Global Knowledge®

Expert Reference Series of White Papers

Amazon Web Services: An Overview

Amazon Web Services: An Overview

Rich Morrow, Global Knowledge Instructor, Cloud and Big Data Analyst

Introduction

Getting a clear understanding of what Amazon Web Services (AWS) is and how it can help your business can be a daunting task. The depth and breadth of AWS is significant, comprising over 30 services in dozens of data centers located in nine regions across the globe. They offer computing, storage, networking, deployment, management, and a host of supporting services such as queues and email services.

There's a great chance that AWS has more than a few products to help your company work faster, smarter, and more cost effectively. So, where should you start? In this white paper, I hope to provide a good understanding of AWS, how it works, and how your company can get started.

Some History

Like many successful dot-com era startups, Amazon found itself with an enviable problem at the turn of the century: the scale of their business had grown beyond the capacities of any available pre-packaged software solutions. They had to re-think their entire infrastructure from the ground up, and they had to design all their systems to deal with a new set of requirements that their users had. Amazon set out to ensure this new infrastructure would provide:

- **High Availability**: Via geographical fault tolerance, redundancy, and horizontal linear scale.
- **Auto Scaling**: The ability to dynamically respond to spikes in demand and increase or decrease capacity.
- **Integrated Back-Up and Disaster Recovery**: Operations like recurring database backups and snapshot restores become as simple as checkboxes and buttons on web forms.
- **"Infinite" Scale**: Users should not have to consider ever "outgrowing" a service. Amazon Simple Storage Service (S3) allows users to store an unlimited number of objects; Amazon Elastic Compute Cloud (EC2) lets users spin up thousands of virtual servers.
- **Ease of Use**: The easier they could make their services, the greater the chance that their internal developers would use them. Amazon decided to build modular, API-driven "services" that could be used either independently or with one another.

When Amazon finished building much of their required infrastructure in the early 2000s, they realized they had another enviable problem: extra capacity that they rarely and sporadically used. In 2006, they opened AWS for limited public beta, and although the offerings were only a fraction of what is currently available, the product line became wildly popular. By 2007, they had attracted more than 300,000 users, and in 2008, the "beta" moniker was dropped. Subsequent years have seen rapid releases of new services, feature additions to existing services, and prominent customers (including Pinterest, NASA, and Netflix).

In addition to the previous requirements, Amazon tacked on the following new features to serve external customers:

- **Flexible Pricing**: Nearly every service is provided via consumption-based pricing with no upfront fees. Pay for only what you use, when you use it.
- **Heightened Security**: Stateful firewalls (known as "security groups"), consolidated user account management via AWS Identity and Access Management (IAM), and integrated encryption capabilities for some services like Amazon S3.
- **Proactive Price Cuts**: Because AWS operates with significant scale, they can purchase hardware, software, power, bandwidth, and nearly everything else at much, much lower prices than competitors. Instead of pocketing those savings as profits, AWS has traditionally passed those savings on to customers, reducing prices over 30 times in the last six years.

For these reasons and many more, AWS has become an attractive way for businesses of all sizes to deploy and serve the computing and storage needs of their customers. For some companies, such as Netflix, trusting AWS with all of their computing needs enables them to operate with significantly reduced headcount and a great deal more agility.

Global Architecture

AWS calls their individual data centers "Availability Zones" or "AZs." Each data center has multiple redundant power and bandwidth providers, and AZs are organized into "Regions"—collections of physically close data centers that are interconnected via high-bandwidth, low-latency fiber. Amazon currently has nine Regions (although one is strictly for US government use), and each Region has between two and five AZs.

