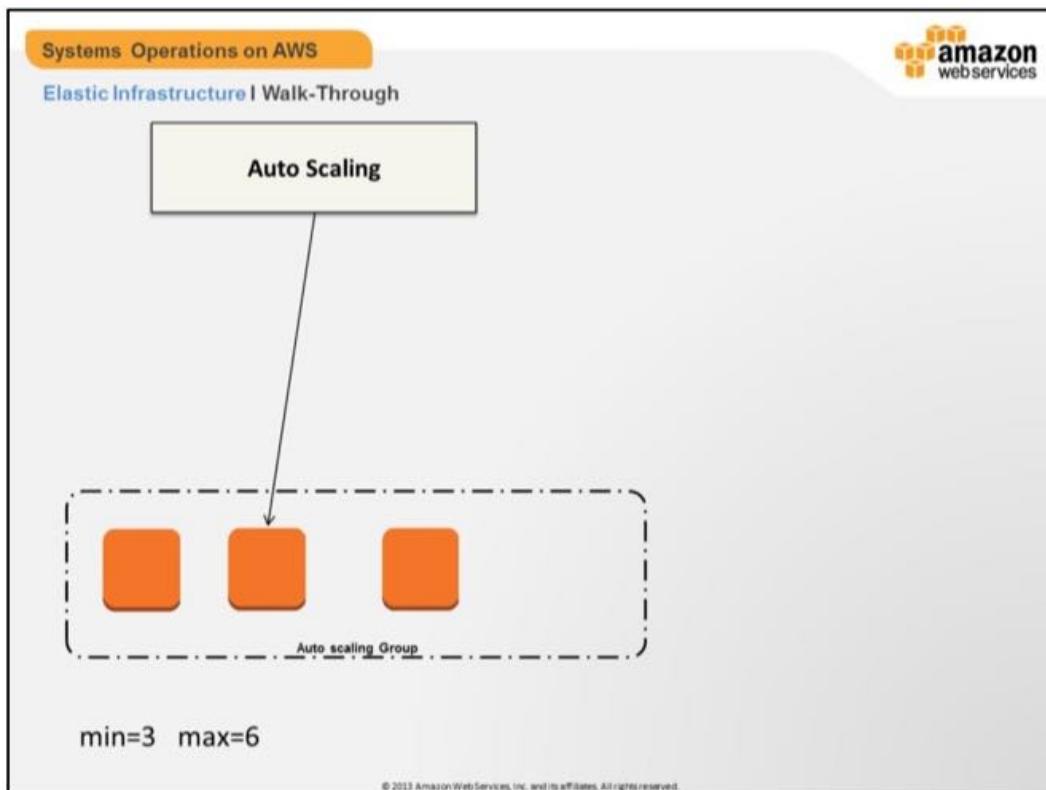
Let's set some constraints around what we would like; an auto scaling group with a minimum of 2x instances and no more than 6x.



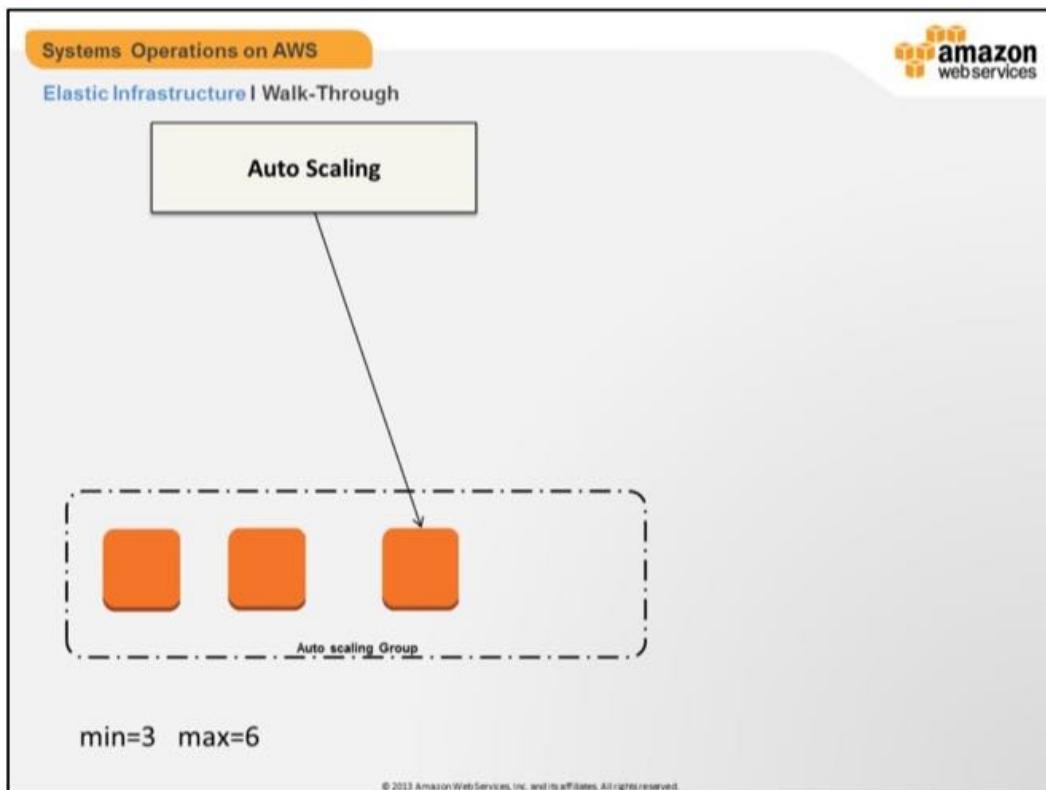
Now we will bring auto scaling onto the scene, it will govern/control the lifecycle of the instances.



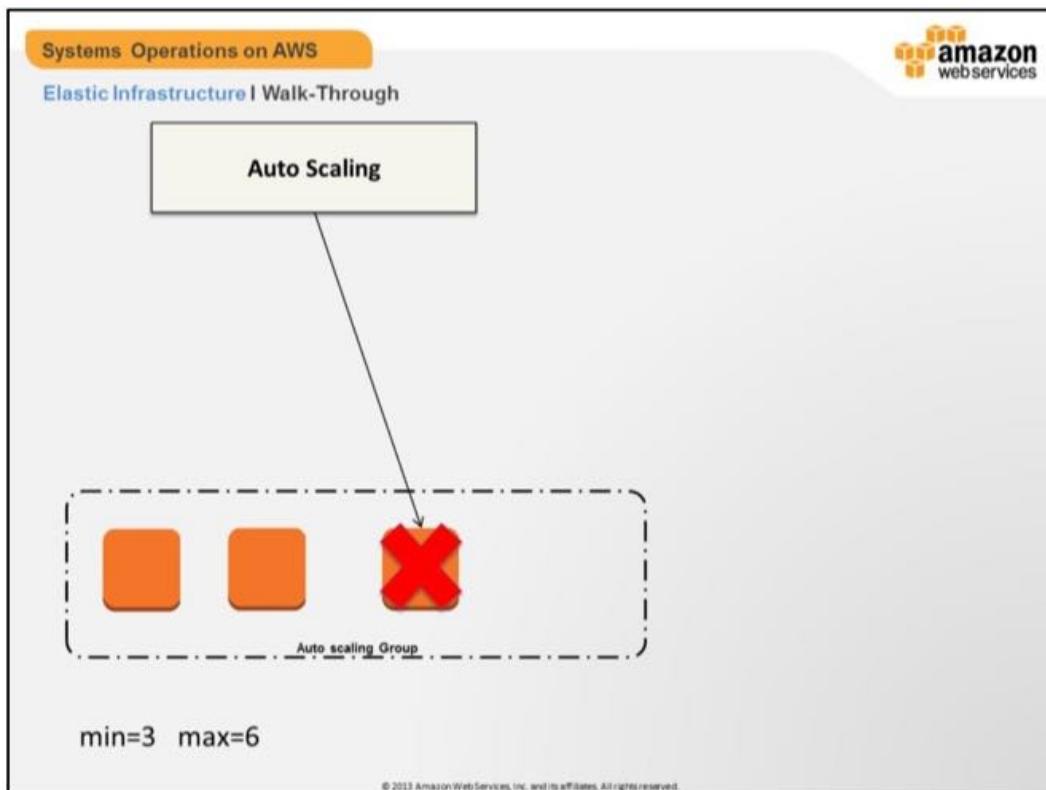
Auto scaling monitors the first instance; all is well!



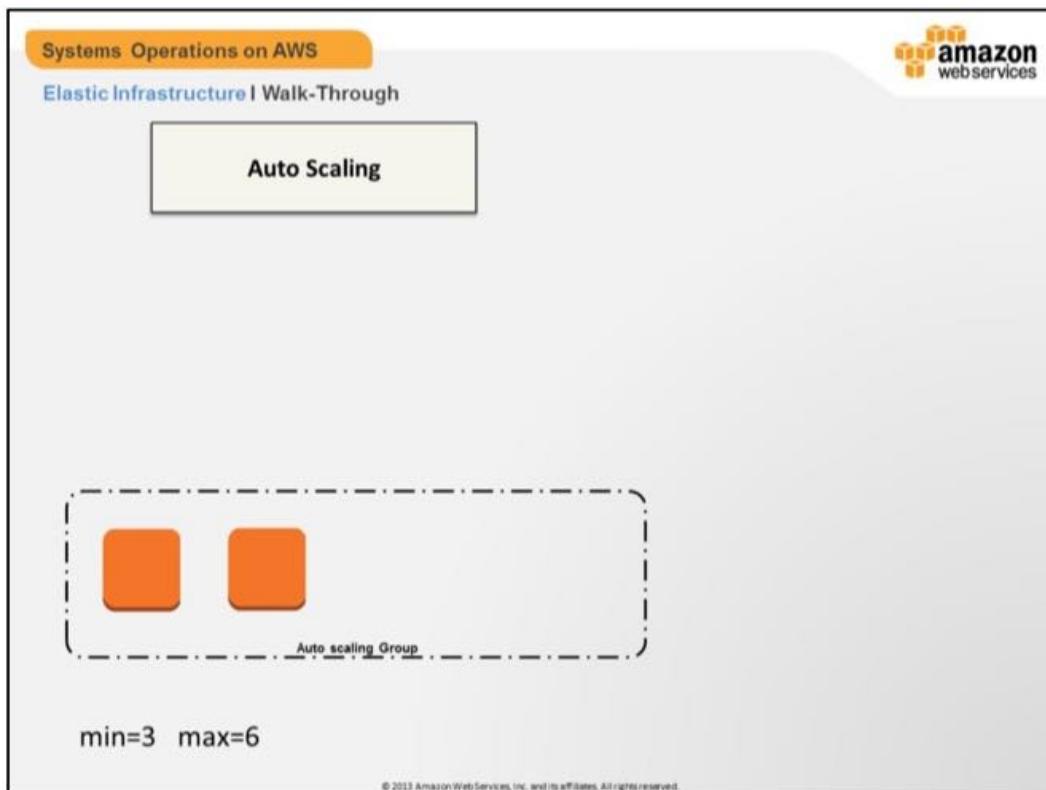
Auto scaling monitors the second instance; all is well!



