AWS Whitepaper

Web Application Hosting in the AWS Cloud

Web Application Hosting in the AWS Cloud: AWS Whitepaper

Table of Contents

Web Application Hosting in the AWS Cloud

Publication date: **August 20, 2021** (*Document revisions*)

Abstract

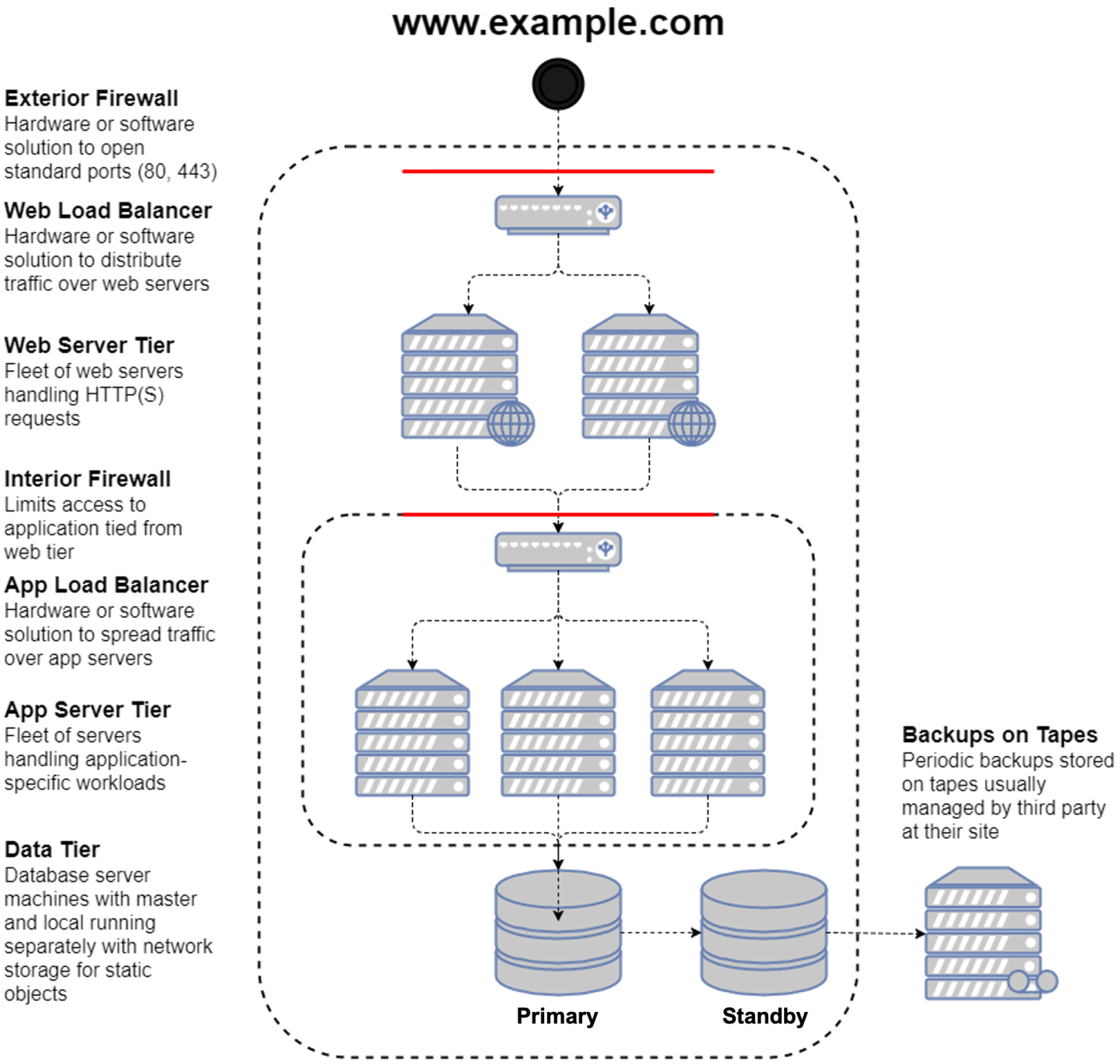
Traditional on-premises web architectures require complex solutions and accurate reserved capacity forecast in order to ensure reliability. Dense peak traffic periods and wild swings in traffic patterns result in low utilization rates of expensive hardware. This yields high operating costs to maintain idle hardware, and an inefficient use of capital for underused hardware.

Amazon Web Services (AWS) provides a reliable, scalable, secure, and highly performing infrastructure for the most demanding web applications. This infrastructure matches IT costs with customer traffic patterns in near-real time.

