Running Adobe Experience Manager® on AWS

First published July 2016

Updated November 25, 2020

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. 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.

Contents

Abstract

This whitepaper outlines the benefits and strategy for hosting for Adobe Experience Manager (AEM) on Amazon Web Services (AWS). It discusses various migration strategies, architecture choices, and deployment strategies including a reference architecture for self-hosting on AWS. It also provides guidance for disaster recovery, DevOps and high compliance workloads, such as government, finance, and healthcare.

This whitepaper is for technical leaders and business leaders responsible for deploying and managing AEM on AWS.

Introduction

Delivering a fast, secure, and seamless experience is essential in today's digital marketing environment. The need to reach a broader audience across all devices is essential, and a shorter time to market can be a differentiator. Companies are turning to cloud-based solutions to boost business agility, harness new opportunities, and gain cost efficiencies.

Adobe Experience Manager (AEM), is a comprehensive content management solution for building websites, mobile apps, and forms. AEM makes it easy to manage your marketing content and assets. Adopting AWS for running AEM presents many benefits such as increased business agility, added flexibility, and reduced costs.

This whitepaper provides technical guidance for running AEM on AWS. With any deployment on AWS, there are many different considerations and options, so your approach might be different from the approach we walk through in this paper.

Lastly, this whitepaper concludes by discussing security and compliance, architectural components, connectivity, and a strategy you can employ for migration.

Why use AEM on AWS?

Hosting AEM on AWS offers some key benefits such as global capacity, security, reliability, fault tolerance, programmability, and usability. This section discusses several ways in which deploying AEM on AWS is different from deploying it to an on-premises infrastructure.

Flexible Capacity

One of the benefits of using the AWS Cloud is the ability to scale up and down as needed. When using AEM, you have full freedom to scale all of your environments quickly and cost-effectively, giving you opportunities to establish new development, quality assurance (QA), and performance-testing environments.

AEM is frequently used in scenarios that have unknown or significant variations in traffic volume. The on-demand nature of the AWS platform allows you to scale your workloads to support your unique traffic peaks during key events, such as holiday shopping seasons, major sporting events, and large sale events.

Flexible capacity also streamlines upgrades and deployments. AWS makes it very easy to set up a parallel environment, so you can migrate and test your application and content in a production-like environment. Performing the actual production upgrade itself can then be as simple as the change of a domain name system (DNS) entry.

Broad Set of Capabilities

As a leading web content management system solution, AEM is often used by customers as the foundation of their digital marketing platform. Running AEM on AWS provides customers with the benefits of easily integrating third-party solutions for auxiliary experiences such as blogs, and providing additional tools for supporting mobile delivery, analytics, and big data management. You can integrate the open and extensible APIs of both AWS and AEM to create powerful new combinations for your firm. Also, AEM can be used to augment or create headless commerce architectures seamlessly.

With services like Amazon Simple Notification Service (Amazon SNS), Amazon Simple Queue Service (Amazon SQS), and AWS Lambda, AEM functionality can easily be integrated with other third-party functionalities in a decoupled fashion. AWS can also provide a clean, manageable, and auditable approach to decoupled integration with backend systems such as Customer Relationship Management (CRM) and commerce systems.

Benefits of Cloud and Global Availability

Organizations considering a transition to the cloud are often driven by their need to become more agile and innovative. The traditional capital expenditure (Capex) funding model makes it difficult to quickly test new ideas. The AWS Cloud model gives you the agility to quickly spin up new instances on AWS, and the ability to try out new services without investing in large and upfront sunk costs (that is, costs that have already been incurred and can't be recovered). AWS helps to lower customer costs through its pay-for-what-you-use pricing model. Also, as of writing, AWS Global Infrastructure spans 24 geographic regions around the world, enabling customers to deploy on a global footprint quickly and easily.

Security and High-Compliance Workloads

Using AWS, you will gain the control and confidence you need to safely run your business with the most flexible and secure cloud computing environment available today. With AWS, you can improve your ability to meet core security and compliance requirements with a comprehensive set of services and features. The AWS Compliance

Programs will help you understand the robust controls in place at AWS to maintain security and compliance in the cloud. Compliance certifications and attestations are assessed by a third-party, independent auditor.

Running AEM on AWS provides customers with the benefits of leveraging the compliance and security capabilities of AWS, along with the ability to monitor and audit access to AEM using AWS Security, Identity and Compliance services. AWS also offers the GovCloud (US) Regions, which are designed to host sensitive data, regulate workloads, and address the most stringent US government security and compliance requirements.

Adobe Experience Manager Overview

This section highlights some of the key technical elements for AEM and offers some best practice recommendations. This whitepaper focuses on AEM 6.5 (released April 2019).

AEM Platform Overview

A standard AEM architecture consists of three environments: author, publish, and dispatcher. Each of these environments consists of one or more instances.