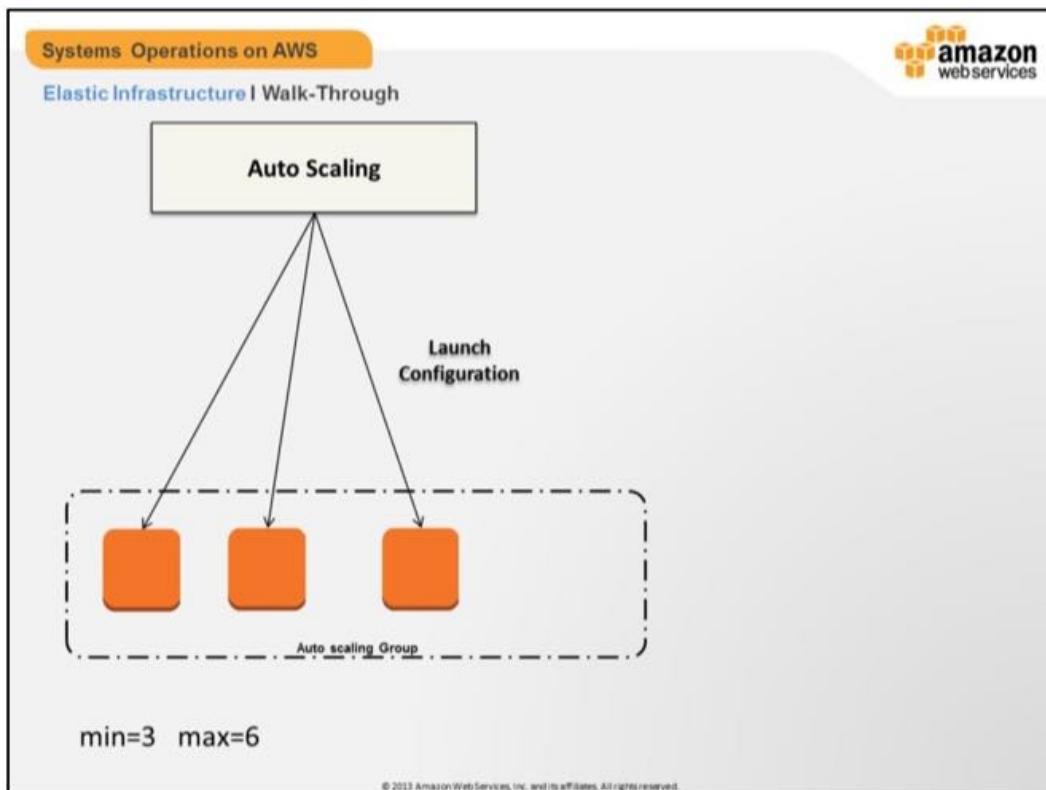
Auto scaling monitors the third instance; all is well... for now!



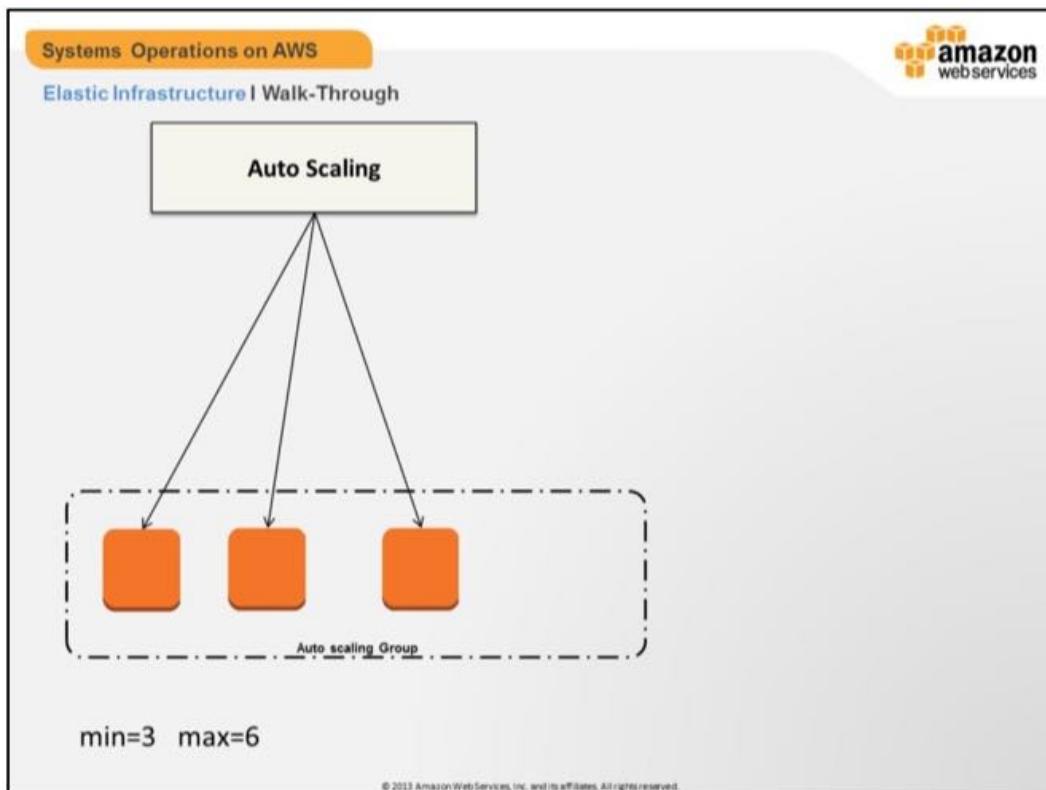
Auto scaling has noticed the third instance has failed



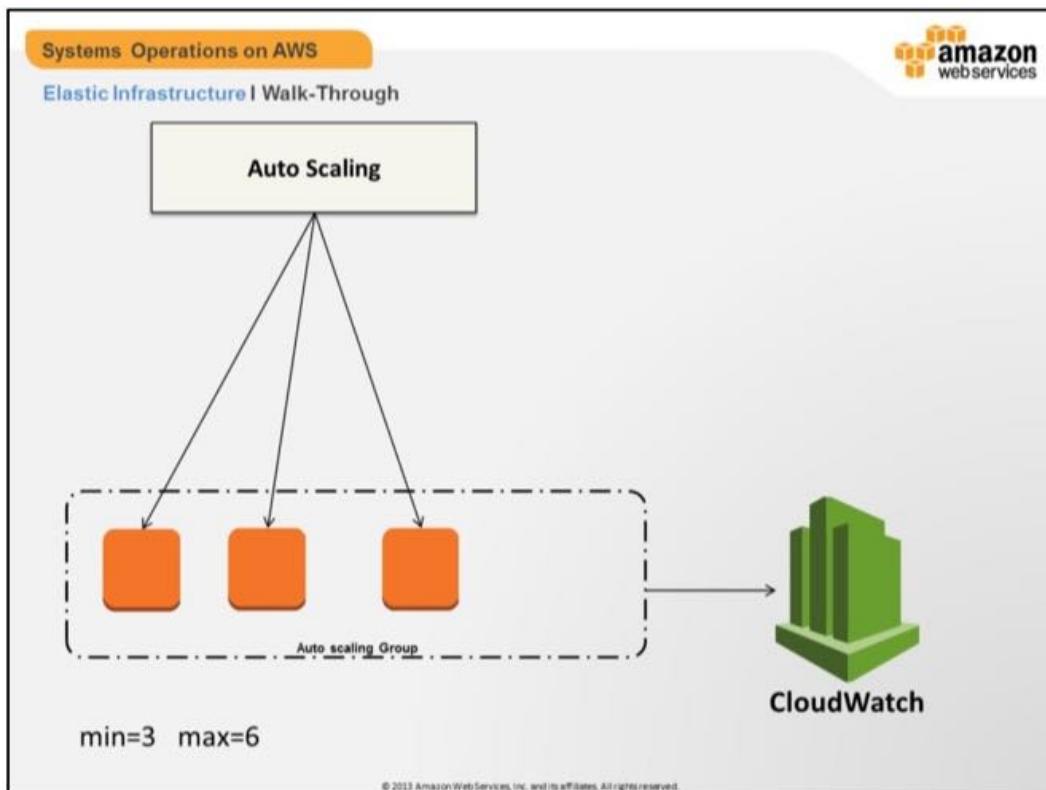
The unhealthy instance is removed from the auto scaling group.



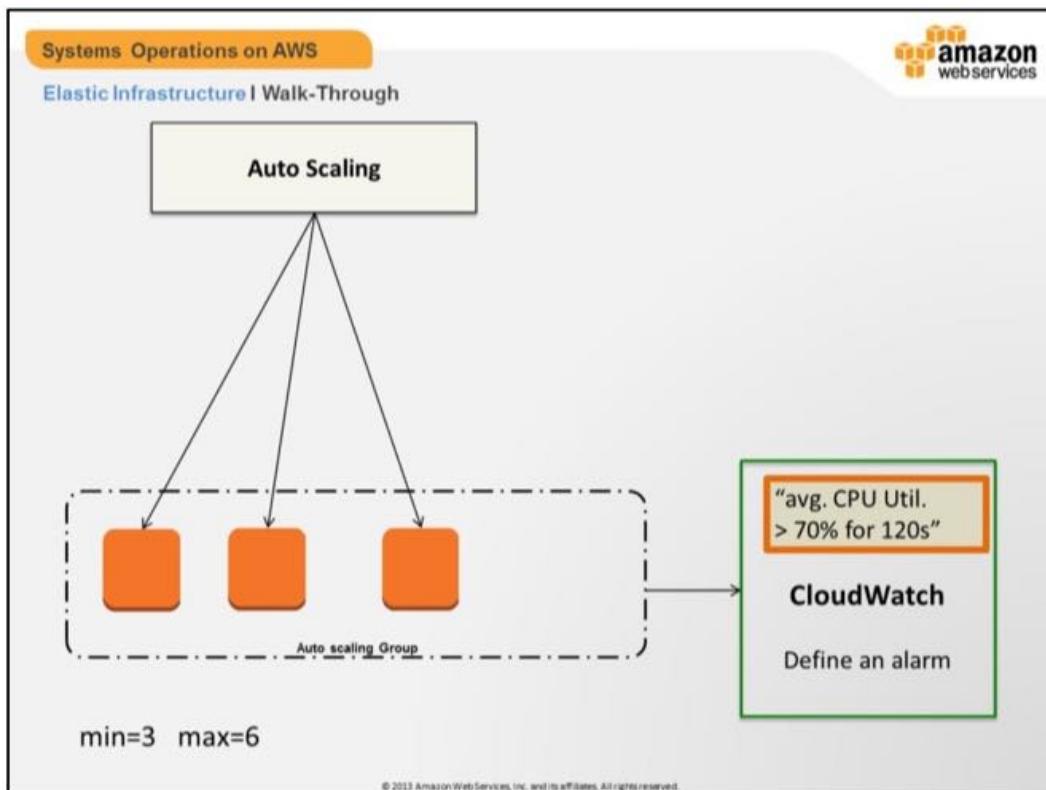
A new instance is launched (based on the launch configuration) into the auto scaling group to replace the unhealthy instance that was just previously removed.