This whitepaper is meant for IT Managers and System Architects who want to understand how to run traditional web architectures in the cloud to achieve elasticity, scalability, and reliability.

An overview of traditional web hosting

Scalable web hosting is a well-known problem space. The following image depicts a traditional web hosting architecture that implements a common three-tier web application model.
In this model, the architecture is separated into presentation, application, and persistence layers. Scalability is provided by adding hosts at these layers. The architecture also has built-in performance, failover, and availability features. The traditional web hosting architecture is easily ported to the AWS Cloud with only a few modifications.



A traditional web hosting architecture

The following sections look at why and how such an architecture should be and could be deployed in the AWS Cloud.

Web application hosting in the cloud using AWS

The first question you should ask concerns the value of moving a classic web application hosting solution into the AWS Cloud. If you decide that the cloud is right for you, you'll need a suitable architecture. This section helps you evaluate an AWS Cloud solution. It compares deploying your web application in the cloud to an on-premises deployment, presents an AWS Cloud architecture for hosting your application, and discusses the key components of the AWS Cloud Architecture solution.

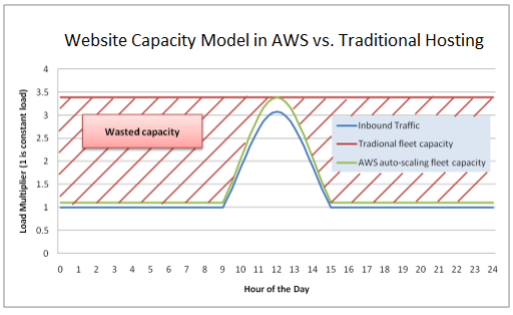
How AWS can solve common web application hosting issues

If you're responsible for running a web application, you could face a variety of infrastructure and architectural issues for which AWS can provide seamless and cost-effective solutions. The following are some of the benefits of using AWS over a traditional hosting model.

A cost-effective alternative to oversized fleets needed to handle peaks

In the traditional hosting model, you have to provision servers to handle peak capacity. Unused cycles are wasted outside of peak periods. Web applications hosted by AWS can leverage on-demand provisioning of additional servers, so you can constantly adjust capacity and costs to actual traffic patterns.

For example, the following graph shows a web application with a usage peak from 9AM to 3PM and less usage for the remainder of the day. An automatic scaling approach based on actual traffic trends, which provisions resources only when needed, would result in less wasted capacity and a greater than 50 percent reduction in cost.



An example of wasted capacity in a classic hosting model

A scalable solution to handling unexpected traffic peaks

A more dire consequence of the slow provisioning associated with a traditional hosting model is the inability to respond in time to unexpected traffic spikes. There are a number of stories about web applications becoming unavailable because of an unexpected spike in traffic after the site is mentioned in popular media. In the AWS Cloud, the same on-demand capability that helps web applications scale to match regular traffic spikes can also handle an unexpected load. New hosts can be launched and are readily available in a matter of minutes, and they can be taken offline just as quickly when traffic returns to normal.