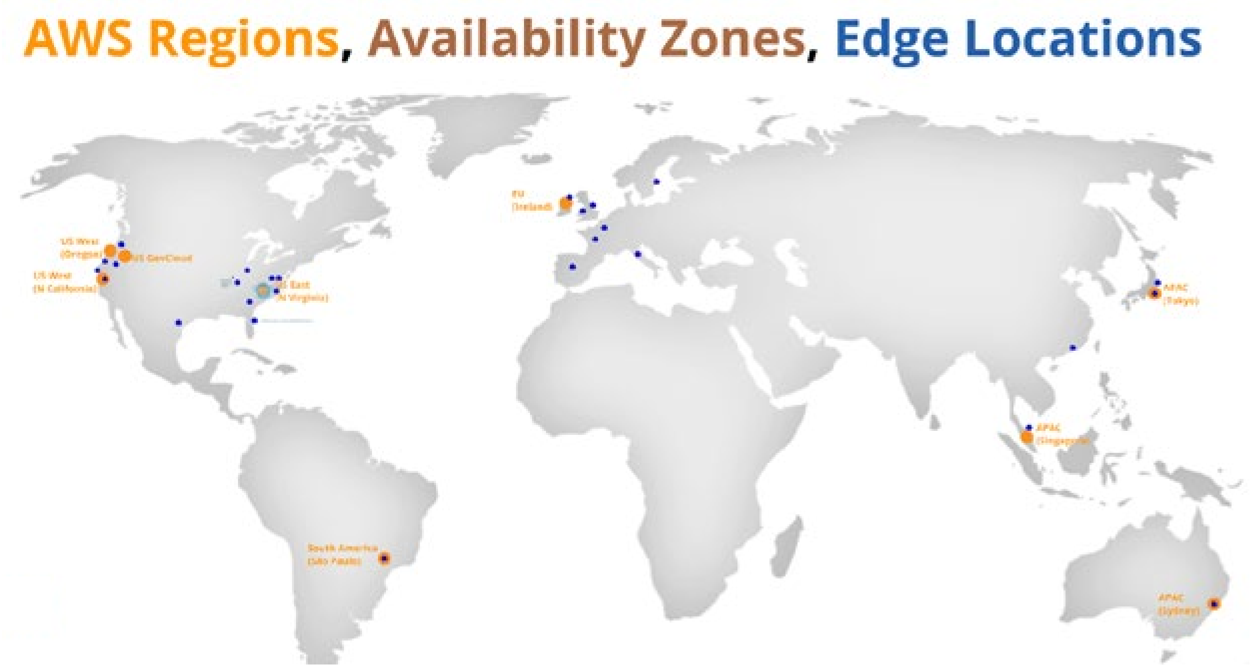

Figure 1. AWS Regions, Availability Zones, and Edge Locations

Most customers will decide to deploy in one Region and use multiple AZs within that region for high availability and disaster recovery. Some services, such as the storage product Amazon S3 (discussed later), deploy to a specific Region, using multiple servers at multiple AZs to allow stored objects to be both incredibly redundant and fault tolerant. Other services, like the virtual server product Amazon EC2, deploy to only one AZ (you can only launch a virtual server into a single data center). However, by using Auto Scaling groups, one could launch a few servers in two or more AZs and then balance across multiple AZs within the region to achieve redundancy and fault tolerance.

These "multi-AZ" deployments are the preferred way to serve web and mobile traffic within AWS.

