Migrating to Amazon OpenSearch Service

AWS Prescriptive Guidance

AWS Prescriptive Guidance: Migrating to Amazon OpenSearch Service

Table of Contents

Migrating to Amazon OpenSearch Service

Amazon Web Services (contributors)

August 2023 (document history)

For many customers, migrating self-managed Elasticsearch or OpenSearch deployments to Amazon OpenSearch Service is challenging. The common challenges are of workload assessment, capacity planning, and architectural optimization. There are also questions concerning how to meet all requirements of operational analytics applications from on-premises data centers in the Amazon Web Services (AWS) Cloud. This guide covers the overall journey of a migration to Amazon OpenSearch Service, and provides best practices that AWS experts have accumulated over time. The step-by-step instructions can help you conduct your migrations in an effective and efficient approach. This guide mainly covers Amazon OpenSearch Service provisioned domains and not Amazon OpenSearch Serverless collections.

Overview

OpenSearch is a distributed, open-source search and analytics suite used for a broad set of operational analytics use cases such as real-time application monitoring, log analytics, data observability, and application and product catalog search. OpenSearch provides low-latency search response. It also offers fast access to large volumes of data with an integrated open-source data visualization tool called *OpenSearch Dashboards*.

Amazon OpenSearch Service supports performing interactive log analytics, real-time application monitoring, website search, and more. Amazon OpenSearch Service offers the latest versions of OpenSearch and support for 19 versions of Elasticsearch (versions 1.5–7.10). It also provides visualization capabilities powered by OpenSearch Dashboards and Kibana (versions 1.5–7.10). Amazon OpenSearch Service currently has tens of thousands of active customers with hundreds of thousands of clusters processing hundreds of trillions of requests per month.

Managing OpenSearch or Elasticsearch clusters on premises or on cloud infrastructure is highly complex, expensive, and tedious work. To run these clusters, you must provision and maintain the infrastructure. The efforts include the following:

- Hardware procurement and setup
- Software installation

- Configuration, patching, and upgrading
- Reliability and availability considerations
- Performance and scalability considerations
- Security and compliance considerations, such as network isolation, fine-grained access control, encryptions, and compliance programs such as the following:
  - Federal Risk and Authorization Management Program(FedRAMP)
  - General Data Protection Regulation (GDPR)
  - Health Insurance Portability and Accountability Act (HIPAA)
  - International Organization for Standardization (ISO)
  - Payment Card Industry Data Security Standard (PCI DSS)
  - System and Organization Controls (SOC).

By comparison, Amazon OpenSearch Service manages these tasks for you. In this guide, you will learn approaches and best practices for migrating on-premises or self-managed Elasticsearch or OpenSearch to the fully managed Amazon OpenSearch Service.

Benefits of migrating to Amazon OpenSearch Service

Amazon OpenSearch Service helps with deployment and ongoing management tasks. It's cost-effective, and it provides scalability, which improves reliability. It also offers security and helps support your compliance needs.

Easier to deploy and manage

It's easier to deploy an OpenSearch cluster by using Amazon OpenSearch Service than it is to deploy a cluster by yourself. Amazon OpenSearch Service helps to manage tasks such as hardware provisioning, software installation and patching, failure recovery, backups, and monitoring. You don't need to have a dedicated team of OpenSearch experts to manage your clusters.

An OpenSearch cluster in Amazon OpenSearch Service is also called a domain. Amazon OpenSearch Service provides domain health monitoring through the Amazon CloudWatch service. You can set up alerts to be notified of any changes to your domains' health. AWS Support provides one-on-one technical support from experienced engineers. Customers with operational challenges or technical questions can contact AWS Support and receive personalized support with reliable response times.

Cost-effective

Amazon OpenSearch Service is cost-effective. It provides a full array of advanced capabilities without charging additional licensing fees. You can use capabilities such as enterprise-grade security, real-time alerting, cross-cluster search, automated index management, and anomaly detection at no extra cost. There are no charges for data transfers between Availability Zones, and hourly snapshots are provided at no extra cost.

With *UltraWarm*, you can run interactive analytics on up to three petabytes of log data while reducing the cost per GB by up to 90 percent compared with the hot storage tier. Furthermore, Amazon OpenSearch Service offers Reserved Instances that provide significant discounts compared with the standard On-Demand Instances. For more information, see Cost-conscious.