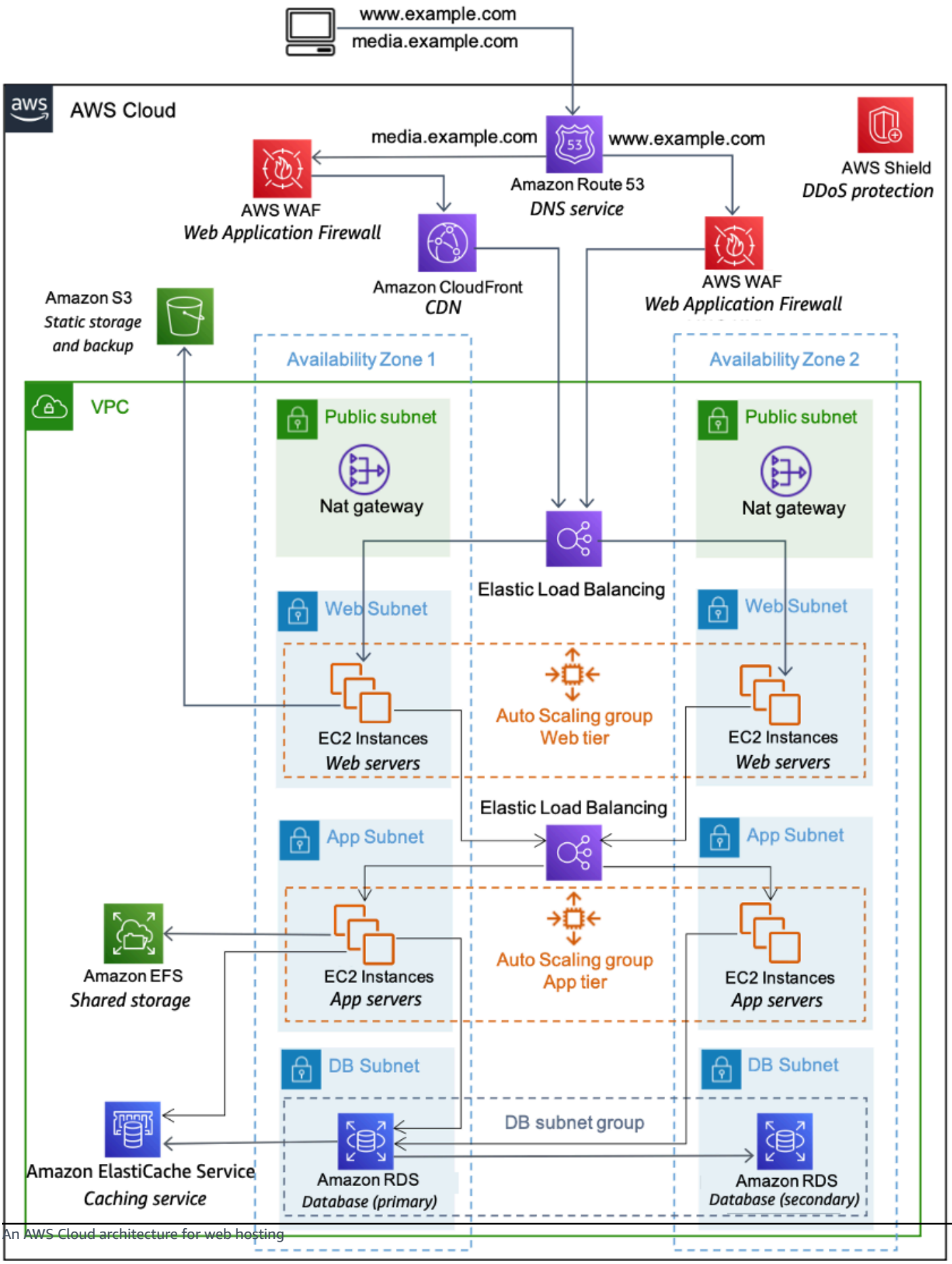
An on-demand solution for test, load, beta, and reproduction environments

The hardware costs of building and maintaining a traditional hosting environment for a production web application don't stop with the production fleet. Often, you need to create preproduction, beta, and testing fleets to ensure the quality of the web application at each stage of the development lifecycle. While you can make various optimizations to ensure the highest possible use of this testing hardware, these parallel fleets are not always used optimally, and a lot of expensive hardware sits unused for long periods of time.

In the AWS Cloud, you can provision testing fleets as and when you need them. This not only eliminates the need for pre-provisioning resources days or months prior to the actual usage, but gives you the flexibility to tear down the infrastructure components when you do not need them. Additionally, you can simulate user traffic on the AWS Cloud during load testing. You can also use these parallel fleets as a staging environment for a new production release. This enables quick switchover from current production to a new application version with little or no service outages.

An AWS Cloud architecture for web hosting

The following figure provides another look at that classic web application architecture and how it can leverage the AWS Cloud computing infrastructure.


www.example.com
media.example.com
AWS Cloud
media.example.com
www.example.com
AWS Shield
DDoS protection
AWS WAF
Web Application Firewall
Amazon Route 53
DNS service
Amazon CloudFront
CDN
AWS WAF
Web Application Firewall
Amazon S3
Static storage and backup
Availability Zone 1
Availability Zone 2
VPC
Public subnet
Nat gateway
Public subnet
Nat gateway
Elastic Load Balancing
Web Subnet
Web Subnet
EC2 Instances
Web servers
Auto Scaling group
Web tier
EC2 Instances
Web servers
Elastic Load Balancing
App Subnet
App Subnet
Amazon EFS
Shared storage
EC2 Instances
App servers
Auto Scaling group
App tier
EC2 Instances
App servers
DB Subnet
DB Subnet
DB subnet group
Amazon ElastiCache Service
Caching service
Amazon RDS
Database (primary)
Amazon RDS
Database (secondary)