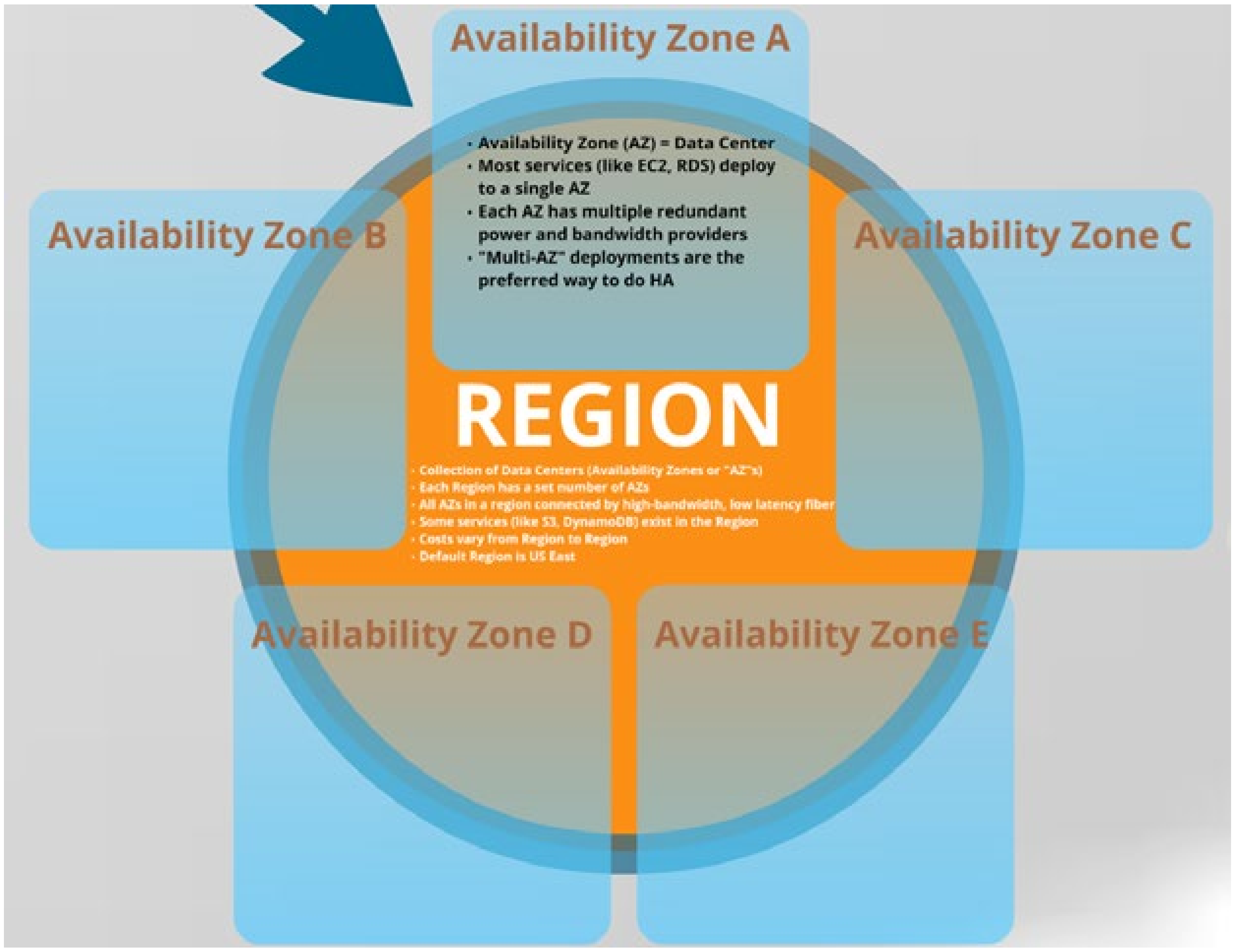
Availability Zone A
• Availability Zone (AZ) = Data Center
• Most services (like EC2, RDS) deploy to a single AZ
• Each AZ has multiple redundant power and bandwidth providers
• "Multi-AZ" deployments are the preferred way to do HA
Availability Zone B
Availability Zone C
REGION
• Collection of Data Centers (Availability Zones or "AZ"s)
• Each Region has a set number of AZs
• All AZs in a region connected by high-bandwidth, low latency fiber
• Some services (like S3, DynamoDB) exist in the Region
• Costs vary from Region to Region
• Default Region is US East
Availability Zone D
Availability Zone E

Figure 2. An AWS Region with Five AZs

In addition to Regions and AZs, AWS also provides dozens of lightweight caching servers known as "Edge Locations." Currently used only by the content delivery network (CDN) product called Amazon CloudFront and the DNS product Amazon Route 53, these Edge Locations cache content and DNS records closer to individual users, whereby reducing latency and increasing web page load times. AWS currently has 42 Edge Locations around the globe.

AWS is aggressively adding AZs, Regions, and Edge Locations to serve users better. End customers get immediate access to new locations as they are added.