More scalable and reliable

With Amazon OpenSearch Service, you can store petabytes of data in a single domain. You can query data across multiple domains and analyze all your data in a single OpenSearch Dashboards

interface. Amazon OpenSearch Service is designed to be highly reliable, using Multi-Availability Zone (Multi-AZ) deployments so that you can replicate data between up to three Availability Zones in the same AWS Region. There is zero downtime when you make software updates and upgrades or scale your environment.

With the Multi-AZ with Standby feature, OpenSearch Service domains are resilient to potential infrastructure failures, such as a node or Availability Zone failure. This enables 99.99 percent availability and consistent performance for business-critical workloads. With Multi-AZ with Standby, clusters are resilient to infrastructure failures such as hardware or networking failures. This option provides improved reliability and the added benefit of simplifying cluster configuration and management by enforcing best practices and reducing complexity.

Secure and compliant

Amazon OpenSearch Service takes care of all security patches. It also offers network isolation through a virtual private cloud (VPC), fine-grained access control, and multi-tenant OpenSearch Dashboards support. You can encrypt your data at rest and in transit. To help you meet industry-specific and regulatory requirements, Amazon OpenSearch Service is HIPAA eligible, and it's compliant with the following standards:

- FedRAMP
- GDPR
- PCI DSS
- ISO
- SOC

For more information, see the Amazon OpenSearch Service documentation.

Migration journey

Depending on your current deployment, migrating to an Amazon OpenSearch Service can be a basic or complex procedure with multiple steps. In the following sections, you will explore migration approaches and key considerations in each step of the process. This includes the best practices based on our experience of helping many AWS customers migrate from existing tooling to Amazon OpenSearch Service. This section also discusses what constitutes an effective migration strategy.

A typical migration journey involves five stages:

1. Planning
2. Proof of concept (PoC)
3. Deployment
4. Data migration
5. Cutover

You might be migrating from a self-managed Elasticsearch or OpenSearch cluster or you might be migrating from another technology to Amazon OpenSearch Service. In most cases, the steps remain the same. The time you spend on each step will vary based on the complexity of your environment.

The migration journey starts with a careful planning activity, followed by a PoC exercise to ensure that the target environment meets your cost, security, performance, and migration goals. The PoC activity is followed by deploying the target environment and migrating the data to it. When you have confirmed that your data is synchronized between the current environment and the new environment, you can cut over to the new environment. After you cut over, you operate the environment following operational best practices. The following sections discuss each stage in detail.

Stage 1 – Planning

Migration starts with planning the target environment that you are going to build to meet your requirements. Planning involves looking at a set of focus areas, each of which will require careful consideration:

- Sizing
- Functionality
- Version dependencies
- Key performance indicators (KPIs) and business continuity
- Operations and security
- Training
- Data flow
- Deployment frameworks

These focus areas will help you make decisions that will form the migration strategy. They also help you achieve your migration goals by reducing the migration complexity and costs.

During the planning stage, it's also critical to assess your current environment and identify pain points that you want to address as part of this migration. These pain points can be around performance, security, reliability, speed of delivery, cost, or ease of operations. As you review the focus areas, consider what improvements you can make as part of the migration.

Sizing

Sizing helps you determine the right instance type, number of data nodes, and storage requirement for your target environment. We recommend that you size first by the storage and then by CPUs. If you're already using Elasticsearch or OpenSearch, the sizing will generally remain the same. However, you need to identify the instance type that is equivalent to your current environment. To help determine the right size, we recommend using the following guidelines.

Storage

Sizing your cluster starts with defining the storage requirements. Identify the raw storage that you need for your cluster. This is determined by assessing the data generated by your source system (for example, servers generating logs, or product catalog raw size). After you identify how much

raw data you have, use the following formula to calculate storage requirements. You can then use the result as a starting point for your PoC.

```
storage needed = (daily source data in bytes × 1.45) (number_of_replicas +
1) × number of days retained
```

The formula takes into consideration the following:

- The on-disk size of an index varies, but it's often 10 percent larger than the source data.
- Operating system overhead of 5 percent is reserved by Linux for system recovery and to safeguard against disk defragmentation problems.
- OpenSearch reserves 20 percent of the storage space of each instance for segment merges, logs, and other internal operations.
- We recommend keeping 10 percent additional storage to help minimize the impact of node failure and Availability Zone outages.