An example of a web hosting architecture on AWS

1. **DNS services with Amazon Route 53** – Provides DNS services to simplify domain management.
2. **Edge caching with Amazon CloudFront** – Edge caches high-volume content to decrease the latency to customers.
3. **Edge security for Amazon CloudFront with AWS WAF** – Filters malicious traffic, including cross site scripting (XSS) and SQL injection via customer-defined rules.
4. **Load balancing with Elastic Load Balancing (ELB)** – Enables you to spread load across multiple Availability Zones and Amazon EC2 Auto Scaling groups for redundancy and decoupling of services.
5. **DDoS protection with AWS Shield** – Safeguards your infrastructure against the most common network and transport layer DDoS attacks automatically.
6. **Firewalls with security groups** – Moves security to the instance to provide a stateful, host-level firewall for both web and application servers.
7. **Caching with Amazon ElastiCache** – Provides caching services with Redis or Memcached to remove load from the app and database, and lower latency for frequent requests.
8. **Managed database with Amazon Relational Database Service (Amazon RDS)** – Creates a highly available, multi-AZ database architecture with six possible DB engines.
9. **Static storage and backups with Amazon Simple Storage Service (Amazon S3)** – Enables simple HTTP-based object storage for backups and static assets like images and video.

Key components of an AWS web hosting architecture

The following sections outline some of the key components of a web hosting architecture deployed in the AWS Cloud, and explain how they differ from a traditional web hosting architecture.

Network management

In the AWS Cloud, the ability to segment your network from that of other customers enables a more secure and scalable architecture. While security groups provide host-level security (see the Host Security section), Amazon Virtual Private Cloud (Amazon VPC) enables you to launch resources in a logically isolated and virtual network that you define.

Amazon VPC is a service that gives you full control over the details of your networking setup in AWS. Examples of this control include creating public-facing subnets for web servers, and private

subnets with no internet access for your databases. Additionally, Amazon VPC enables you to create hybrid architectures by using hardware virtual private networks (VPNs), and use the AWS Cloud as an extension of your own data center.

Amazon VPC also includes IPv6 support in addition to traditional IPv4 support for your network.

Content delivery

When your web traffic is geo-dispersed, it's not always feasible and certainly not cost effective to replicate your entire infrastructure across the globe. A Content Delivery Network (CDN) provides you the ability to utilize its global network of edge locations to deliver a cached copy of web content such as videos, webpages, images and so on to your customers. To reduce response time, the CDN utilizes the nearest edge location to the customer or originating request location to reduce the response time. Throughput is dramatically increased given that the web assets are delivered from cache. For dynamic data, many CDNs can be configured to retrieve data from the origin servers.

You can use CloudFront to deliver your website, including dynamic, static, and streaming content, using a global network of edge locations. CloudFront automatically routes requests for your content to the nearest edge location, so content is delivered with the best possible performance. CloudFront is optimized to work with other AWS services, like Amazon S3 and Amazon Elastic Compute Cloud (Amazon EC2). CloudFront also works seamlessly with any origin server that is not an AWS origin server, which stores the original, definitive versions of your files.

Like other AWS services, there are no contracts or monthly commitments for using CloudFront – you pay only for as much or as little content as you actually deliver through the service.

Additionally, any existing solutions for edge caching in your web application infrastructure should work well in the AWS Cloud.

Managing public DNS

Moving a web application to the AWS Cloud requires some Domain Name System (DNS) changes. To help you manage DNS routing, AWS provides Amazon Route 53, a highly available and scalable cloud DNS web service. Route 53 is designed to give developers and businesses an extremely reliable and cost-effective way to route end users to internet applications by translating names such as "www.example.com" into numeric IP addresses such as 192.0.2.1, that computers use to connect to each other. Route 53 is fully compliant with IPv6 as well.

Host security

In addition to inbound network traffic filtering at the edge, AWS also recommends web applications apply network traffic filtering at the host level. Amazon EC2 provides a feature named *security groups*. A security group is analogous to an inbound network firewall, for which you can specify the protocols, ports, and source IP ranges that are allowed to reach your EC2 instances.

You can assign one or more security groups to each EC2 instance. Each security group allows appropriate traffic in to each instance. Security groups can be configured so that only specific subnets, IP addresses, and resources have access to an EC2 instance. Alternatively, they can reference other security groups to limit access to EC2 instances that are in specific groups.

In the AWS web hosting architecture in Figure 3, the security group for the web server cluster might allow access only from the web-layer Load Balancer and only over TCP on ports 80 and 443 (HTTP and HTTPS). The application server security group, on the other hand, might allow access only from the application-layer Load Balancer. In this model, your support engineers would also need to access the EC2 instances, what can be achieved with AWS Systems Manager Session Manager. For a deeper discussion on security, see AWS Cloud Security, which contains security bulletins, certification information, and security whitepapers that explain the security capabilities of AWS.