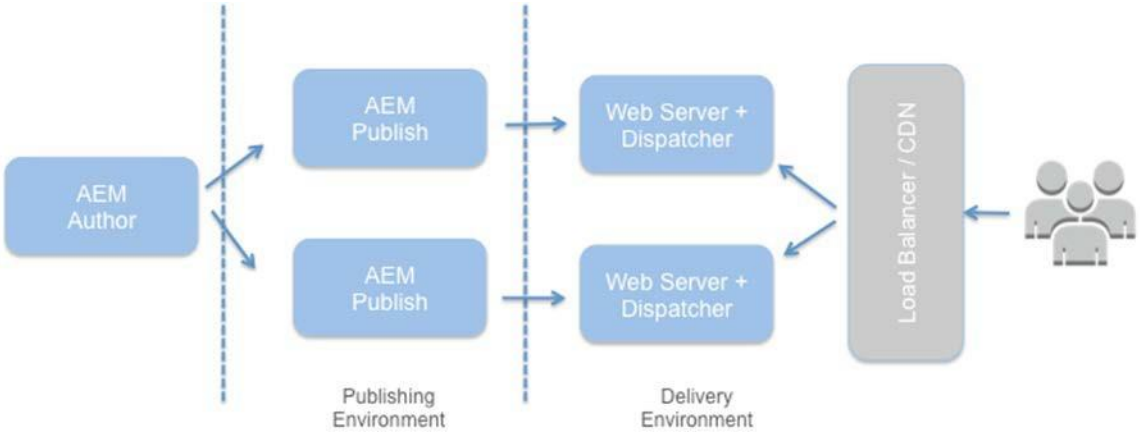

Figure 1 – Sample AEM Architecture

The *author* environment is used for creating and managing the content and layout of an AEM experience. It provides functionality for reviewing and approving content updates, and publishing approved versions of content to the publish environment.

The *publish* environment delivers the experience to the intended audience. It renders the actual pages, with an ability to personalize the experience based on audience characteristics or targeted messaging.

The author and publish instances are Java web applications that have identical installed software. They are differentiated by configuration only.

The *dispatcher* environment is a caching and/or load balancing tool that helps realize a fast and dynamic web authoring environment.

For caching, the dispatcher works as part of an HTTP server, such as Apache HTTP Server, with the aim of storing (or caching) as much of the static website content as possible, and accessing the website's publisher layout engine as infrequently as possible. For caching, the dispatcher module uses the web server's ability to serve static content. The dispatcher places the cached documents in the document root of the web server.

Repositories

Within AEM, everything is content and stored in the underlying repository. AEM's repository is called CRX, it implements the Content Repository API for Java (JCR), and it is based on Apache Jackrabbit Oak.

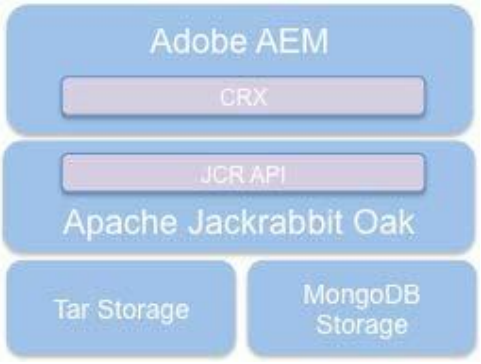

Figure 2 – AEM Storage Options

The Oak storage layer provides an abstraction layer for the actual storage of the content.

MicroKernels act as persistence managers in AEM. There are two primary storage implementations available in AEM 6: Tar Storage and MongoDB Storage. The Tar storage uses tar files. It stores the content as various types of records within larger segments. Journals are used to track the latest state of the repository. The MongoDB

storage leverages MongoDB for sharding and clustering. The repository tree is kept in one MongoDB database where each node is a separate document.

At a high level, Tar MicroKernel (TarMK) is used for performance and MongoDB is used for scalability.

Publish instances are always TarMK. Multiple publish instances with each instance running its own TarMK are referred to as TarMK farm. This is the default deployment for publish environments.

Author instances can either use TarMK for a single author instance or MongoDB when horizontal scaling is required. For TarMK author instance deployments, a cold standby TarMK instance can be configured in another availability zone to provide backup in case the primary author instance fails, although the failover is not automatic.

TarMK is the default persistence system in AEM for both author and publish configurations. Although AEM can be configured to use a different persistence system (such as MongoDB), TarMK is performance-optimized for typical JCR use cases and is very fast. TarMK uses an industry-standard data format that can be quickly and easily backed up, providing high performance and reliable data storage with minimal operational overhead and lower total cost of ownership (TCO).

MongoDB is recommended for AEM author deployments when there are more than 1,000 unique users per day, 100 concurrent users, or high volumes of page edits. (For details, refer to When to use Mongo DB). MongoDB provides high availability, redundancy, and automated failovers for author instances, although performance can be lower than TarMK. A minimum deployment with MongoDB typically involves a MongoDB replica consisting of one primary node and two secondary nodes, with each node running in its separate availability zone.

In AEM, binary data can be stored independently from the content nodes. The binary data is stored in a data store, whereas content nodes are stored in a node store. You can use Amazon Simple Storage Service (Amazon S3) as a shared datastore between publish and author instances to store binary files. This approach makes the cluster high performant. For details, see How to configure S3 as a datastore.

AEM Implementation on AWS

This section outlines the following two deployment options and the key design elements to consider for deploying AEM on AWS.

- Self- or partner-managed deployment
- AEM Managed Services by Adobe

Self- or Partner-Managed Deployment

In a self-managed deployment, the organization itself is responsible for the deployment and maintenance of AEM and the underlying AWS infrastructure. In partner-managed deployment, the organization engages with a partner from the AWS Partner Network (APN) for the deployment and maintenance of AEM and the underlying AWS infrastructure. AEM customizations in both models can be done by the organization or the partner.

For organizations who cannot manage their own deployment of AEM on AWS (either because they do not have the resources or because they are not comfortable), there are several APN partners that specialize in providing managed hosting deployments of AEM on AWS. These companies take care of all aspects of deploying, securing, patching, and maintaining AEM. Some partners also provide design services and custom development for AEM. You can use AWS Partner Finder to find and compare providers that specialize in Adobe products on AWS.