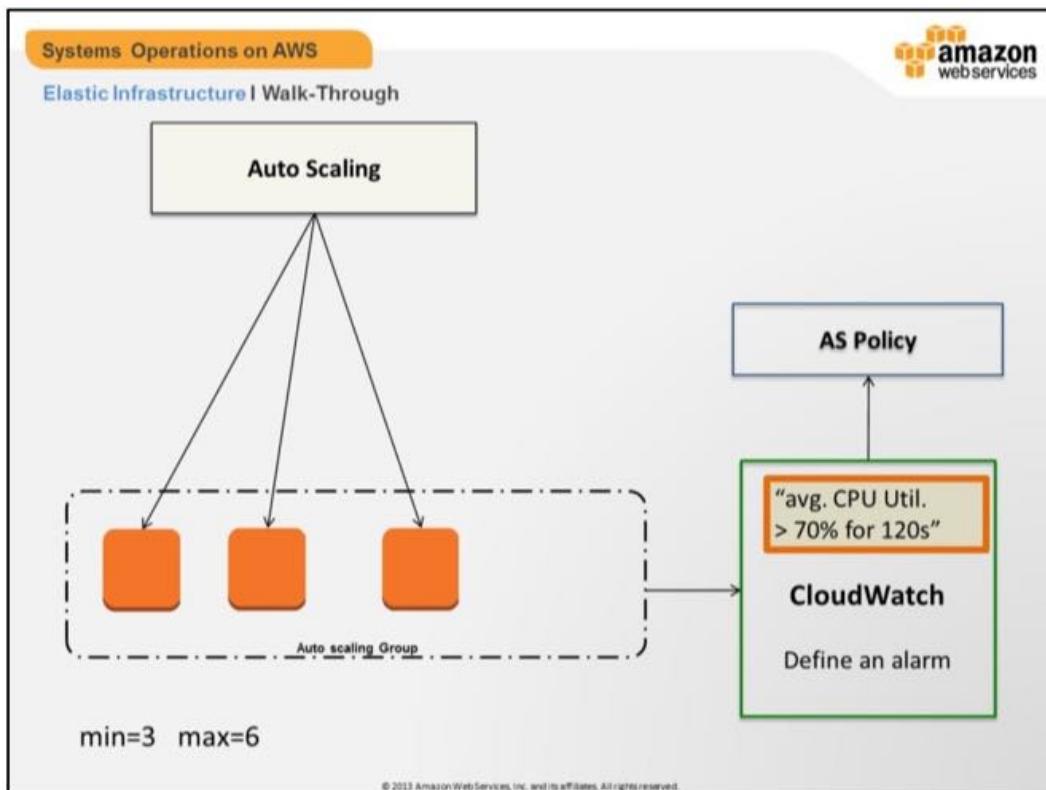
So far we have seen how auto scaling can maintain the size of our auto scaling group; we haven't looked at other mechanisms of triggering auto scaling events yet, that's coming up!



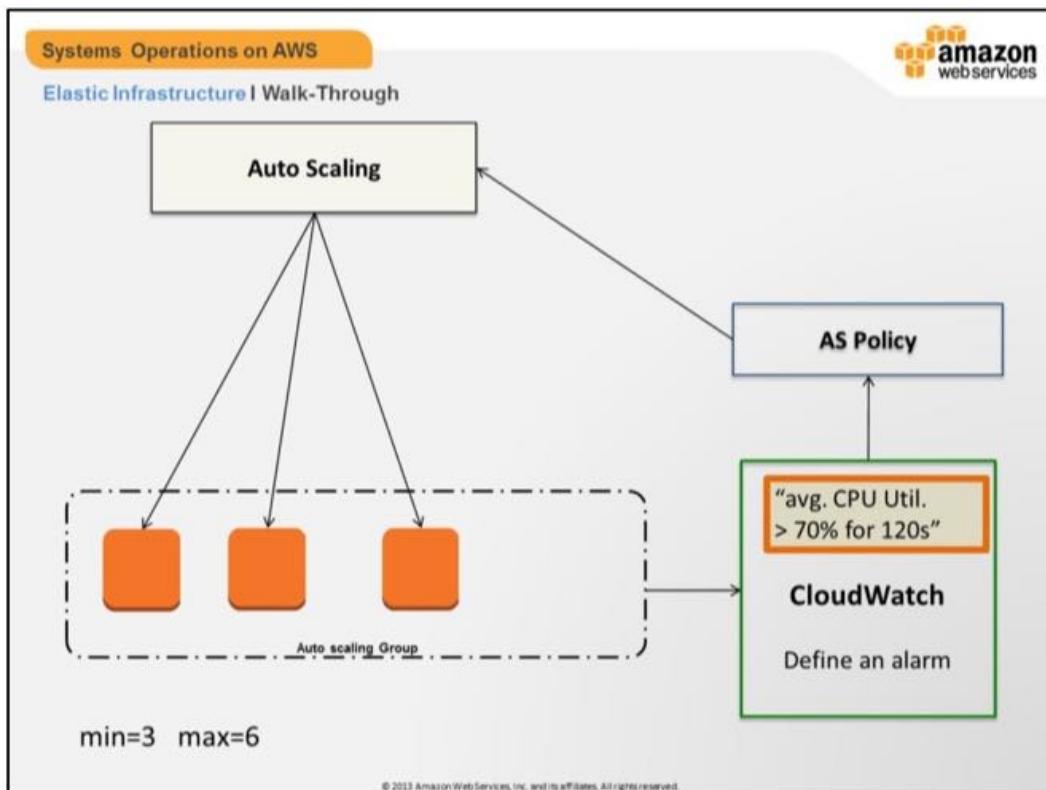
Bring on CloudWatch; auto scaling provides some CloudWatch metrics that we can use as triggers for auto scaling. In fact, you could trigger on any CloudWatch metric.



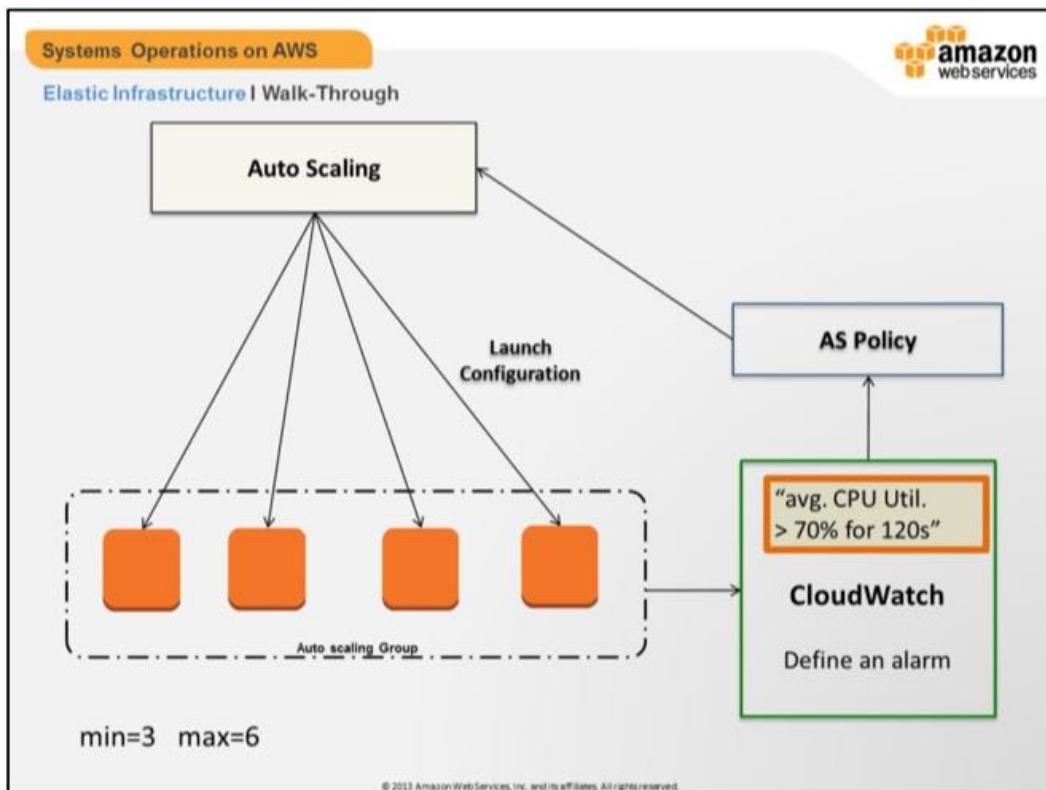
We'll define a CloudWatch alarm that will fire when the average CPU utilization across the auto scaling group is greater than 70% utilization for 120x seconds.