Load balancing across clusters

Hardware load balancers are a common network appliance used in traditional web application architectures. AWS provides this capability through the Elastic Load Balancing (ELB) service. ELB automatically distributes incoming application traffic across multiple targets, such as EC2 instances, containers, IP addresses, AWS Lambda functions, and virtual appliances. It can handle the varying load of your application traffic in a single Availability Zone or across multiple Availability Zones. Elastic Load Balancing offers four types of load balancers that all feature the high availability, automatic scaling, and robust security necessary to make your applications fault tolerant.

Finding other hosts and services

In the traditional web hosting architecture, most of your hosts have static IP addresses. In the AWS Cloud, most of your hosts have dynamic IP addresses. Although every EC2 instance can have both public and private DNS entries and will be addressable over the internet, the DNS entries and the IP addresses are assigned dynamically when you launch the instance. They cannot be manually assigned. Static IP addresses (Elastic IP addresses in AWS terminology) can be assigned to running

instances after they are launched. You should use Elastic IP addresses for instances and services that require consistent endpoints, such as primary databases, central file servers, and EC2-hosted load balancers.

Caching within the web application

In-memory application caches can reduce load on services and improve performance and scalability on the database tier by caching frequently used information. Amazon ElastiCache is a web service that makes it easy to deploy, operate, and scale an in-memory cache in the cloud. You can configure the in-memory cache you create to automatically scale with load and to automatically replace failed nodes. ElastiCache is protocol-compliant with Memcached and Redis, which simplifies migration from your current on-premises solution.

Database configuration, backup, and failover

Many web applications contain some form of persistence, usually in the form of a relational or non-relational database. AWS offers both relational and non-relational database services. Alternatively, you can deploy your own database software on an EC2 instance. The following table summarizes these options, which are discussed in greater detail in this section.

Table 1 — Relational and non-relational database solutions

	Relational Database Solutions	NoSQL Solutions
Managed database service	Amazon RDS for MySQL, Oracle, SQL Server, MariaDB, PostgreSQL, Amazon Aurora	Amazon DynamoDB , Amazon Keyspaces, Amazon Neptune, Amazon QLDB, Amazon Timestream
Self-managed	Hosting a relational database management system (DBMS) on an Amazon EC2 instance	Hosting a non-relational database solution on an EC2 instance

Amazon RDS

Amazon Relational Database Service (Amazon RDS) gives you access to the capabilities of a familiar MySQL, PostgreSQL, Oracle, and Microsoft SQL Server database engine. The code, applications,

and tools that you already use can be used with Amazon RDS. Amazon RDS automatically patches the database software and backs up your database, and it stores backups for a user-defined retention period. It also supports point-in-time recovery. You benefit from the flexibility of being able to scale the compute resources or storage capacity associated with your relational database instance by making a single API call.

Amazon RDS Multi-AZ deployments increase your database availability and protect your database against unplanned outages. Amazon RDS Read Replicas provide read-only replicas of your database, so you can scale out beyond the capacity of a single database deployment for read-heavy database workloads. As with all AWS services, no upfront investments are required, and you pay only for the resources you use.

Hosting a relational database management system (RDBMS) on an Amazon EC2 instance

In addition to the managed Amazon RDS offering, you can install your choice of RDBMS (such as MySQL, Oracle, SQL Server, or DB2) on an EC2 instance and manage it yourself. AWS customers hosting a database on Amazon EC2 successfully use a variety of primary/standby and replication models, including mirroring for read-only copies and log shipping for always-ready passive replicas.

When managing your own database software directly on Amazon EC2, you should also consider the availability of fault-tolerant and persistent storage. For this purpose, we recommend that databases running on Amazon EC2 use Amazon Elastic Block Store (Amazon EBS) volumes, which are similar to network-attached storage.

For EC2 instances running a database, you should place all database data and logs on EBS volumes. These will remain available even if the database host fails. This configuration allows for a simple failover scenario, in which a new EC2 instance can be launched if a host fails, and the existing EBS volumes can be attached to the new instance. The database can then pick up where it left off.

EBS volumes automatically provide redundancy within the Availability Zone. If the performance of a single EBS volume is not sufficient for your databases needs, volumes can be striped to increase input/output operations per second (IOPS) performance for your database.

For demanding workloads, you can also use EBS Provisioned IOPS, where you specify the IOPS required. If you use Amazon RDS, the service manages its own storage so you can focus on managing your data.