AEM Managed Services

AEM Managed Services by Adobe enables customers to launch faster by deploying on the AWS cloud and also by leaning on best practices and support from Adobe. Organizations and business users can engage customers in minimal time, drive market share, and focus on creating innovative marketing campaigns while reducing the burden on IT.

Cloud Manager, part of the AEM Managed Services offering, is a self-service portal that further enables organizations to self-manage AEM Manager in the cloud. It includes a continuous integration and continuous delivery (CI/CD) pipeline that lets IT teams and implementation partners speed up the delivery of customizations or updates without compromising performance or security. Cloud Manager is only available for Adobe Managed Service customers.

Architecture Options

This section presents a reference architecture for running AEM on AWS along with various architectural options to consider when planning AEM on AWS deployment. Alternately, you can also consider adopting AEM OpenCloud, an open source framework for running AEM on AWS.

Reference Architecture

The following reference architecture is recommended for both self or partner-managed deployment methods. For reference architecture details, see Hosting Adobe Experience Manager® on AWS.

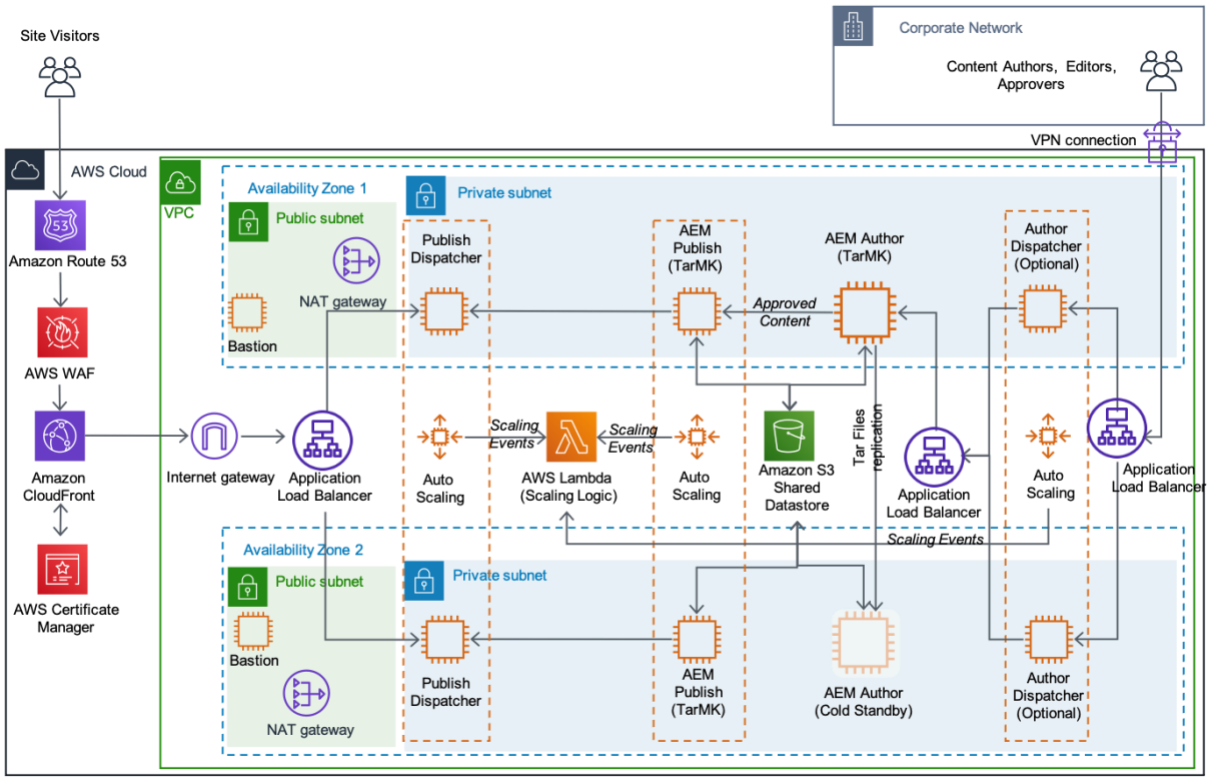

Figure 3 –AEM on AWS Reference Architecture

Reference Architecture Components

Architecture Sizing

For AEM, the right instance type depends on the usage scenario. For AEM author and publish instances in the most common publishing scenario, a solid mix of memory,

CPU, and I/O performance is necessary. Therefore, the Amazon EC2 General Purpose M5 family of instances are good candidates for these environments, depending upon sizing.

Amazon EC2 M5 Instances are the next generation of the Amazon EC2 General Purpose compute instances. M5 instances offer a balance of compute, memory, and networking resources for a broad range of workloads. Additionally, M5d, M5dn, and M5ad instances have local storage, offering up to 3.6TB of NVMe-based SSDs.

AEM Dispatcher is installed on a web server (Apache httpd on Amazon EC2 instance), and it is a key caching layer. It provides caching, load balancing, and application security. Therefore, sizing memory and compute is important, but optimization for I/O is critical for this tier. Amazon Elastic Block Store (Amazon EBS) I/O optimized volumes are recommended. Each dispatcher instance is mapped to a publish instance in a one-to-one fashion in each availability zone.

For all of these instances, Amazon EBS optimization is important. EBS volumes on which AEM is installed should use either General Purpose SSD (GP2) volumes or provisioned Input/Output operations Per Second (IOPS) volumes. This configuration provides a specific level of performance and lower latency for operations.