Combined, these overheads and reservations require 45 percent additional space based on the actual raw data in the source. That's why you multiply the source data by 1.45. Next, multiply this by number of copies of data (for example, one primary plus the number of replicas you will use). The replica count depends on your resiliency and throughput requirement. For an average use case, you start with one primary and one replica. Finally, multiply by the number of days that you want to retain data in a hot-storage tier.

Amazon OpenSearch Service offers hot, warm, and cold storage tiers. The warm storage tier uses UltraWarm storage. UltraWarm provides a cost-effective way to store large amounts of read-only data on Amazon OpenSearch Service. Standard data nodes use hot storage, which takes the form of instance stores or Amazon Elastic Block Store (Amazon EBS) volumes attached to each node. Hot storage provides the fastest possible performance for indexing and searching new data. UltraWarm nodes use Amazon Simple Storage Service (Amazon S3) as storage and a sophisticated caching solution to improve performance. For indexes that you are not actively writing to, or query less frequently, and do not have the same performance requirements, UltraWarm offers significantly lower costs per GiB of data. For more information about UltraWarm, see the AWS documentation.

When you create an OpenSearch Service domain and use hot storage, you might need to define the EBS volume size. It depends on your choice of instance type for the data nodes. You can use the same storage-requirement formula to determine the volume size for Amazon EBS backed instances. We recommend using gp3 volumes for latest-generation T3, R5, R6G, M5, M5g, C5, and C6g instance families. Using Amazon EBS gp3 volumes, you can provision performance

independent of storage capacity. Amazon EBS gp3 volumes also provide better baseline performance, at a 9.6 percent lower cost per GB than existing gp2 volumes on OpenSearch Service. With gp3, you also get denser storage on R5, R6g, M5, and M6g instance families, which can help you to further optimize your costs. You can create EBS volumes up to the supported quota. For more information on quotas, see Amazon OpenSearch Service quotas.

For data nodes that have NVM Express (NVMe) drives, such as i3 and r6gd instances, the volume size is fixed, so EBS volumes are not an option.

Number of nodes and instance types

The number of nodes is based on the number of CPUs required to operate your workload. The number of CPUs is based on the shard count. An index in OpenSearch is made up of multiple shards. When you create an index, you specify the number of shards for the index. Therefore, you need to do the following:

1. Calculate the total shard count that you intend to store in the domain.
2. Determine the CPU.
3. Find the most cost-effective node type and count that gives you the required number of CPUs and storage.

This is usually a starting point. Run tests to determine that the estimate size is meeting your functional and nonfunctional requirements.

Determining the indexing strategy and shard count

After you know the storage requirements, you can decide how many indexes you need and identify the shard count for each. Generally, search use cases have one or a few indexes, each representing a searchable entity or a catalog. For log analytics use cases, an index can represent a daily or weekly log file. After you decide how many indexes, begin with the following scale guidance, and determine appropriate shard count:

- Search use cases – 10–30 GB/shard
- Log analytics use cases – 50 GB/shard

You can divide the total volume of data in a single index by the shard size you are aiming for in your use case. This will give you the number of shards for the index. Identifying the total number of shards will help you find the right instance types that suit your workload. The shards shouldn't

be too large or too numerous. Large shards can make it difficult for OpenSearch to recover from failure, but because each shard uses some amount of CPU and memory, having too many small shards can cause performance issues and out-of-memory errors. Moreover, imbalance in shard allocation to data nodes can lead to skewing. When you have indexes with multiple shards, try to make the shard count an even multiple of the data node count. This helps to ensure that shards are evenly distributed across data nodes, and prevents hot nodes. For example, if you have 12 primary shards, your data node count should be 2, 3, 4, 6, or 12. However, shard count is secondary to shard size—if you have 5 GiB of data, you should still use a single shard. Balancing replica shard count evenly across the Availability Zone also helps improve resilience.