Non-relational databases

In addition to support for relational databases, AWS also offers a number of managed non-relational databases:

- **Amazon DynamoDB** is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. Using the AWS Management Console or the DynamoDB API, you can scale capacity up or down without downtime or performance degradation. Because DynamoDB handles the administrative burdens of operating and scaling distributed databases to AWS, you don't have to worry about hardware provisioning, setup and configuration, replication, software patching, or cluster scaling.
- **Amazon DocumentDB** (with MongoDB compatibility) is a database service that is purpose-built for JSON data management at scale, fully managed and runs on AWS, and enterprise-ready with high durability.
- **Amazon Keyspaces** (for Apache Cassandra) is a scalable, highly available, and managed Apache Cassandra-compatible database service. With Amazon Keyspaces, you can run your Cassandra workloads on AWS using the same Cassandra application code and developer tools that you use today.
- **Amazon Neptune** is a fast, reliable, fully managed graph database service that makes it easy to build and run applications that work with highly connected datasets. The core of Amazon Neptune is a purpose-built, high-performance graph database engine optimized for storing billions of relationships and querying the graph with milliseconds latency.
- **Amazon Quantum Ledger Database (Amazon QLDB)** (QLDB) is a fully managed ledger database that provides a transparent, immutable, and cryptographically verifiable transaction log owned by a central trusted authority. QLDB can be used to track each and every application data change and maintains a complete and verifiable history of changes over time.
- **Amazon Timestream** is a fast, scalable, and serverless time series database service for IoT and operational applications that makes it easy to store and analyze trillions of events per day up to 1,000 times faster and at as little as 1/10th the cost of relational databases.

Additionally, you can use Amazon EC2 to host other non-relational database technologies you may be working with.

Storage and backup of data and assets

There are numerous options within the AWS Cloud for storing, accessing, and backing up your web application data and assets. Amazon S3 provides a highly available and redundant object store. Amazon S3 is a great storage solution for somewhat static or slow-changing objects, such as images, videos, and other static media. Amazon S3 also supports edge caching and streaming of these assets by interacting with CloudFront.

For attached file system-like storage, EC2 instances can have EBS volumes attached. These act like mountable disks for running EC2 instances. Amazon EBS is great for data that needs to be accessed as block storage and that requires persistence beyond the life of the running instance, such as database partitions and application logs.

In addition to having a lifetime that is independent of the EC2 instance, you can take snapshots of EBS volumes and store them in Amazon S3. Because EBS snapshots only back up changes since the previous snapshot, more frequent snapshots can reduce snapshot times. You can also use an EBS snapshot as a baseline for replicating data across multiple EBS volumes and attaching those volumes to other running instances.

EBS volumes can be as large as 16TB, and multiple EBS volumes can be striped for even larger volumes or for increased input/output (I/O) performance. To maximize the performance of your I/O-intensive applications, you can use Provisioned IOPS volumes. Provisioned IOPS volumes are designed to meet the needs of I/O-intensive workloads, particularly database workloads that are sensitive to storage performance and consistency in random access I/O throughput.

You specify an IOPS rate when you create the volume and Amazon EBS provisions that rate for the lifetime of the volume. Amazon EBS currently supports IOPS per volume ranging from maximum of 16000 (for all instance types) up to 64,000 (for instances built on Nitro System). You can stripe multiple volumes together to deliver thousands of IOPS per instance to your application. Apart from this, for higher throughput and mission critical workloads requiring sub-millisecond latency, you can use io2 block express volume type which can support up-to 256,000 IOPS with a maximum storage capacity of 64TB.

Automatically scaling the fleet

One of the key differences between the AWS Cloud architecture and the traditional hosting model is that AWS can automatically scale the web application fleet on demand to handle changes in traffic. In the traditional hosting model, traffic forecasting models are generally used to provision

hosts ahead of projected traffic. In AWS, instances can be provisioned on the fly according to a set of triggers for scaling the fleet out and back in.

The Auto Scaling service can create capacity groups of servers that can grow or shrink on demand. Auto Scaling also works directly with CloudWatch for metrics data and with Elastic Load Balancing to add and remove hosts for load distribution. For example, if the web servers are reporting greater than 80 percent CPU utilization over a period of time, an additional web server could be quickly deployed and then automatically added to the load balancer for immediate inclusion in the load balancing rotation.

As shown in the AWS web hosting architecture model, you can create multiple Auto Scaling groups for different layers of the architecture, so that each layer can scale independently. For example, the web server Auto Scaling group might trigger scaling in and out in response to changes in network I/O, whereas the application server Auto Scaling group might scale out and in according to CPU utilization. You can set minimums and maximums to help ensure 24/7 availability and to cap the usage within a group.

Auto Scaling triggers can be set both to grow and to shrink the total fleet at a given layer to match resource utilization to actual demand. In addition to the Auto Scaling service, you can scale Amazon EC2 fleets directly through the Amazon EC2 API, which allows for launching, terminating, and inspecting instances.