Adobe recommends Intel Xeon or AMD Opteron CPU with at least 4 cores, and 16 GB of RAM for AEM environments. This translates to Amazon EC2 M5.XL instance type. Typically, you can start with Amazon EC2 M5.2XL instance type and then adjust based on your workload needs. For guidance on selecting the right instance, refer to the Adobe hardware sizing guide.

The specific sizing for the number of servers you need depends on your AEM use case (for example, experience management or digital asset management) and the level of caching that should be applied. At minimum, you need five total servers for a high availability configuration utilizing two Availability Zones. This architecture places a dispatcher-publisher pair in each of the two Availability Zones, and a single author node in one Availability Zone (fronting each of the publish instances with a dispatcher instance). For guidelines for calculating the number of servers required, refer to the Adobe support site.

Load Balancing

In an AEM setup, Elastic Load Balancing is configured to balance traffic to the dispatchers. By default, a load balancer distributes incoming requests evenly across its enabled Availability Zones (AZs). To ensure that a load balancer distributes incoming

requests evenly across all back-end instances (regardless of the Availability Zone that they are in), enable cross-zone load balancing.

For authenticated AEM experiences, authentication is maintained by a login token. When a user logs in, the token information is stored under the tokens node of the corresponding user node in the repository. The value of the token (that is, the session ID) is also stored in the browser as a cookie named login-token. In this case, the load balancer should be configured for sticky sessions, routing requests with the login-token cookie to the same instance. AEM can be configured to recognize the authentication cookie across all publish instances. However, it also requires that all relevant user session information (for example, a shopping cart) is available across all publish instances.

Elastic Load Balancing can be used in front of the dispatchers to provide a Single CNAME URL for the application. The load balancer, in conjunction with AWS Certificate Manager, can be used to provide an HTTPS access and to offload SSL. By using the load balancer, you can further secure your website deployment by moving the publisher instances into a private subnet, allowing access from only the load balancer. The load balancer can also translate the port access from port 80 to the default publish port 4503.

High Availability

For a highly available AEM architecture, the architecture should be set up to leverage AWS strengths. Configure each instance in the AEM cluster for Amazon EC2 Auto Recovery. Additionally, when the cluster is built in conjunction with a load balancer, you can use AWS Auto Scaling to automatically provision nodes across multiple Availability Zones. We recommend that you provision nodes across multiple Availability Zones for high availability, and use multiple AWS Regions to address global deployment considerations as needed. In a multi-Region deployment, you can set up Amazon Route 53 to perform DNS failover based on health checks.

Scaling

A simple way to accomplish scaling is to create separate Amazon Machine Images (AMIs) for the publish instance, dispatcher instance (mapped to publish), and dispatcher instance (mapped to author, if in use). Three separate launch configurations can be created using these AMIs and included in separate Auto Scaling groups.

Newly launched dispatcher instances require a corresponding publish instance and need to author instances to receive future invalidation calls. AWS Lambda can provide scaling logic in response to scale up/down events from Auto Scaling groups. The

scaling logic consists of pairing/unpairing the newly launched dispatcher instance to an available publish instance (or the other way around), updating the replication agent (reverse replication, if applicable) between the newly launched publish instance and author instance, and updating AEM content health check alarms. Each dispatcher instance is mapped to a publish instance in a one-to-one fashion in separate availability zones.

For faster startup and synchronization, you can place the AEM installation on a separate Amazon EBS volume. By taking frequent snapshots of the volume and attaching those snapshots to the newly-launched instances, the need to replicate large amounts of data from the author can be cut down. In the startup process, the publish instance can then trigger author—publish replication to fully ensure the latest content.

Content Delivery

AEM can use a content delivery network (CDN), such as Amazon CloudFront, as a caching layer in addition to the standard AEM dispatcher. When you use a CDN, you need to consider how content is invalidated and refreshed in the CDN when content is updated.

Explicit configuration regarding how long particular resources are held in the CloudFront cache, along with expiration and cache-control headers sent by dispatcher, can help in controlling the CDN cache. Cache control headers can be controlled by using the mod_expires Apache Module.

For API-based invalidation associated with content replication, one approach is to build a custom invalidation workflow and set up an AEM Replication Agent that will use your own ContentBuilder and TransportHandler to invalidate the Amazon CloudFront cache using API. For more details, refer to Using Dispatcher with a CDN.

Dynamic Content

The dispatcher is the caching layer with the AEM product. It allows for defining caching rules at the web server layer. To realize the full benefit of the dispatcher, pages should be fully cacheable. Any element that isn't cacheable will "break" the cache functionality.

To incorporate dynamic elements in a static page, the recommended approach is to use client-side JavaScript, Edge Side Includes (ESIs), or web server level Server Side Includes (SSIs). Within an AWS environment, ESIs can be configured using a solution such as Varnish, replacing the dispatcher. However, using such configuration may not be supported by Adobe.

Amazon S3 Data Store

Binary data can be stored independently from the content nodes in AEM. When deploying on AWS, the binary data store can be Amazon S3, simplifying management and backups. Also, the binary data store can then be shared across author instances and even between author and publish instances, reducing overall storage and data transfer requirements. Refer to *Amazon S3 Data Store* documentation by Adobe to learn how to configure S3 for AEM.