The Top Services

Although AWS currently offers more than 30 services, there are a select few that provide the majority of the utility and benefits. If your company starts using AWS, it's highly likely that you'll be using one or more of the following services:

- Amazon Simple Storage Service (Amazon S3)
- Amazon Elastic Compute Cloud (Amazon EC2)
- Amazon Elastic Block Store (Amazon EBS)
- Amazon Relational Database Service (Amazon RDS)
- Elastic Load Balancing
- Auto Scaling
- Amazon CloudWatch

Next, we'll explain each of these services and what they do.

Amazon S3

Amazon S3 is where many AWS customers begin. It's a simple Write Once Read Many (WORM) object store that lets users securely, durably, and cost-effectively store assets in a Region. "Write once" means that objects cannot be changed after it is written, and "read many" means that multiple copies of the object are made. So, when users access the file, many users can retrieve the file quickly (by doing parallel reads across many disk heads). When a user puts an object into Amazon S3, multiple copies of that object are made on multiple servers across multiple AZs. This means that the file is highly durable, and data loss is nearly impossible.

Amazon S3 is especially well suited for storing and serving static web assets like HTML, CSS, Javascript, and image files. When users load a web page in their browser, the browser makes a remote call every time it hits a CSS, Javascript, and image tag. If Amazon S3 is used to serve those assets, it can typically serve the assets much faster than your company website, which now runs much faster since it's only serving HTML.

Amazon EC2 and Amazon EBS

It's estimated that most customers spend north of 70 percent of their AWS bill on Amazon EC2 instances alone. Amazon EC2 is likely to be at the heart of your AWS deployment. At its essence, Amazon EC2 is simply a virtual server that contains a hardware footprint (known as "instance type") and a software footprint (known as an Amazon Machine Image or "AMI"). There are currently 18 instance types available, with varying amounts of CPU, RAM, and local storage—some even offering SSD disks.

You'll typically start out with an Amazon-provided vanilla basic that is OS only (both Linux and Windows supported), and then you will install and configure your application into that AMI. Amazon gives you the ability to burn a custom AMI (also known as "golden image") of your software, and then, for instance, you can use that image to deploy other servers of that type (think of a farm of web servers, all configured exactly the same, sitting behind a load balancer).

Amazon also now offers a "Marketplace" that contains thousands of vendor-provided AMIs that you can launch. The Marketplace can be a great way to evaluate a particular piece of software before deciding to purchase.

Many companies use Amazon EC2 with Auto Scaling (discussed later) to allow their web apps to automatically scale up to and serve variable peak demand, while automatically scaling down to very small numbers of servers (which saves money) during low usage times like nights and weekends.

Although most of the instance types come with some local disk, that disk is volatile and tied directly to the instance's lifetime. If, like most AWS customers, you need a persistent disk that exists independent of instance lifetime, you'll soon be using Amazon Elastic Block Store (Amazon EBS).

Amazon EBS functions nearly the same as a physical 3.5-inch disk that you would install directly into a server. You first provision the size (and optionally I/O throughput) of the volume, and then attach one or more volumes to a single Amazon EC2 instance, format them, and use them for Amazon EC2-accessible storage. Amazon EBS is very fast—in most cases faster than even local disk—and a user could even stripe multiple volumes together for very fast reads and writes. Amazon EBS is a great place to store database data, but if you're looking for a database, AWS has an even better option.

Amazon RDS

Amazon RDS is a fully managed, administration-free install of MySQL, Oracle, or SQL Server. Like many AWS managed services, the user gives up some control (like fine-tuning performance) in exchange for having an easy-to-use, scalable database.

With Amazon RDS, a user provisions a server type (similar to the Amazon EC2 instance types) and receives a client interface URL against which they can interact with their database. The exposed interface can then be used to create, insert into, and query database tables.

Like Amazon EC2, the user can resize their instance up or down and can attach security groups—stateful, easy-to-configure firewalls that limit access to the resource.

With Amazon RDS, setting up weekly or daily database backups is as simple as using a checkbox in the web GUI. For MySQL particularly, Amazon RDS allows quick, easy setup of Slaves and Read Replicas, allowing your database layer to be fault tolerant and scale up with read volume.