CPU utilization

The next step is to identify how many CPUs you need for your workload. We recommend starting with a CPU count 1.5 times that of your active shards. An active shard is any shard for an index that is receiving substantial writes. Use the primary shard count to determine active shards for indexes that are receiving substantial read or write requests. For log analytics, only the current index is generally active. For search use cases, all primary shards will be considered as active shards. Although we recommend 1.5 CPU per active shard, this is highly workload-dependent. Be sure to test and monitor CPU utilization and scale accordingly.

A best practice for maintaining your CPU utilization is to make sure that the OpenSearch service domain has enough resources to perform its tasks. A cluster that has consistently high CPU utilization can degrade cluster stability. When your cluster is overloaded, OpenSearch Service will block incoming requests, which results in request rejections. This is to protect the domain from failing. General guidelines on the CPU usage will be about 60 percent average, 80 percent max CPU utilization. Occasional spikes of 100 percent are still acceptable and might not require scaling or reconfiguration.

Instance types

Amazon OpenSearch Service provides you with a choice of several instance types. You can choose the instance types that best fit your use case. Amazon OpenSearch Service supports the R, C, M, T, and I instance families. You choose an instance family based on the workload: memory optimized, compute optimized, or mixed. After you identify an instance family, choose the latest-generation instance type. Generally, we recommend Graviton and later generations because they are built to provide improved performance with lower costs compared with previous-generation instances.

Based on various testing that was performed for log analytics and search use cases, we recommend the following:

- For log analytics use cases , a general guideline is to begin with the R family of Graviton instances for data nodes. We recommend that you run tests, establish benchmarks for your requirements, and identify the appropriate instance size for your workload.
- For search use cases, we recommend using R and C family Graviton instances for data nodes, because search use cases require more CPU compared with log analytics use cases. For smaller workloads, you can use M family Graviton instances for both search and logs. I family instances offer NVMe drives and are used by customers with fast-indexing and low-latency search requirements.

The cluster is composed of data nodes and cluster manager nodes. Although dedicated master nodes don't process search and query requests, their size is highly correlated with the instance size and number of instances, indexes, and shards that they can manage. AWS documentation provides a matrix that recommends minimum dedicated cluster manager instance type.

AWS offers general purpose (M6g), compute optimized (C6g), and memory optimized (R6g and R6gd) for Amazon OpenSearch Service version 7.9 or later powered by AWS Graviton2 processors. These instances are built using custom silicon designed by Amazon. They are Amazon-designed hardware and software innovations that enable the delivery of efficient, flexible, and secure cloud services with isolated multi-tenancy, private networking, and fast local storage.

The Graviton2 instance family reduces indexing latency by up to 50 percent and improves query performance by up to 30 percent when compared with the previous generation Intel-based instances available in OpenSearch Service (M5, C5, R5).

Functionality

The functionality focus area helps you ensure that you do not lose any functionality when you migrate to a target Amazon OpenSearch Service environment. We recommend paying close attention to the following aspects:

- Current solution functionality
- Amazon OpenSearch Service functionality
- Packaged plugins

Current solution functionality

We recommend that you analyze your current solution and determine the features, plugins, and APIs that you use in the current technology stack (for example, Elasticsearch, OpenSearch, or another solution). Determine what functionality is critical to your business, what can be modified, and what can be dropped during the migration.

Amazon OpenSearch Service functionality

To ensure that required functionality is available after migration, we recommend that you perform an analysis of the latest OpenSearch version supported by Amazon OpenSearch Service, including the features it offers and the plugins that are available in Amazon OpenSearch Service. You want to confirm that the target platform supports the features you need (for example, index state management, which automates rolling over of the indexes, or machine learning features such as anomaly detection). Map the existing functionality of your current solution to features in Amazon OpenSearch Service that provide you with equivalent capability so that you can continue to support your workloads.

For more information about the functionality available within each supported version of the Elasticsearch or OpenSearch software, see the Amazon OpenSearch Service documentation.

Packaged plugins

Amazon OpenSearch Service supports a number of plugins that are part of the open-source OpenSearch project. If you are using any *licensed plugin* from the Elasticsearch suite that is part of X-Pack or otherwise, you might want to determine an equivalent plugin or native feature within the OpenSearch offerings. You might also want to capture that as a point to prove in the PoC stage.