Additional security features

The number and sophistication of Distributed Denial of Service (DDoS) attacks are rising. Traditionally, these attacks are difficult to fend off. They often end up being costly in both mitigation time and power spent, as well as the opportunity cost from lost visits to your website during the attack. There are a number of AWS factors and services that can help you defend against such attacks. One of them is the scale of the AWS network. The AWS infrastructure is quite large, and enables you to leverage our scale to optimize your defense. Several services, including Elastic Load Balancing, Amazon CloudFront, and Amazon Route 53, are effective at scaling your web application in response to a large increase in traffic.

The infrastructure protection services in particular help with your defense strategy:

- **AWS Shield** is a managed DDoS protection service that helps safeguard against various forms of DDoS attack vectors. The standard offering of AWS Shield is free and automatically active throughout your account. This standard offering helps to defend against the most common network and transportation layer attacks. In addition to this level, the advanced offering grants

higher levels of protection against your web application by providing you with near real-time visibility into an ongoing attack, as well as integrating at higher levels with the services mentioned earlier. Additionally, you get access to the AWS DDoS Response Team (DRT) to help mitigate large-scale and sophisticated attacks against your resources.

- **AWS WAF** (Web Application Firewall) is designed to protect your web applications from attacks that can compromise availability or security, or otherwise consume excessive resources. AWS WAF works in line with CloudFront or Application Load Balancer, along with your custom rules, to defend against attacks such as cross-site scripting, SQL injection, and DDoS. As with most AWS services, AWS WAF comes with a fully featured API that can help automate the creation and editing of rules for your AWS WAF instance as your security needs change.
- **AWS Firewall Manager** is a security management service which enables you to centrally configure and manage firewall rules across your accounts and applications in AWS Organizations. As new applications are created, AWS Firewall Manager makes it easy to bring new applications and resources into compliance by enforcing a common set of security rules.

Failover with AWS

Another key advantage of AWS over traditional web hosting is the Availability Zones that give you easy access to redundant deployment locations. Availability Zones are physically distinct locations that are engineered to be insulated from failures in other Availability Zones. They provide inexpensive, low-latency network connectivity to other Availability Zones in the same AWS Region. As the AWS web hosting architecture diagram shows, AWS recommends that you deploy EC2 hosts across multiple Availability Zones to make your web application more fault tolerant.

It's important to ensure that there are provisions for migrating single points of access across Availability Zones in the case of failure. For example, you should set up a database standby in a second Availability Zone so that the persistence of data remains consistent and highly available, even during an unlikely failure scenario. You can do this on Amazon EC2 or Amazon RDS with the click of a button.

While some architectural changes are often required when moving an existing web application to the AWS Cloud, there are significant improvements to scalability, reliability, and cost-effectiveness that make using the AWS Cloud well worth the effort. The next section discusses those improvements.

Key considerations when using AWS for web hosting

There are some key differences between the AWS Cloud and a traditional web application hosting model. The previous section highlighted many of the key areas that you should consider when deploying a web application to the cloud. This section points out some of the key architectural shifts that you need to consider when you bring any application into the cloud.

No more physical network appliances

You cannot deploy physical network appliances in AWS. For example, firewalls, routers, and load balancers for your AWS applications can no longer reside on physical devices, but must be replaced with software solutions. There is a wide variety of enterprise-quality software solutions, whether for load balancing or establishing a VPN connection. This is not a limitation of what can be run on the AWS Cloud, but it is an architectural change to your application if you use these devices today.

Firewalls everywhere

Where you once had a simple demilitarized zone (DMZ) and then open communications among your hosts in a traditional hosting model, AWS enforces a more secure model, in which every host is locked down. One of the steps in planning an AWS deployment is the analysis of traffic between hosts. This analysis will guide decisions on exactly what ports need to be opened. You can create security groups for each type of host in your architecture. You can also create a large variety of simple and tiered security models to enable the minimum access among hosts within your architecture. The use of network access control lists within Amazon VPC can help lock down your network at the subnet level.

Consider the availability of multiple data centers

Think of Availability Zones within an AWS Region as multiple data centers. EC2 instances in different Availability Zones are both logically and physically separated, and they provide an easy-to-use model for deploying your application across data centers for both high availability and reliability. Amazon VPC as a Regional service enables you to leverage Availability Zones while keeping all of your resources in the same logical network.

Treat hosts as ephemeral and dynamic

Probably the most important shift in how you might architect your AWS application is that Amazon EC2 hosts should be considered ephemeral and dynamic. Any application built for the AWS Cloud should not assume that a host will always be available and should be designed with the knowledge that any data in the EC2 instant stores will be lost if an EC2 instance fails.