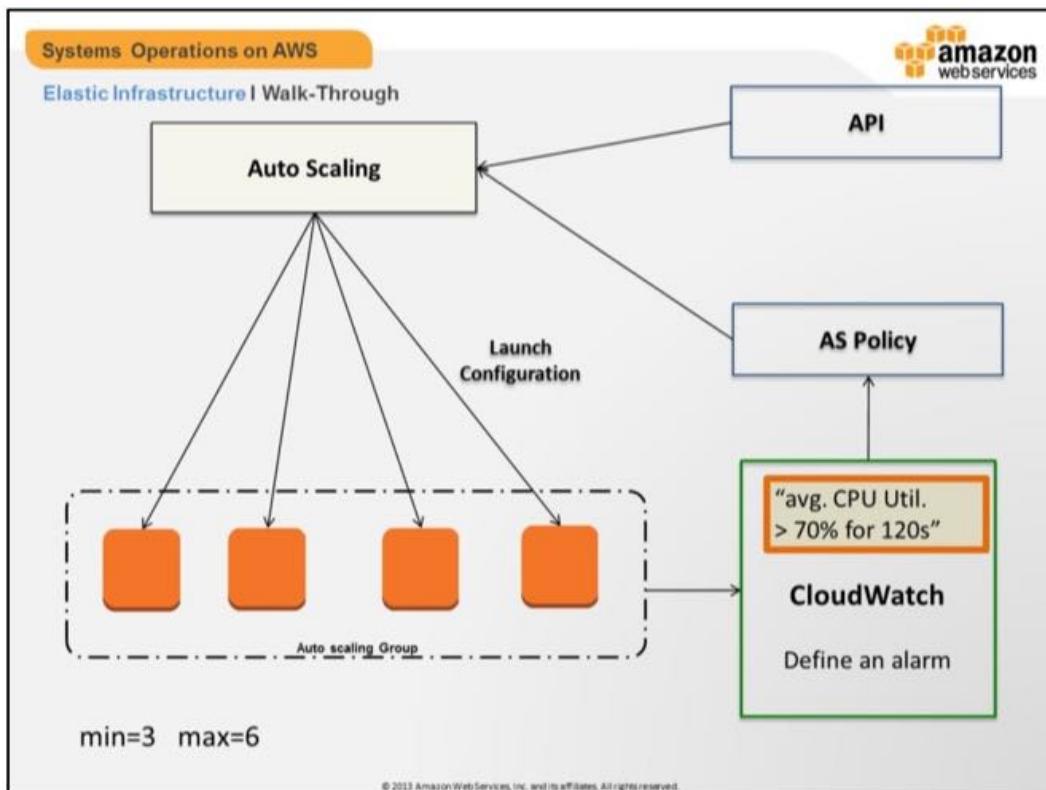
The alarm action will be to execute an auto scaling policy. Remember that we looked at auto scaling policies when we were discussing the various components of auto scaling earlier on in the presentation.



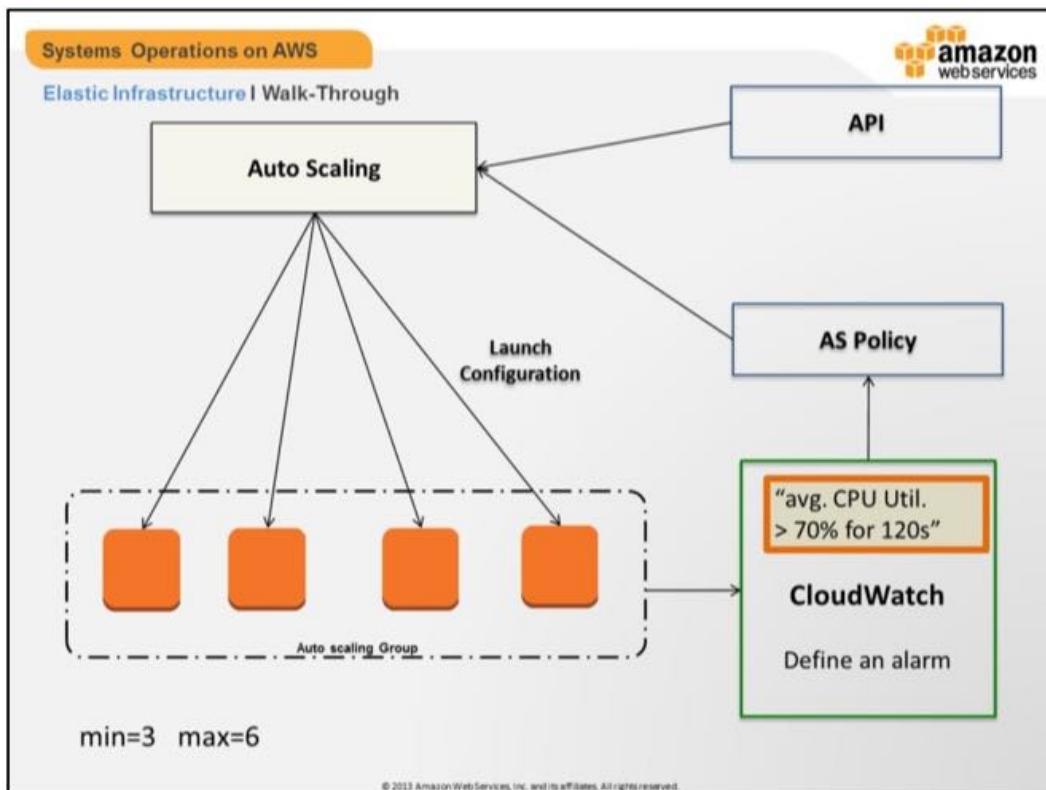
Auto scaling will kick in and ramp up the capacity in the auto scaling group to handle the increase in CPU utilization.



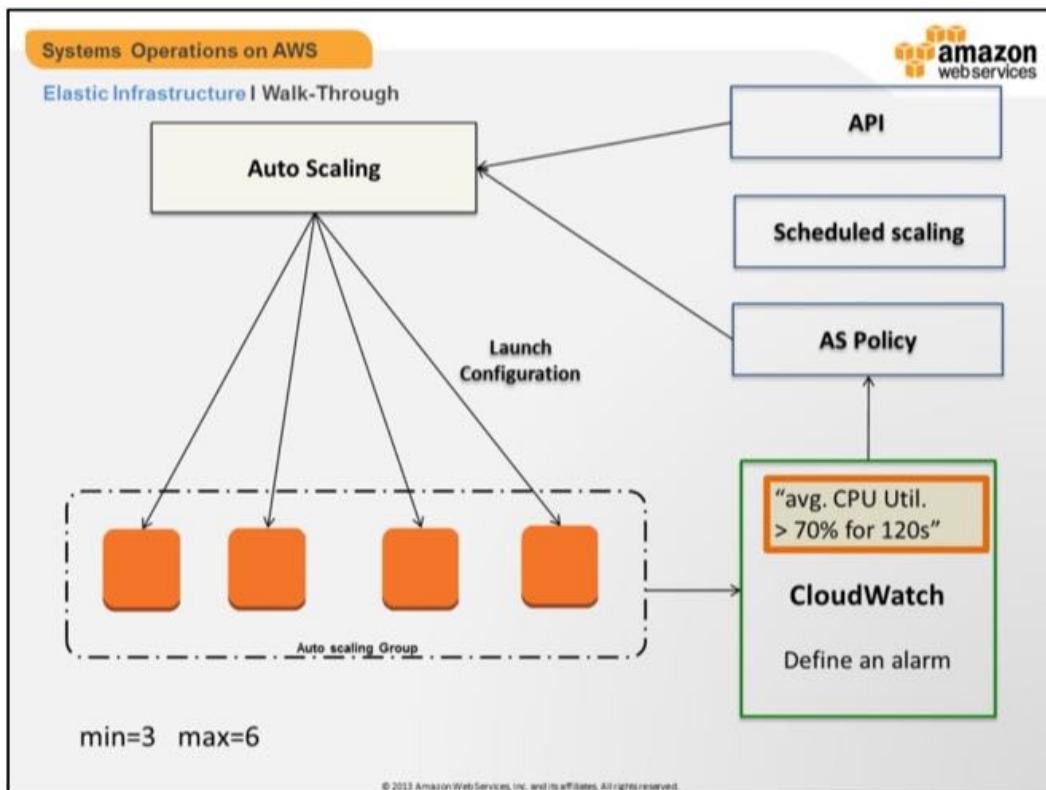
Another EC2 instance will be provisioned as per the launch configuration associated with the auto scaling group.



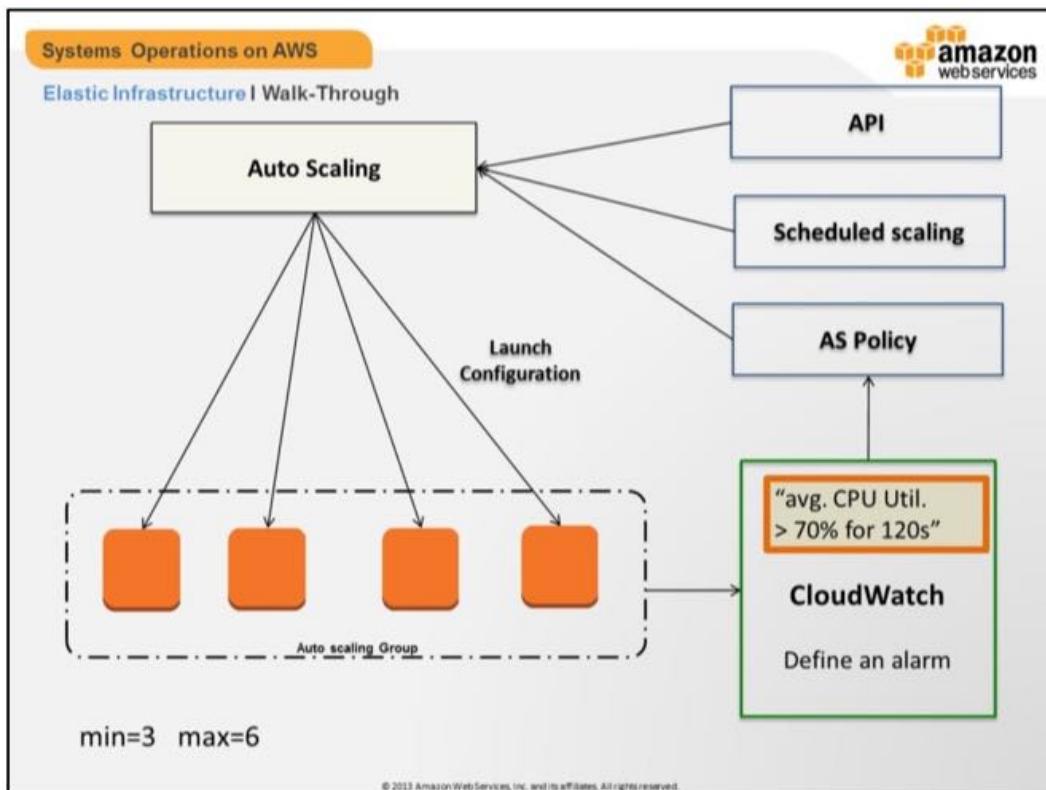
The API is another way to trigger an auto scaling event; essentially the “push-button” approach to auto scaling!