Auto Scaling, Elastic Load Balancing, and Amazon CloudWatch

With the popularity of smartphones and tablets, more users are increasingly creating greater demands on websites and the back-end servers. There's also arguably more variability in the minute-to-minute load on a website, with bursts of traffic from email drops and press releases becoming trickles of traffic later on that evening when users are in bed.

Non-cloud websites would have dealt with that traffic by over-provisioning and under-utilizing large farms of servers, configured for the expected peak demand, plus maybe 10 percent. The problem with these architectures was not only the waste associated with powering and cooling (say, 100 servers at night when you only need two), but also with an even worse scenario of not keeping ahead of demand (having perhaps 100 servers when you really need 200).

Auto Scaling solves this problem by dynamically growing and shrinking the number of servers you need based on the real-time demand. Auto Scaling typically involves three services: Auto Scaling groups, Amazon CloudWatch alarms, and Elastic Load Balancing.

You simply preconfigure your Auto Scaling group to absolute maximums, minimums, Amazon EC2 instance types, and AMI, and then set up several Amazon CloudWatch alarms to monitor metrics (like load balancing requests), and dynamically add and remove instances from the group (and from the load balancing rotation).

As your Elastic Load Balancing then sees request volume increase and decrease, it fires the preconfigured Amazon CloudWatch alarms, which then immediately add capacity to or take capacity from your web tier. No more under-utilization, and most importantly, no more serving 404s to users because you don't have enough servers to deal with their requests.

Interfaces

To interact with all of these various services, AWS provides three interfaces, each with various benefits:

- AWS web GUI
- API (and the command line tools that wrap them)
- Language- and platform-specific SDKs

The web GUI is probably how most users begin using AWS, and it's the first thing you see when you create an account online. It exposes most of the features of the most popular services that AWS provides, and it has great point-and-click support for Amazon EC2, Elastic Load Balancing, Amazon EBS, Amazon RDS, Amazon Cloud-Watch, and Amazon S3. The one glaring omission from that list is Auto Scaling, which is currently only supported via the APIs and their command-line tools.

Although the web GUI works perfectly well for most beginner users, in order to access the full power of AWS, one must use the APIs. Only the APIs offer 100 percent access to all the services and features that AWS provides, and of the APIs, the REST API (not SOAP API), is without a doubt the most common.

Almost all the services have command-line tools that wrap the REST APIs, making them even simpler to use. For example, using the Amazon EC2 command-line tools, one could create two servers simply by making the following call:

```
ec2-run-instances ami-1a2b3c4d -n 2 -k my-key-pair --availability-zone us-east-1a
```

Command-line tools exist for each of the popular services, and each is downloaded independently (one down-loads the command-line tools for Amazon EC2 separately from the Amazon RDS command-line tools).

You can also access AWS via the language- and platform-specific SDKs. Supported languages and platforms vary from service to service, but commonly, PHP, Java, Python, and most other popular web languages are supported, in addition to Android and iOS platforms. Using the SDKs, one could write custom web or mobile apps that inter-act directly with AWS using the language your developers use daily.

Learn More

Learn more about how you can improve productivity, enhance efficiency, and sharpen your competitive edge through training.

AWS Essentials

Architecting on AWS

Developing on AWS

Systems Operations on AWS

Visit **www.globalknowledge.com** or call **1-800-COURSES (1-800-268-7737)** to speak with a Global Knowledge training advisor.

About the Author

Rich Morrow is a full-stack generalist with particular depth in and love for cloud and big data technologies. When he's not flying around the country for Global Knowledge providing training on AWS and Hadoop to the Fortune 500, he's probably writing or speaking about cloud, big data, and mobile topics for GigaOM.

Additional Reading

Hopefully, we've been able to clarify some of the inner workings of AWS and share its value with you. If you're inspired to learn more about the services, the links below can be helpful resources.

AWS Documentation

Amazon EC2 Documentation

Auto Scaling Documentation

Elastic Load Balancing Documentation

Amazon S3 Documentation

Amazon RDS Documentation

Amazon CloudWatch Documentation

AWS Free Tier

Getting Started with AWS Guide

Global Knowledge AWS Training