When a new host is brought up, you shouldn't make assumptions about the IP address or location within an Availability Zone of the host. Your configuration model must be flexible, and your approach to bootstrapping a host must take the dynamic nature of the cloud into account. These techniques are critical for building and running a highly scalable and fault-tolerant application.

Consider containers and serverless

This whitepaper primarily focuses on a more traditional web architecture. However, consider modernizing your web applications by moving to Containers and Serverless technologies, leveraging services like AWS Fargate and AWS Lambda to enable you to abstracts away the use of virtual machines to perform compute tasks. With serverless computing, infrastructure management tasks like capacity provisioning and patching are handled by AWS, so you can build more agile applications that allow you to innovate and respond to change faster.

Consider automated deployment

- **Amazon Lightsail** is an easy-to-use virtual private server (VPS) that offers you everything needed to build an application or website, plus a cost-effective, monthly plan. Lightsail is ideal for simpler workloads, quick deployments, and getting started on AWS. It's designed to help you start small, and then scale as you grow.
- **AWS Elastic Beanstalk** is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, NGINX, Passenger, and IIS. You can simply upload your code, and Elastic Beanstalk automatically handles the deployment, capacity provisioning, load balancing, automatic scaling, and application health monitoring. At the same time, you retain full control over the AWS resources powering your application and can access the underlying resources at any time.
- **AWS App Runner** is a fully managed service that makes it easy for developers to quickly deploy containerized web applications and APIs, at scale and with no prior infrastructure experience

required. Start with your source code or a container image. App Runner automatically builds and deploys the web application and load balances traffic with encryption. App Runner also scales up or down automatically to meet your traffic needs.

- **AWS Amplify** is a set of tools and services that can be used together or on their own, to help front-end web and mobile developers build scalable full stack applications, powered by AWS. With Amplify, you can configure app backends and connect your app in minutes, deploy static web apps in a few clicks, and easily manage app content outside the AWS Management Console.

Conclusion and contributors

Conclusion

There are numerous architectural and conceptual considerations when you are contemplating migrating your web application to the AWS Cloud. The benefits of having a cost-effective, highly scalable, and fault-tolerant infrastructure that grows with your business far outstrips the efforts of migrating to the AWS Cloud.

Contributors

The following individuals and organizations contributed to this document:

- Amir Khairalomoum, Senior Solutions Architect, AWS
- Dinesh Subramani, Senior Solutions Architect, AWS
- Jack Hemion, Senior Solutions Architect, AWS
- Jatin Joshi, Cloud Support Engineer, AWS
- Jorge Fonseca, Senior Solutions Architect, AWS
- Shinduri K S, Solutions Architect, AWS

Further reading

- Deploy Django-based application onto Amazon LightSail
- Deploying a high availability Drupal website to Elastic Beanstalk
- Deploying a high availability PHP application to Elastic Beanstalk
- Deploying a Node.js application with DynamoDB to Elastic Beanstalk
- Getting Started with Linux Web Applications in the AWS Cloud
- Host a Static Website
- Hosting a static website using Amazon S3
- Tutorial: Deploying an ASP.NET core application with Elastic Beanstalk
- Tutorial: How to deploy a .NET sample application using Elastic Beanstalk

Document revisions

To be notified about updates to this whitepaper, subscribe to the RSS feed.

Change	Description	Date
Whitepaper updated	Multiple sections and diagrams updated with new services, features, and updated service limits.	August 20, 2021
Whitepaper updated	Updated icon label for “Caching with ElastiCache” in Figure 3.	September 29, 2019
Whitepaper updated	Multiple sections added and updated for new services. Updated diagrams for additional clarity and services. Addition of VPC as the standard networking method in AWS in “Network Management.” Added section on DDoS protection and mitigation in “Additional Security Features.” Added a small section on serverless architectures for web hosting.	July 1, 2017
Whitepaper updated	Multiple sections updated to improve clarity. Updated diagrams to use AWS icons. Addition of “Managing Public DNS” section for detail on Amazon Route 53. “Finding Other Hosts and Services” section updated for	September 1, 2012

	clarity. “Database Configuration, Backup, and Failover” section updated for clarity and DynamoDB. “Storage and Backup of Data and Assets” section expanded to cover EBS Provisioned IOPS volumes.	
Initial publication	Whitepaper published.	May 1, 2010

Notices

This document is provided for informational purposes only. It represents AWS's current product offerings and practices as of the date of issue of this document, which are subject to change without notice. Customers are responsible for making their own independent assessment of the information in this document and any use of AWS's products or services, each of which is provided "as is" without warranty of any kind, whether express or implied. This document does not create any warranties, representations, contractual commitments, conditions or assurances from AWS, its affiliates, suppliers or licensors. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.