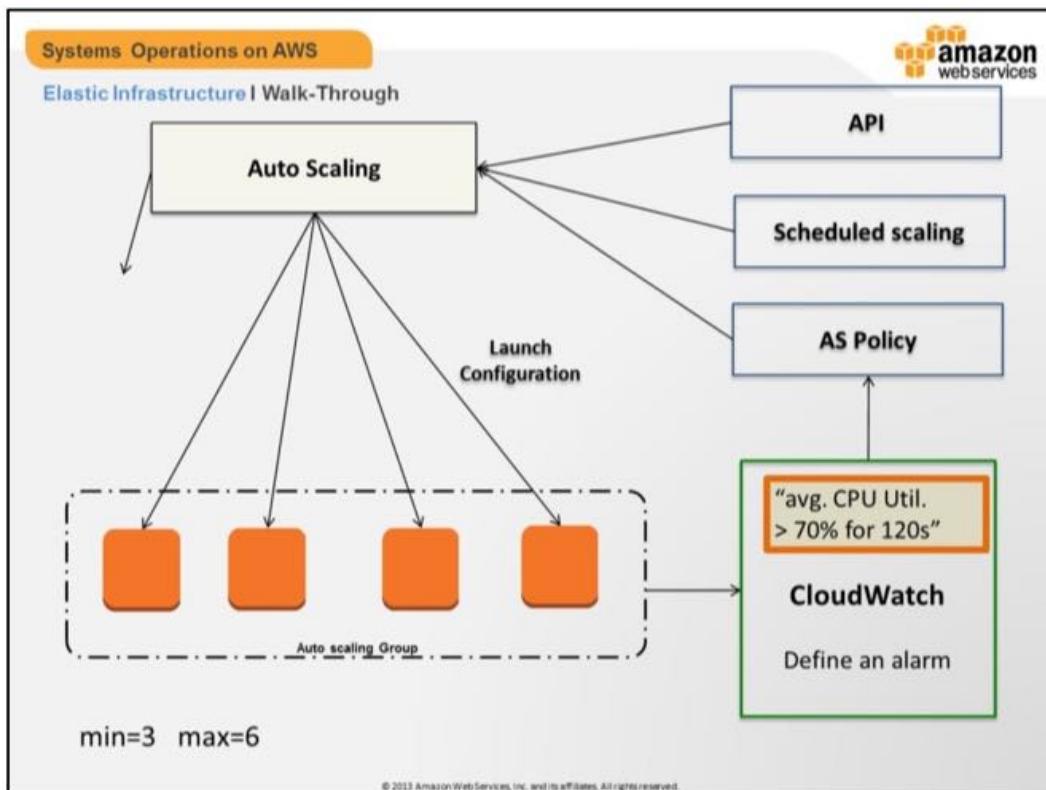
The API call (via CLI, SDK etc.) can tell the auto scaling control plane what to do, i.e. scale up or scale down.



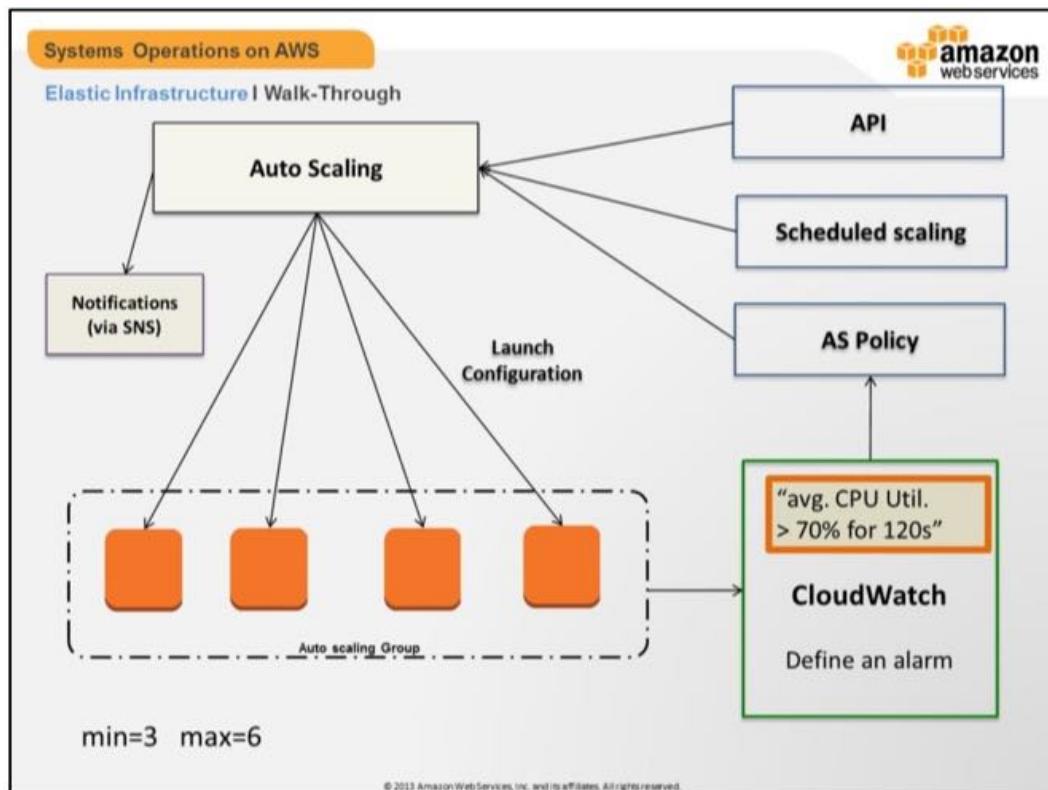
If you have predictable spikes in traffic, i.e. you know when the servers are more utilized you can leverage “scheduled scaling”, which is triggering an auto scaling event to occur at a specific time.



The auto scaling control plane will handle the scheduled trigger.



If you want to be notified whenever auto scaling events have occurred or are occurring; i.e. we're scaling up or down then we can integrate with SNS.



You specify an SNS topic and the auto scaling events that auto scaling should send notifications to. You can subscribe to receive e-mails or even drive orchestration activities based on the notification.

Systems Operations on AWS

Elastic Infrastructure | Summary

amazon
webservices

- What is Auto Scaling?
- Use cases for Auto Scaling
- Components of Auto Scaling
- Auto Scaling and Amazon CloudWatch
- Auto Scaling and Amazon SNS
- Auto Scaling and Elastic Load Balancers
- Scheduled Auto Scaling
- Auto Scaling with Spot Instances

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

In this module we discussed auto scaling, what it is, some use cases and the various components that constitute it.

Systems Operations on AWS

Elastic Infrastructure I Lab Exercise

Lab Objectives

- The following topics are covered in these lab instructions:
- Amazon EC2 – Amazon Machine Image for deployment
 - Using EC2 Instance UserData
 - Using a script with EC2 Instance UserData
- Elastic Load Balancing
 - Creating an Elastic Load Balancer
 - Configuring a Health Check
- Amazon EC2 – Auto Scaling
 - Auto Scaling Launch Configurations
 - Auto Scaling Groups
 - Auto Scaling Policies
- Cloud Watch
 - Cloud Watch Alarms for Auto Scaling.



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The following topics are covered in the Elastic Infrastructure lab:

- Amazon EC2 – Amazon Machine Image for deployment
 - Using EC2 Instance UserData
 - Using a script with EC2 Instance UserData
- Elastic Load Balancing
 - Creating an Elastic Load Balancer
 - Configuring a Health Check
- Amazon EC2 – Auto Scaling
 - Auto Scaling Launch Configurations
 - Auto Scaling Groups
 - Auto Scaling Policies
- Cloud Watch
 - Cloud Watch Alarms for Auto Scaling.



Lab

Creating an Elastic Infrastructure

Amazon EC2 Auto Scaling

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Elastic Infrastructure I Lab Exercise

Lab Objectives

- The following topics are covered in these lab instructions:
- Amazon EC2 – Amazon Machine Image for deployment
 - Using EC2 Instance UserData
 - Using a script with EC2 Instance UserData
- Elastic Load Balancing
 - Creating an Elastic Load Balancer
 - Configuring a Health Check
- Amazon EC2 – Auto Scaling
 - Auto Scaling Launch Configurations
 - Auto Scaling Groups
 - Auto Scaling Policies
- Cloud Watch
 - Cloud Watch Alarms for Auto Scaling.



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

The following topics are covered in the Elastic Infrastructure lab:

- Amazon EC2 – Amazon Machine Image for deployment
 - Using EC2 Instance UserData
 - Using a script with EC2 Instance UserData
- Elastic Load Balancing
 - Creating an Elastic Load Balancer
 - Configuring a Health Check
- Amazon EC2 – Auto Scaling
 - Auto Scaling Launch Configurations
 - Auto Scaling Groups
 - Auto Scaling Policies
- Cloud Watch
 - Cloud Watch Alarms for Auto Scaling.



Module 12: Cost Control

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Cost Control I Overview

What We Will Cover

- Use-cases
- Cost allocation vs. cost optimization
- Prerequisites of cost control
- Billing dashboard
- Cost allocation and billing reports
- Correlating cost allocation with utilization/performance
- Interpreting outputs for efficient cost control
- Billing alarms
- AWS Trusted Advisor



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- **Use-cases** - we'll start off with some blanket use cases or scenarios that will be helpful in setting the stage for this module
- **Cost allocation vs. cost optimization** – what is the difference between the two?
- **Prerequisites of cost control** – some things need to be done prior to getting allocation data and subsequently the reports; customers also need to determine their own use cases to identify how that want data presented
- **Billing dashboard** – gives a high-level overview of spend-to-date as well as a nice breakdown of where costs have been incurred
- **Cost allocation report** – AWS provide this; but only after it's been set up; this is why the *Prerequisites* are important!
- **Correlating cost allocation with utilization/performance** – so we know where our spend has been; but could we have lowered costs? Should instances have been terminated if they were underutilized? CloudWatch metrics can help!
- **Interpreting outputs for efficient cost control** – the reports will allow adjustments to be made to the AWS infrastructure in terms of efficient utilization to minimize/reduce costs – we'll discuss this at the end

Systems Operations on AWS
Cost Control I Overview
Use Cases

- Minimizing IT spend
 - Budget constraints
- Cost allocation
 - Who pays for what?
- Project & portfolio management
 - Tracking ongoing costs
 - Realizing ROI



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Moving everything into the cloud doesn't mean forgetting about it. It's all very well moving to an OPEX model; but enterprises and larger organizations still need to be just as critical as to what their IT infrastructure is costing them. "It will be cheaper!", or will it? Without effective cost control measures this question cannot be answered. Cost metrics are some of the key indicators that organizations will use to make decisions about IT spend.