AEM OpenCloud

AEM OpenCloud is an open-source platform for running AEM on AWS. It provides an out-of-the-box solution for provisioning a highly-available AEM architecture which implements auto-scaling, auto-recovery, chaos-testing, CDN, multi-level backup, blue-green deployment, repository upgrade, security, and monitoring capabilities by leveraging a multitude of AWS services.

AEM OpenCloud code base is open source and available on GitHub with an Apache 2 license. The code base is maintained by Shine Solutions Group, an APN Partner. You are free to use AEM OpenCloud on your own or engage with the Shine Solutions Group for custom use cases and implementation support.

AEM OpenCloud supports multiple AEM versions from 6.2 to 6.5, using Amazon Linux 2 or RHEL7 operating system, with two architecture options: full-set and consolidated. This platform can also be built and run in multiple AWS Regions. It is highly configurable and provides a number of customization points where users can provision various other software into their AEM environment provisioning automation.

AEM OpenCloud is available through the AEM OpenCloud on AWS Quick Start, an architecture based on AWS best practices you easily launch in a few clicks.

AEM OpenCloud Full-Set Architecture

A *full-set architecture* is a full-featured environment, suitable for production and staging environments. It includes AEM Publish, Author-Dispatcher, and Publish-Dispatcher EC2 instances within Auto Scaling groups which (combined with an Orchestrator application) provide the capability to manage AEM capacity as the instances scale out and scale in corresponding to the load on the Dispatcher instances. Orchestrator application manages AEM replication and flush agents as instances are created and terminated. This architecture also includes chaos-testing capability by using Netflix Chaos Monkey, which can be configured to randomly terminate either one of those instances within the

auto-scaling groups, or allow the architecture to live in production, continuously verifying that AEM OpenCloud can automatically recover from failure.

AEM Author Primary and Author Standby are managed separately where a failure on Author Primary instance can be mitigated by promoting an Author Standby to become the new Author Primary as soon as possible, while a new environment is being built in parallel and will take over as the new environment, replacing the one which lost its Author Primary.

Full-set architecture uses Amazon CloudFront as the CDN, sitting in front of AEM Publish-Dispatcher load balancer, providing global distribution of AEM cached content.

Full-set offers three types of content backup mechanisms: AEM package backup, live AEM repository EBS snapshots (taken when all AEM instances are up and running), and offline AEM repository EBS snapshots (taken when AEM Author and Publish are stopped). You can use any of these backups for blue-green deployment, providing the capability to replicate a complete environment, or to restore an environment from any point of time.

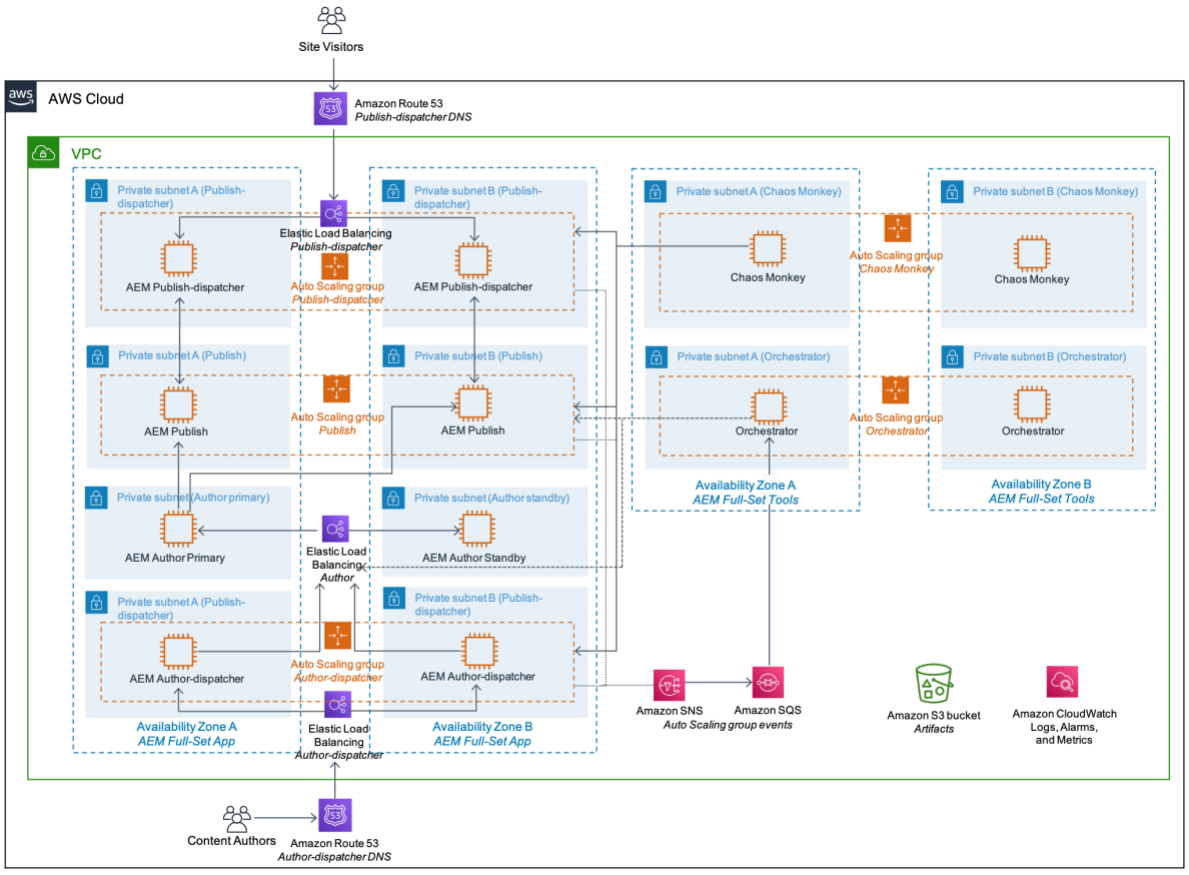

Figure 4 – AEM OpenCloud Full-Set Architecture

On the security front, this architecture provides a minimal attack surface with one public entry point to either Amazon CloudFront distribution or an AEM Publish-Dispatcher load balancer, whereas the other entry point is for AEM Author-Dispatcher load balancer. AEM OpenCloud supports encryption using AWS Key Management Service (AWS KMS) keys across its AWS resources.