OpenSearch has several plugins that provide enterprise-grade features equivalent to those licensed plugins. To determine the correct plugin and version for the target environment, review the OpenSearch Service documentation's list of plugins by versions. While Amazon OpenSearch Service supports a number of OpenSearch plugins out of the box, you might be using an open-source OpenSearch plugin that isn't currently available within Amazon OpenSearch Service. To request adding the plugin to the Amazon OpenSearch Service future roadmap, contact AWS.

Custom plugins

At the time of writing this guide, custom plugins are not supported. Therefore, you will need to consider alternate ways to deliver the custom plugin function and experience. If your solution uses custom plugins, analyze the functionality to determine whether you can port the custom plugins to the target environment using Amazon OpenSearch Service supported plugins or native features within OpenSearch. We recommend testing and proving all plugin choices during the PoC stage. Migration is a good time to evaluate current solution functionality to determine whether it is critical to your business.

Version dependencies

The version dependencies focus area helps you to build a roadmap of your migration journey through various versions to reach the latest version of Amazon OpenSearch Service. Consider the following key points:

- Selecting the engine version
- Upgrading to the latest version
- Version upgrade strategy
- Pre-upgrade checks

Selecting the engine version

It is very important to consider version dependencies carefully. Amazon OpenSearch Service supports a number of Elasticsearch versions and all major OpenSearch engine versions. (However, the latest version of OpenSearch can take a few weeks to be supported in Amazon OpenSearch Service from the date of release.) We recommend that you review the features supported by engine version in the Amazon OpenSearch Service documentation to identify the right version for your requirements. By choosing the same major (and closest minor) version, you can use the snapshot restore approach to migrate. This is often the most direct approach.

Upgrading to the latest OpenSearch Service version

While you might be able to operate an earlier version of Amazon OpenSearch Service, we highly recommend upgrading to the latest available version. This helps you take advantage of the performance improvements, reliability, cost savings, and many new features that are available in

the latest versions of the engine. Migration is a good opportunity to reduce the technical debt that can be incurred from running earlier versions of software.

Version upgrade strategy

If you decide that you want to upgrade to the latest version of the software during the migration, determine the steps and an upgrade strategy. Amazon OpenSearch Service documentation provides information on upgrade paths. It's important to understand the breaking changes between different versions. In some cases, the breaking changes might require you to plan for adjustments to your index modeling and design.

> **Note**
>
> **Note:** The *Multiple mapping types* functionality is available only in Elasticsearch versions 5.x and earlier. Indexes created in versions 6.x and later support only a single mapping type for each index. If you are using multiple mapping types, we recommend remodeling that data into multiple indexes.

In the case of a time-sensitive migration, consider a basic option where you perform an equivalent version migration (for example, 5.x to 5.x), and then upgrade the OpenSearch Service version at a later date. OpenSearch Service offers in-place upgrades for domains that run Elasticsearch versions 5.1 (if compatible) or later, and OpenSearch 1.0 or later. Perform a test to see if your indexes are compatible for in-place upgrades when you are running Elasticsearch version 5.x. This means you might be able to migrate to the equivalent version, and perform an in-place upgrade after you have made the necessary changes to make your indexes and other functionality compatible with the latest version. Review the upgrade domain documentation carefully.

Pre-upgrade checks

Amazon OpenSearch Service upgrade functionality can perform pre-upgrade checks by scanning the environment to determine issues that can block the upgrade. The upgrade doesn't proceed to the next step unless these checks succeed.

KPIs and business continuity

It's essential that during the migration you establish your business goals and key performance indicators (KPIs) to measure success. It's important to determine your goals at the beginning of

the migration process and establish a baseline for your current system so that you can determine measurable improvements. Common goals in customer journeys include the following:

- Improve operational agility.

  Under this goal, you can measure and compare your existing deployment with the target environment by using the following metrics:

  - Mean time to provision cluster.
  - Time to roll out the deployment to a new geography
  - Mean time to configure cluster security
  - Mean time to scale your environment (such as adding nodes and adding storage)
  - Mean time to detect slow-performing queries and mean time to repair them
  - Mean time to upgrade the software version
- Reduce total cost of ownership (TCO).