- Minimizing IT spend – budgets will dictate how much money can be spent on IT; if we don't know how much we're spending, then how do we know if we are over or under our allocation? Sure; a monthly figure tells us how much we spent overall, but we need more granularity in order to optimize the allocation.
- Apportioning allocation – Large organizations with multiple internal teams may be leveraging one or more AWS accounts, they may be sharing resources within the cloud, some of them could be using more than others;

how do we accurately apportion the charges; who will pay for what?

- Project & portfolio management – Managing a project effectively means being able to identify exactly how much is being, or has been spent at any one time. Often projects will realize ROI opportunities during different phases of the project lifecycle (usually measured with financial metrics such as NPV; Net Present Value); if a project has implemented some IT asset, we need to be able to track that spend and measure it against what that asset may be making, saving etc. in terms of revenue

Systems Operations on AWS

Cost Control I Allocation vs. Optimization

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- Allocation => what cost what
 - Identify how resources are charged
- Optimization => efficiency
 - Smarter spending

Service	Allocation (%)
EC2	~15%
RDS	~25%
Redshift	~45%
S3	~10%

Cost allocation is the process of assigning costs/charges to individual AWS services for an account (with Detailing Billing Reports granularity goes to an individual resource within a service!). Cost allocation assists the customer to identify where money is actually being spent, e.g. EC2 vs RDS vs S3.

Cost optimization is actually the process of interpreting the cost allocation (along with other metrics such as performance) and making decisions as to how best to efficiently manage those costs on an ongoing basis. This may be terminating instances that are turned on but effectively idle or perhaps vertically down-scaling RDS database instances that are under-utilized. This is why it's a good idea to have another source of metrics to assist in this decision making process, e.g. allocation vs performance; without having insight to which instances may be idle, you can't make the decision to potentially turn them off.

***** NOTE:** Trusted Advisor is another mechanism to assist in cost

optimization, we'll talk briefly about this at the end of the deck. Other partners are also assisting with this effort.

Cost control is the over-arching term for both allocation and optimization.

Systems Operations on AWS
Cost Control I Overview
Prerequisites of Cost Control



- Several cost billing constructs to choose from
- Know the organizational use cases
- Enable programmatic access
- Sign up for consolidated billing

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Several options are available to the customer for reporting costs, these will be mentioned in the next slide, but for brevity:

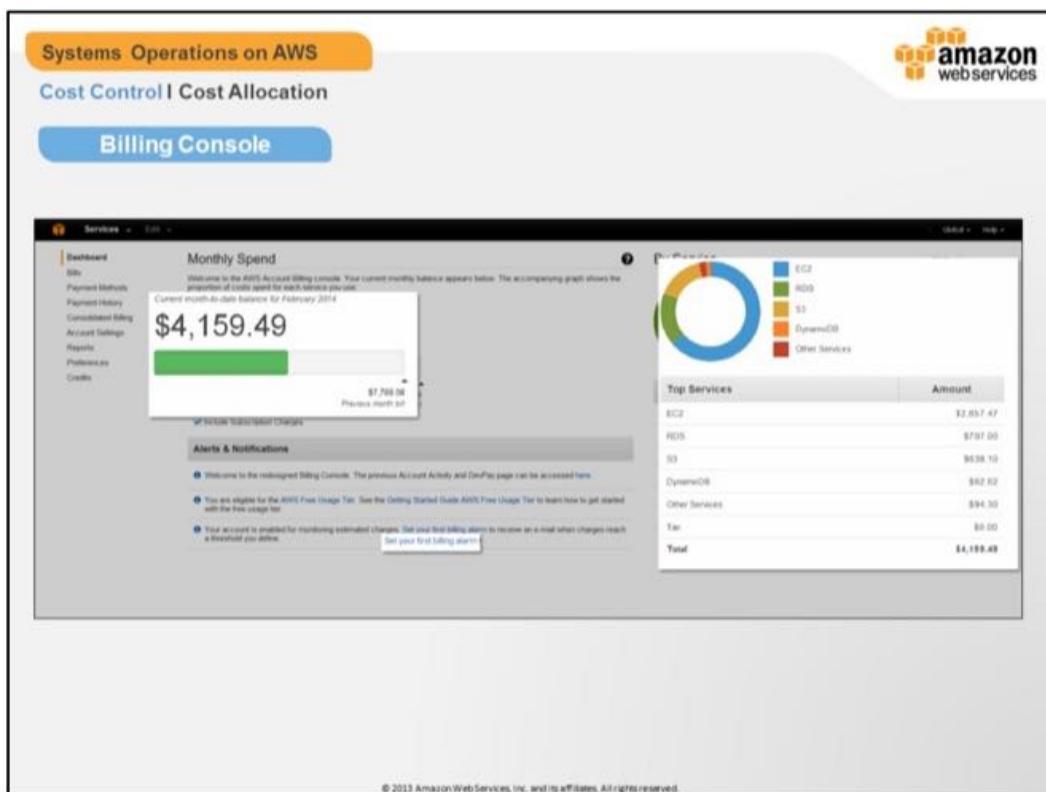
- Monthly Report - aggregated charges month to date by AWS account and product name (e.g. EC2, RDS etc.)
- Cost Allocation Report – costs are aggregated tags defined by the customer
- Detailed Billing Report – granular hourly based report aggregated by AWS account and product name
- Detailed Billing Report (with resources and tags) – granular hourly based report by resource id and tags

Know the organization use cases: is detailed billing really required? Perhaps a cost allocation report would be adequate? Identify how cost data should be visualized, determine if other departments need to be charged-back for AWS utilization and so on. Answering these kinds of questions will help the customer identify the type of cost reporting to be enabled.

Enable programmatic access: For anything other than the Monthly Report, programmatic access needs to be enabled for the account. This essentially is creating an (or use an existing) S3 bucket and giving access to AWS to create a csv (or in the case of Detailed Billing Reports a zip) file in that bucket.

Sign up for consolidated billing: Lastly, large organizations typically have multiple AWS accounts; consolidated billing allows these accounts to be consolidated to a single bill.

single account for billing purposes.



The billing console is a great way to get; at a glance; a current breakdown of your AWS spend. You can easily see what your spend has been to date for the past month as well as a nice breakdown by service. Let's discuss briefly:

- (1. We can see for the month of February that we have incurred charges of ~\$3100; the previous month's bill was ~\$7788. If the current month's spend is greater than the previous months (or looks like it's heading in that direction) – this may be a hint to investigate how you are provisioning AWS resources. Then again; maybe your usage patterns have changed and the spend is accurate. We'll see later how you can analyze the Detailed Billing Reports to get more insight.
- (2. The service breakdown clearly shows us that the majority of the bill is actually incurred by EC2 resources followed by RDS and S3 storage charges.
- (3. Also you are able to quickly enable billing alarms from the Billing Console – we discuss billing alarms in more detail later in this module.

Systems Operations on AWS

Cost Control | Cost Allocation and Billing Reports

Report Types

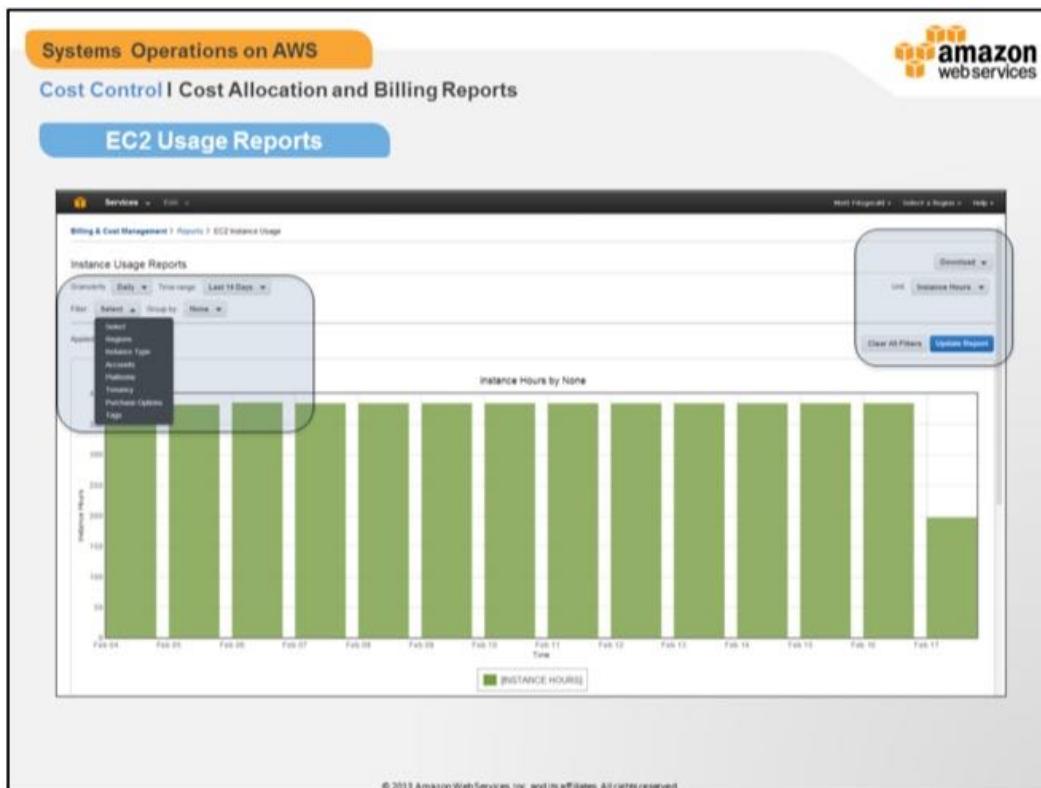
- EC2 Usage Reports (interactive reports)
 - Instance Usage Report
 - Reserved Instance Utilization Report
- AWS Usage Report
 - CSV based report on usage of specific AWS services
- Monthly report
- Cost allocation report
- Detailed billing report
- Detailed billing report with resources and tags

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

EC2 usage reports are interactive; you can visualize them directly in the billing console; the next slide will show an example of what one of these reports looks like and the options available for presentation.