The full-set architecture also includes an Amazon CloudWatch Monitoring Dashboard which visualizes the capacity of AEM Author-Dispatcher, Author Primary, Author Standby, Publish, and Publish-Dispatcher, along with their CPU, memory, and disk consumptions. Amazon CloudWatch Alarms are also configured across the most important AWS resources, allowing notification mechanism via an SNS topic.

Consolidated Architecture

A *consolidated architecture* is a cut-down environment where an AEM Author Primary, an AEM Publish, and an AEM Dispatcher are all running on a single Amazon EC2 instance. This architecture is a low-cost alternative suitable for development and testing environments.

This architecture also offers those three types of backup, just like full-set architecture, where the backup AEM package and EBS snapshots are interchangeable between consolidated and full-set environments. This option is useful, for example, when you want to restore production backup from a full-set environment to multiple development environments running consolidated architecture. Another example is if you wanted to upgrade an AEM repository to a newer version in a development environment, which is then pushed through to testing, staging, and eventually production.

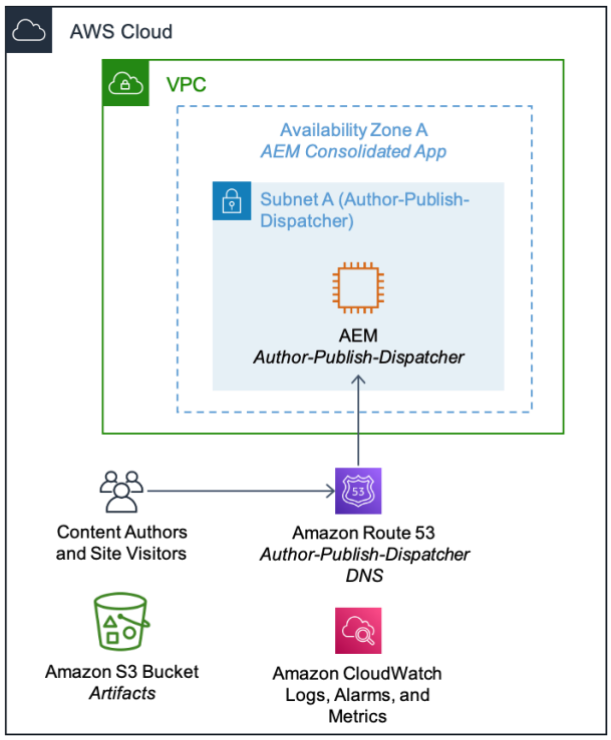

Figure 5 – AEM OpenCloud Consolidated Architecture

Environment Management

To manage multiple environments with a mixture of full-set and consolidated architectures, AEM OpenCloud has a Stack Manager that handles the command executions within AEM instances via AWS Systems Manager. These commands include taking backups, checking environment readiness, running the AEM security checklist, enabling and disabling CRXDE and SAML, deploying multiple AEM packages configured in a descriptor, flushing AEM Dispatcher cache, and promoting the AEM Author Standby instance to Primary.

Other than the Stack Manager, there is also AEM OpenCloud Manager which currently provides Jenkins pipelines for creating and terminating AEM full-set and consolidated architectures, baking AEM Amazon Machine Images(AMIs), executing operational tasks via Stack Manager, and upgrading an AEM repository between versions, (for example, from AEM 6.2 to 6.4, or from AEM 6.4 to 6.5).