The following reports need to be enabled (as shown in the previous slide)

- **Monthly report** is a CSV report that gives AWS usage charges for the month
- **Cost allocation report** provides a report based on utilization breakdown using tags as a reconciliation point – we'll discuss tags with billing shortly; they're also used with the detailed billing reports
- **Detailed billing report** is a breakdown of AWS service charges aggregated by service type in hourly granularity
- **Detailed billing report with resources and tags** is a breakdown of AWS service charges aggregated by individual resources (or tagged resources) in hourly granularity



Here's an example of an *EC2 usage report*. You can visualize EC2 costs or instance hours over time for up to 3 years at monthly granularity (shorter periods can use daily or hourly granularity), in addition to setting options to *filter* and *group* on region, instance type, tags and so on. This is a great tool to leverage to see exactly how your EC2 spend and utilization is going. Lastly, you also have the option of downloading the graph or CSV containing the data that the graph is generated from. These reports do not need to be turned on; you can run them at any time!

The screenshot shows the AWS Billing Reports interface. At the top, there are tabs for 'Systems Operations on AWS' and 'Cost Control | Cost Allocation'. Below that, a 'Billing Reports' button is highlighted. To the right is the Amazon Web Services logo. A large blue arrow points downwards from the 'Receive Billing Reports' button towards a detailed view of the 'Preferences' screen.

Billing Reports

- Enable programmatic access
- CSV formatted files
- Visualize in Excel or other tool

Receive Billing Reports

Preferences

Receive PDF Invoice By Email

Receive Billing Alerts

Receive Billing Reports

Save to S3 Bucket: Verify

Report

Monthly report
Detailed billing report
Cost allocation report
Detailed Billing Report with resources and tags

* Hosted by S3 Usage Metrics

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

- Programmatic access, as discussed in the prerequisites needs to be enabled in order to turn on detailed billing and cost allocation reports. In order to do this, an S3 bucket needs to be created with a special policy allowing AWS to write to it (it can be a pre-existing bucket if desired).
- Billing files will be dumped into your designated S3 bucket by AWS, from here you can download them directly from S3, interactively or via API calls.
- Once you have the data you can load it into Excel or some other visualization tool to generate nice reports and charts.

The screenshot shows the 'IAM Configuration' section of the AWS Management Console. It includes a list of best practices:

- Best practice
- Using the root/owner account:
 - Enable IAM user access to the AWS Website
 - Complete Configure Security Challenge Questions
- Give permission to an IAM user
 - E.g. AWS Account Activity Access policy template
- Access account activity information with IAM user

Below the list are two screenshots of the AWS website:

- IAM user access to the AWS Website:** A message stating that IAM user access to the AWS Website enables IAM users with appropriate permissions configured to access the Account Activity and Usage Reports pages. It includes a link to 'Using IAM'.
- Activate the following pages:** A section where users can activate 'Account Activity Page' and 'Usage Report'. There is also a 'Deactivate Now' button.
- Configure Security Challenge Questions:** A section explaining how security challenge questions help identify the user. It includes fields for 'Question' and 'Answer' for three questions: childhood nickname, first live concert, and college name.
- My Account -> Manage Your Account:** A link to manage account preferences.
- My Account -> Personal Information:** A link to manage personal information.

At the bottom of the screenshot, it says: © 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

One of AWS' best practices is to use the root/owner account as little as possible. We can do most things related to Account Activity such as *usage reports, billing alerts and preferences* as a separate privileged IAM user.

1. The first thing to do is (as the root account) ensure that "IAM user access to the AWS Website" is enabled from *Manage Your Account*. Even if the user has an explicit permission allowing them access to Account Activity they won't get very far if this option hasn't been enabled.
2. Also ensure that the root/owner account has the security challenge questions set from *Personal Information*; this is also another requirement before IAM users can access the Account Activity section of the AWS Management Console.
3. Create a new user (or give access to an existing user) and then assign the appropriate permission that gives the user access to Account Activity (the AWS Account Activity Access policy template is a good place to start!)

Once the three items above have been completed you should use that IAM user to access Account Activity information going forward.

***** Note:** root/owner accounts are the only accounts that can configure consolidated billing.

Systems Operations on AWS

Cost Control I Cost Allocation and Billing Reports

Tags

Account

- Account Activity
- AWS Identity and Access Management
- AWS Management Console
- Consolidated Billing
- DevPay
- Manage Your Account
- Payment Method
- Personal Information
- Security Credentials
- Usage Reports
- Billing Alerts
- Billing Preferences

Cost Allocation Report

- **Manage Cost Allocation Report**

Manage Cost Allocation Report

The following list displays tag keys that you have created for your AWS resources. These tag keys can be used to organize your AWS costs.

Select the check box next to each tag key that you want to include in your Cost Allocation Report, or clear the check box for each tag key you want to exclude. You can filter the list by selecting **Yes** or **No** under the **Included** drop-down menu. Select **Yes** to display the tag keys currently included in the report. Select **No** to display the tag keys currently excluded from the report.

By default, tags that you add using the API or AWS Management Console for each service are automatically excluded from the Cost Allocation Report. Add them to the report by selecting them from the following list.

For more information, see About Cost Allocation.

Key	Included
Application	<input checked="" type="checkbox"/>
aws:autoscaling:groupName	<input type="checkbox"/>
aws:cloudformation:logical-id	<input type="checkbox"/>
aws:cloudformation:stack-Id	<input type="checkbox"/>
aws:cloudformation:stack-name	<input type="checkbox"/>
Cost Center	<input type="checkbox"/>
Department	<input type="checkbox"/>

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Cost Allocation and Detailed Billing Reports can leverage resource tagging. That is to say each line item will be attributed to any tags that may have been defined (if the resource supports tagging). For example; a project may have tagged EC2 instances with a project name; this project name tag can be called out in the billing reports.

By default only the ‘Name’ tag is represented; to include more tags you must “enable” them via the “Manage Cost Allocation Report” link from the “Cost Allocation Report” section. You will see a consolidated list of all tags that can be turned on/off using the corresponding check—box. The settings will be saved automatically upon change.

Systems Operations on AWS

Cost Control | Cost Optimization

Correlating Cost & Performance

- Assists in cost optimization
- Performance metrics from:
 - CloudWatch
 - Ganglia
 - Other enterprise monitoring software

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Assists in cost optimization:

It's hard to make decisions to adjust expenditure without being able to see what (or what doesn't) impacts or correlates with it. Pulling in performance/utilization statistics for various system and application metrics is a good way of assisting you to determine whether expenditure is inline with operational performance.

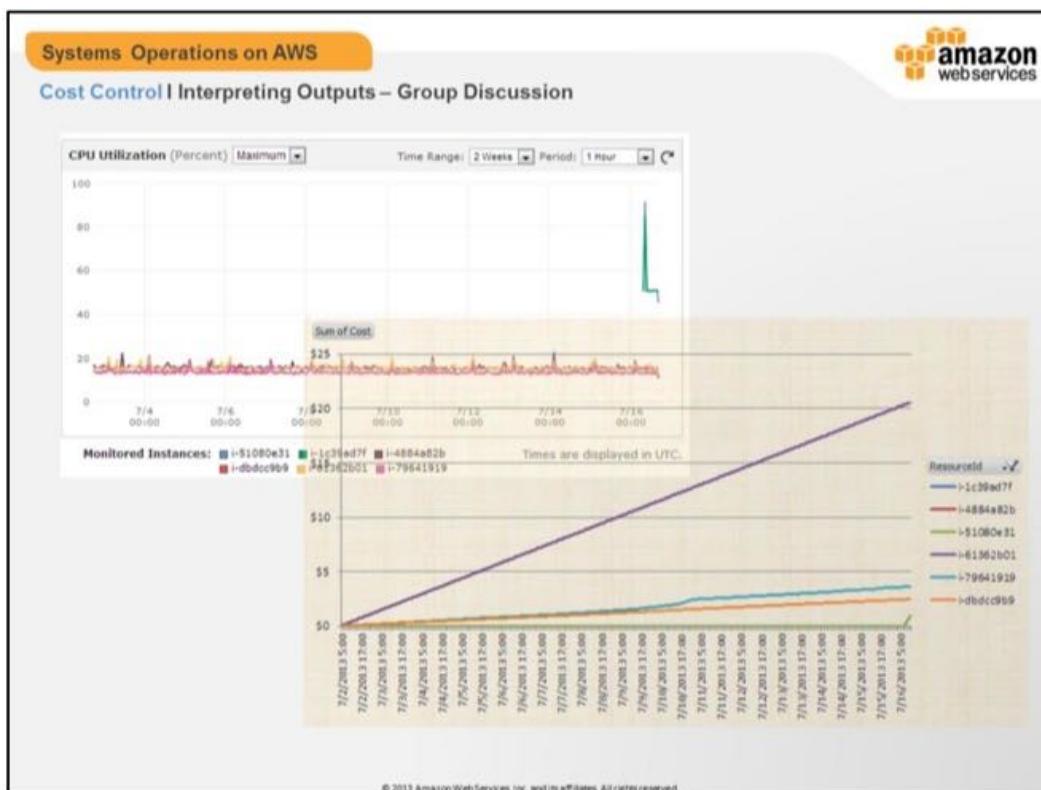
For example; consider AWS RDS metrics – if you're paying a lot for RDS instances; consider the following RDS CloudWatch metrics:

- Read/write IOPS – are these consistently low?
- CPU utilization – also low and essentially idle?
- DB Connections – low, or no connections?

If you notice some of these things, you might like to re-consider the way the database tier is used, or whether it is used at all!

Performance metrics:

CloudWatch is not the only place to get performance metrics, you can leverage data pulled from other tools such as IBM Tivoli, HP SiteScope or BPM (Business Process Monitor) and Ganglia.



Here we have two charts (*ok, so the data is not so good here, however; we can revise as we move forward!!!*):

Top: this depicts the CPU performance of our media transcoder instances over a two week period

Bottom: here we have the running total of costs for the same media transcode instances

This isn't the most accurate way of doing cost optimization because the charts were generated in two different places and we have to visually observe the relationship; the performance chart was taken directly from the AWS Management Console, and the second one was created as a Pivot Chart in MS Excel; they're both "over-time" reports which are useful for trend analysis. The transparency/overlay – is really to emphasize the point that the two graphs need to be viewed together, or on top of each other.

We can look at both side by side and make some conclusions:

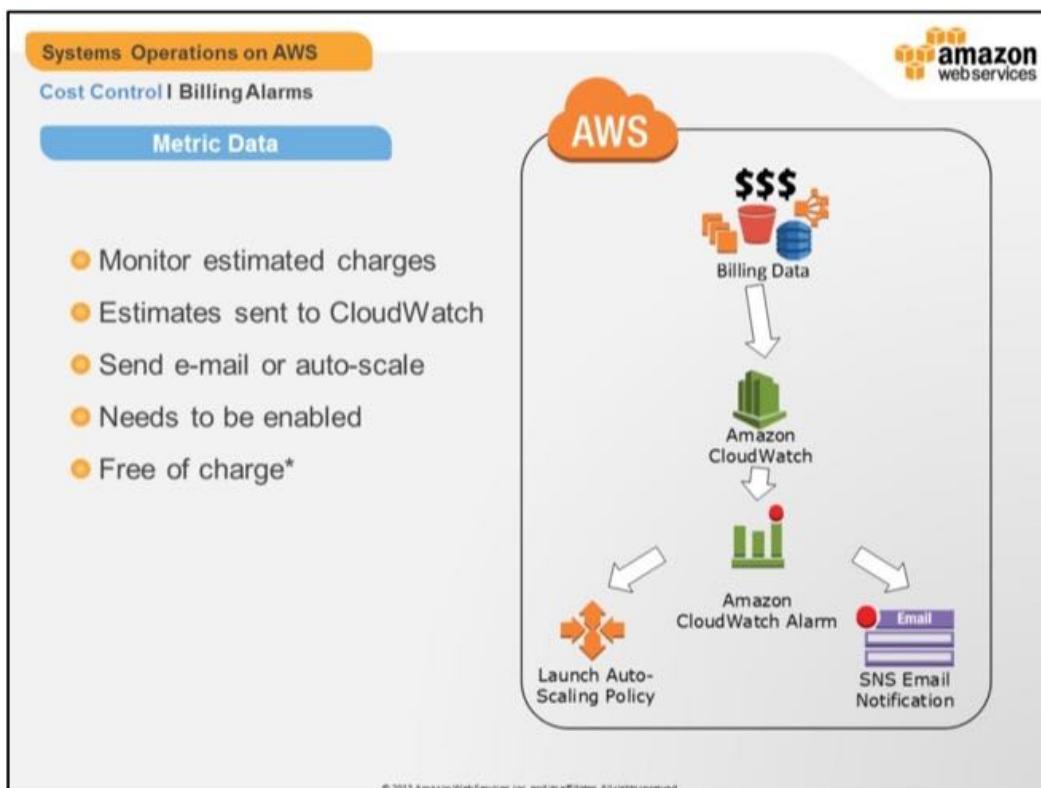
* One instance is costing significantly more, why? (It's a large instance type of m1.large, the others are m1.small; however some of the m1.smalls are spot instances)

* Cost are increasingly almost linearly over the two week period yet, performance of the CPU is idle; what can we do to optimize costs? (Reduce the

size of the instances perhaps?) We could conclude that we terminate or stop the instances, yet the processes of the box may not be CPU bound but IO bound. Maybe we need to bring other performance metrics into the picture?

Comparing just one performance metric again cost occurrence is not ideal; many performance metrics need to be considered when evaluating cost optimization. Ideally, pull the performance metric raw-data into the same charting tool and leverage stacked-line charts.

**** NOTE **** - The costs here are a running total of ALL costs associated with an EC2 instance, i.e. the hourly charge of the instance plus any charges relating to data in/out. Play around with how you measure costs, there is a ***UsageType*** column in the detailed billing reports that can assist in this! Also, don't just look at the EC2 costs, consider other AWS resources that may be in use such as RDS database instances, EBS volumes, SQS queues, SNS topics and so on.



One mechanism to assist in cost control is billing alarms.

When this feature is enabled, estimated charges for all AWS resources are sent to CloudWatch as metric data. These metrics can then be alerted upon when a resource's estimated charge is greater than a predefined threshold. An action can then be carried out either in the form of an SNS e-mail notification or as a trigger for auto scaling activity.

Once the monitoring for estimated charges has been enabled, it cannot be disabled. However, billing alarms can be created and deleted as desired.

* It's also important to note that there is no charge for enabling this feature. Also, keep in mind that you get 10 CloudWatch and 1000 SNS email notifications per month for free – if you exceed these then.

Systems Operations on AWS
Cost Control | Cost Optimization
AWS Trusted Advisor

Presented as annual savings with optimal usage of:

- EC2 reserved and on-demand instances
- Elastic Load Balancers (ELBs)
- Elastic Block Store (EBS) volumes
- Elastic IP (EIP) addresses
- RDS instances



The screenshot shows the AWS Trusted Advisor dashboard with a focus on Cost Optimizing. It displays five circular performance indicators: Cost Optimizing (Excellent), Security (Good), Fault Tolerance (Good), and Performance (Good). Below the indicators, there is a section titled 'Recently Launched Checks' with several items listed, each with a status (e.g., 'New') and a brief description.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Taken from the AWS Trusted Advisor web page:

"AWS Trusted Advisor draws upon best practices learned from AWS' aggregated operational history of serving hundreds of thousands of AWS customers. The AWS Trusted Advisor inspects your AWS environment and makes recommendations when opportunities exist to save money, improve system performance, or close security gaps."

This slide is not intended to highlight all the features of AWS Trusted Advisor, but to point out that there is a "Cost Optimizing" component to it, and that it's worth mentioning whenever discussing cost optimization in the context of AWS. Areas of focus specifically are:

- Optimal reserved instance usage
- Under-utilized EC2 instances
- Idle ELBs
- Under-utilized EBS volumes
- Unused EIPs
- Idle RDS instances

***** NOTE:** Trusted Advisor is available for customers on Business and Enterprise-level support.

Systems Operations on AWS

Cost Control Group Activity



Group Activity

Analyzing a Detailed Billing Report

Download Spreadsheet with [Sample Billing Data](#)

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Group Activity: Detailed Billing Report Spreadsheet

Systems Operations on AWS

Cost Control I Group Activity

Analyzing a Detailed Billing Report

This activity, form into pairs or small groups. You will:

- Load some pre-existing billing data files into Excel
- Answer the following:
 - a. How much did all Amazon RDS resources cost?
 - b. What were the Amazon S3 charges for the *mf75billing* bucket?
 - c. What were the Amazon SQS+SNS+SWF costs for the period 7/13 to 7/15 (3 days)?
 - d. How much did the Amazon EC2 instances cost in Ireland?
- Spend 15 minutes generating some pivot charts and tables to visual the data for your own use cases

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.



Activity

A sample Excel spreadsheet has been provided which include pivot charts designed to answer the four sample questions above.

Go through the process of creating these pivot charts following the instructions provided above.

Additionally, using the sample charts you may change the filters (Availability Zone, Product Name, Resource Id, Date/Time, etc.) to explore other aspects of the example billing info.

Answers: (a. ~\$830 (b. \$0.0019 (c. \$0.27 (d. \$19.37

What have we just done? We've created a line chart over the time period of the billing data that represents the rate of increase in costs for each AWS product that was utilized.

Systems Operations on AWS

Cost Control I Summary

amazon
webservices

- Cost allocation vs. optimization
- Various billing / cost reporting options
- Enable programmatic access for detailed billing
- Discussed correlating billing with performance metrics
- Visualizing data
- Optimization
- Billing alarms
- AWS Trusted Advisor



© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

In this module we discussed various items related to cost control/allocation and optimization. AWS provides several billing constructs that we discussed; some of these require that “Programmatic Access” is enabled to ensure we get a billing report delivered to our S3 bucket.

We also talked about cost optimization and efficient spending. It’s difficult to refine our spend patterns when we don’t have anything to measure them against. Visualizing cost data alongside performance/utilization metrics assists us in making decisions around cutting costs or utilizing what we’re paying for more effectively, either by eliminating some AWS resources, or by re-architecting our applications so they utilize the AWS resources in a better way.



Course Summary

© 2013, 2014 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Systems Operations on AWS

Course Summary | Course Objectives

By the end of this course, you should be able to:

- Use Amazon EC2 features to provision, monitor, scale and distribute compute infrastructure.
- Create Amazon Virtual Private Cloud (VPC) resources such as subnets, network access control lists, and security groups.
- Backup AWS and on-premise resources using AWS services.
- Use Amazon CloudWatch metrics to monitor the health and utilization of AWS resources.
- Leverage resource tagging to allocate costs and optimize resource planning.
- Create a gold image and employ auto scaling into the VPC.

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Learning Objectives

At the end of this course, you should be able to:

- Use Amazon EC2 features to provision, monitor, scale and distribute compute infrastructure
- Create Amazon Virtual Private Cloud (VPC) resources such as subnets, network access control lists, and security groups
- Backup AWS and on-premise resources using AWS services
- Use Amazon CloudWatch metrics to monitor the health and utilization of AWS resources
- Leverage resource tagging to allocate costs and optimize resource planning
- Create a gold image and employ auto scaling into the VPC.

Systems Operations on AWS

Course Summary | AWS Training & Certification

Self-Paced Labs

Try products, gain new skills, and get hands-on practice working with AWS technologies

aws.amazon.com/training/self-paced-labs

Training

Skill up and gain confidence to design, develop, deploy and manage your applications on AWS

aws.amazon.com/training

Certification

Demonstrate your skills, knowledge, and expertise with the AWS platform

aws.amazon.com/certification

© 2013 Amazon Web Services, Inc. and its affiliates. All rights reserved.

Notes:

We have several programs available to help you deepen your knowledge and proficiency with AWS. We encourage you to check out the following resources:

- Get hands-on experience testing products and gaining practical experience working with AWS technologies by taking an AWS Self-Paced Lab at run.qwiklab.com. Available anywhere, anytime, you have freedom to take self-paced labs on-demand and learn at your own pace. AWS self-paced labs were designed by AWS subject matter experts and provide an opportunity to use the AWS console in a variety of pre-designed scenarios and common use cases, giving you hands-on practice in a live AWS environment to help you gain confidence working with AWS. You have flexibility to choose from topics and products about which you want to learn more.
- Take an instructor-led AWS Training course. We have a variety of role-based courses to meet the requirements of your job role and business need, whether you're a Solutions Architect, Developer, SysOps Administrator, or just interested in learning AWS fundamentals.
- AWS Certification validates your skills, knowledge and expertise in working with

AWS services. Earning certification enables you to gain visibility and credibility for your skills.