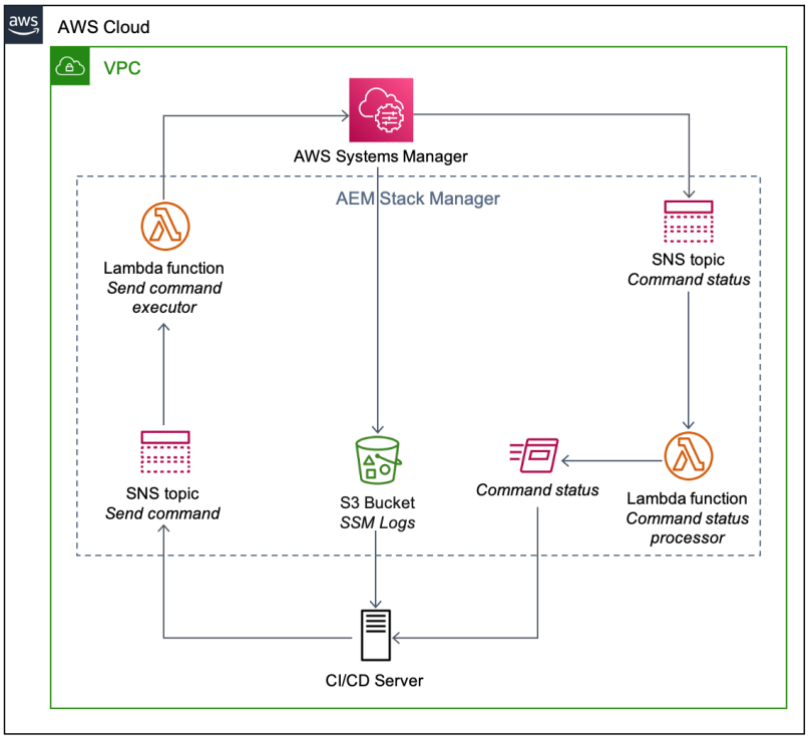

Figure 6 – AEM OpenCloud Stack Manager

Security

The security of the AEM hosting environment can be broken down into two areas: application security and infrastructure security. A crucial first step for application security is to follow the Security Checklist for AEM and the Dispatcher Security Checklist. These checklists cover various parts of security considerations, from running AEM in production mode to using `mod_rewrite` and `mod_security` modules from Apache to prevent Distributed Denial of Service (DDoS) attacks and cross site scripting (XSS) attacks.

From an infrastructure level, AWS provides several security services to secure your environment. These services are grouped into five main categories – network security; data protection; access control; detection, audit, monitoring, and logging; and incident response.

Network Security

One of the core components of network security is Amazon Virtual Private Cloud (Amazon VPC). This service provides multiple layers of network security for your application such as public and private subnets, security groups, and network access

control lists for subnets. Also, VPC endpoints for S3 enable you to privately connect your VPC to Amazon S3.

Amazon CloudFront can offload direct access to your backend infrastructure, and using the Web Application Firewall (WAF) provided by the AWS WAF service, you can apply rules to prevent the application from getting compromised by scripted attacks. The same rules that are encoded in Apache `mod_security` on the dispatcher can be moved or replicated in AWS WAF. Since AWS WAF integrates with Amazon CloudFront CDN, this enables earlier detection, minimizing overall traffic and impact. AWS WAF provides centralized control, automated administration, and real-time metrics.

Additionally, AWS Shield is a managed Distributed Denial of Service (DDoS) protection service that safeguards applications running on AWS. AWS Shield provides always-on detection and automatic inline mitigations that minimize application downtime and latency, so there is no need to engage AWS Support to benefit from DDoS protection. There are two tiers of AWS Shield: Standard and Advanced. All AWS customers benefit from the automatic protections of AWS Shield Standard, at no additional charge.

Data Protection

Organizations should encrypt data at rest and in transit. AEM provides SSL wizard to easily configure SSL certificates. AWS data protection services provide encryption and key management and threat detection that continuously monitors and protects your AWS infrastructure. For example, AWS Certificate Manager can provision, manage, and deploy public and private SSL/TLS certificates; AWS KMS can help with Key storage and management; and Amazon Macie can discover and protect your sensitive data at scale.

Access Control

AWS Identity & Access Management (IAM) helps securely manage access to AWS services and resources. In addition, AWS provides identity services to connect your on-prem directory service or use AWS Directory Service as a managed Microsoft Active Directory to provide access to AEM infrastructure as needed within your organization.

Detection, Audit, Monitoring, and Logging

Amazon GuardDuty is a threat-detection service that continuously monitors for malicious activity and unauthorized behavior to protect your AWS accounts and workloads. With AWS Security Hub, you have a single place that aggregates, organizes, and prioritizes your security alerts, or findings, from multiple AWS services,

such as Amazon GuardDuty, Amazon Inspector, and Amazon Macie, as well as from APN Partner solutions.

AWS also provides audit tools such as AWS Trusted Advisor, which inspects your AWS environment and makes recommendations for cost saving, improving system performance and reliability, and security. Amazon Inspector automatically assesses applications for vulnerabilities or deviations from best practices. After performing an assessment, Amazon Inspector produces a detailed report with prioritized steps for remediation. This can support system management and gives security professionals the necessary visibility into vulnerabilities that need to be fixed. In addition to Amazon Inspector, you can use other third-party products such as Burp Suite or Qualys SSL Test (for certificate validation.

Finally, having an audit log of all API actions and configuration changes can be useful in determining what changed and who changed it. AWS CloudTrail and AWS Config provide you with the capability to capture extensive audit logs. We recommend that you enable these services in your hosting environment.

Incident Response

AWS provides services such as AWS Lambda and AWS Config Rules which can evaluate whether your AWS resources comply with your desired settings and set them back into compliance or notify you. Amazon Detective is another service that simplifies the process of investigating security findings and identifying the root cause. Amazon Detective analyzes events from multiple data sources such as VPC Flow Logs, AWS CloudTrail logs, and Amazon GuardDuty findings, and automatically creates a graph model that provides you with a unified, interactive view of your resources, users, and the interactions between them over time.

Compliance and GovCloud

The AWS GovCloud (US) gives government customers and their partners the flexibility to architect secure cloud solutions that comply with many compliance programs (FedRAMP High, FISMA, DoD SRG, ITAR, and CJIS, to name a few). AWS GovCloud (US-East) and (US-West) Regions are operated by employees who are U.S. citizens on U.S. soil. AWS GovCloud (US) is only accessible to U.S. entities and root account holders who pass a screening process.

Service mapping to compliance programs is detailed on the AWS Services in Scope by Compliance Program page.

Digital Asset Management

AEM includes a Digital Asset Management (DAM) solution called AEM Assets. AEM assets enables your enterprise users to manage and distribute digital assets such as images, videos, documents, audio clips, 3D files, and rich media.

When planning for your AWS architecture, you should evaluate the potential use of the AEM Assets solution as part of your planning. With AEM Assets, the number of large files usually increases and often involves resource intensive processes such as image transformations and renditions. Various architecture best practices should be considered depending on the scenario, and they are described in detail in Best Practices for Assets.

Automated Deployment

AWS provides API access to all AWS services, and Adobe does this for AEM as well. Many of the various commands to deploy code or content, or to create backups, can be invoked through an HTTP service interface. This allows for a very clean organization of the continuous integration and deployment process with the use of Jenkins as a central hub, invoking AEM functionality through CURL or similar commands.

Jenkins can support manual, scheduled, and triggered deployments, and can be the central point for your AEM on AWS deployment. If necessary, you can enable additional automation using Jenkins with AWS CodeBuild and AWS CodeDeploy, enabling the creation of a complete environment from the Jenkins console. Refer to Set up a Jenkins Build Server on AWS to set up Jenkins.

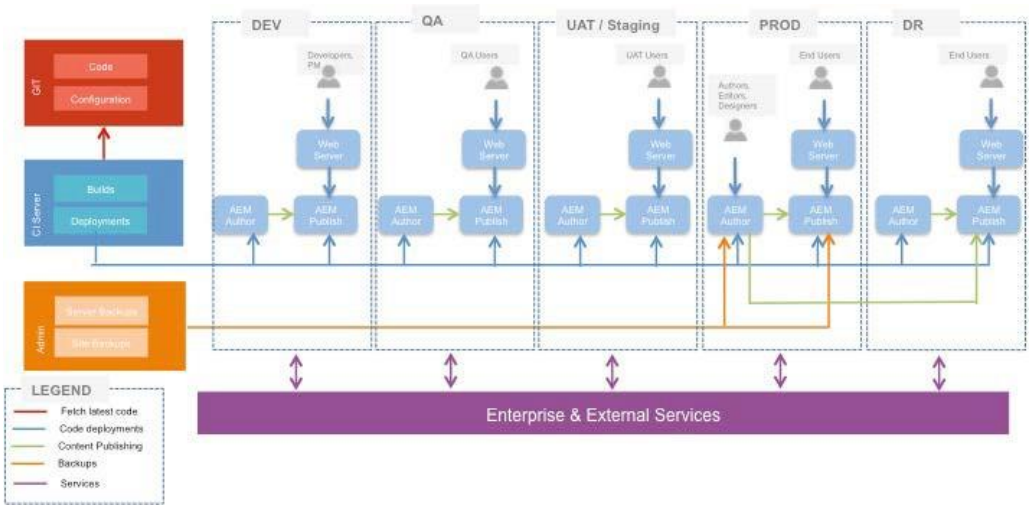

Figure 7 – Example CI Setup for an AEM Jenkins Architecture

Automated Operations

One of the key benefits of running AEM on AWS is the streamlined AEM Operations process. To provision instances, AWS CloudFormation or AWS OpsWorks can be leveraged to fully automate the deployment process, from setting up the architecture to provisioning the necessary instances. Using the AWS CloudFormation embedded stacks functionality, scripts can be organized to support the different architectures outlined in the earlier sections. Also, AEM OpenCloud manager provides automated operations functionality out of the box with little effort.

When using AEM's Tar Storage, repository content is stored on the file system. To create an AEM backup, you must create a file system snapshot. You can make a file system snapshot on AWS through Amazon Data Lifecycle Manager Alternately, you can create a centralized back up plan using AWS Backup. You should use Amazon Data Lifecycle Manager when you want to automate the creation, retention, and deletion of EBS snapshots. You should use AWS Backup to manage and monitor backups across the AWS services you use, including EBS volumes, from a single place.

Lastly, review the best practices and checks (such as log file monitoring, AEM performance monitoring, and Replication Agent monitoring) outlined in the Monitoring and Maintaining AEM guide to ensure smooth operations of your AEM environment.

Additional AWS Services

You can use additional services and capabilities from both AWS and the AEM platform to add further value to your AEM deployment on AWS. With AEM, you can integrate with a variety of third-party services out-of-the box, as well as Amazon SNS for mobile notifications relating to changes to the AEM environment.

AEM offers tools to manage targeting within experiences delivered through the solution. Adobe also has complementary products (which integrate well with AEM) that further personalize and target the experience for customers. Combined with AWS services such as Amazon Personalize, Amazon Kinesis, and AWS Lambda, you can create a powerful targeting engine to deliver one-to-one personalization.

Conclusion

This paper presented the business and technology drivers for running AEM on AWS, along with the strategies and considerations. Running AEM on AWS provides a secure and scalable foundation for delivering great digital experiences for customers. As you prepare for your AEM migration to AWS, we recommend that you consider the guidance outlined in this document.

Contributors

Contributors to this document include:

- Anuj Ratra, Sr. Solutions Architect, Amazon Web Services
- Cliffano Subagio, Principal Engineer, Shine Solutions Group
- Michael Bloch, Senior DevOps Engineer, Shine Solutions Group
- Matthew Holloway, Manager, Solutions Architects, Amazon Web Services
- Pawan Agnihotri, Sr Mgr, Solution Architecture, Amazon Web Services
- Martin Jacobs, GVP, Technology, Razorfish

Further Reading

For additional information, see:

- Hosting Adobe Experience Manager on AWS Reference Architecture

Document Revisions

Date	Description
November 2020	Updated Reference Architecture for AEM 6.5. Added AEM OpenCloud framework as an alternative option.
July 2016	First publication