  To calculate your current TCO, you can use the following metrics:

  - Number of staff hours to build and operate the solution (development, DevOps, monitoring, scale, backup, restore)
  - License cost associated with the existing software
  - Data center costs (hardware procurement and refresh, electricity, cooling, space, racks, networking gears)
  - Staff hours to configure solution (software installations, networking)
  - Cost for compliance audits (HIPAA, PCI DSS, SOC, ISO, GDPR, FedRAMP)
  - Cost of configuring security (at-rest and in-transit encryption, configuring authentication and authorization, fine-grain access control)
  - Cost of retaining a large volume of warm and cold data
  - Cost of configuring high availability across Availability Zones
  - Cost of overprovisioning to avoid frequent hardware procurement or handling peak loads

  This list is not exhaustive.
- Monitor uptime and other service-level agreements (SLAs). SLAs that you can measure and improve by migrating to the new environment include the following:
  - Total uptime (historical uptime data of existing deployment compared with 99.9 percent SLA provided by Amazon OpenSearch Service)

- Failure recovery (recovery point objective and recovery time objective)
- Response time associated with various functions (for example, search and indexing)
- Number of concurrent users
- Replication time between different geographies and clusters.

As you migrate to Amazon OpenSearch Service, use an iterative process to verify whether you are meeting or exceeding those KPIs and whether you are achieving the desired outcomes.

Operational performance

A key area to look at in your current solution is performance metrics. Establish a benchmark, and determine improvements that you expect to achieve within your target environment. This includes your uptime SLA and latency requirements. This will help you establish and, in most cases, improve your current service levels. Usually, customers look at the following service level indicators

- Reads and writes per second
- Read and write latency
- Uptime percentage

When you architect your own SLAs, it's important to fully understand the Amazon OpenSearch Service - Service Level Agreement.

Process performance

To establish business continuity goals, it's important to assess your current process performance. Identify and review existing runbooks or standard operating procedures (SOPs) of the current platform, and determine areas where your team spends most of its time. Migration is a good opportunity to work on improving those areas so your team can focus on innovating, building business functionality, and improving the customer experience. You can identify pain points of your existing environment by reviewing the historical support or trouble ticket data to determine time spent by your support and development staff resolving these issues. Capturing the following metrics can help you measure improvements delivered by your target environment:

- Mean time to failure (MTTF) (uptime)
- Mean time between failures (MTBF)
- Mean time to detect (MTTD) a failure

- Mean time to repair (resolve) (MTTR)
- Number of support tickets received

Smooth transition to new services

To ensure business continuity of your services, it's important to carefully plan a seamless transition. Migration is a good time to modernize your application and the services associated with your search or log analytics platform. However, you want to plan a careful cutover strategy that will not impact your existing services. The cutover strategy section in this document provides information on how to plan a seamless cutover to the target environment.

Financial metrics

There could be many reasons to migrate to Amazon OpenSearch Service, but cost is generally a major factor. Understand the total cost of ownership (TCO) of the existing environment so you can measure the cost savings you get by moving to the managed service. You may start with the list of metrics that are listed under the *Reduce total cost of ownership* goal. AWS has published a cloud value benchmarking study that can help teams make a business case for migrating to the AWS Cloud. While the study is not specific to Amazon OpenSearch Service, it covers key value areas that are common across most cloud migrations, including migrating to Amazon OpenSearch Service.

In most cases, Amazon OpenSearch Service delivers lower TCO. When calculating TCO, it's critical to incorporate staffing cost. Understanding the time and cost that your engineers spend to maintain the current environment is an important factor. Many customers compare only the cost of storage, compute, and networking infrastructure with the cost of the managed service. However, that might not provide you with an accurate total cost of ownership. Amazon OpenSearch Service provides your team with operational efficiencies by managing tasks that otherwise had to be performed by your engineers. This includes the following tasks:

- Scaling a cluster by adding or removing nodes
- Patching
- Upgrading in-place
- Taking backups
- Configuring monitoring tools to capture logs and metrics

These activities are automated by the service, and AWS offers a production-level support team. This means that your staff can focus on activities that add direct value to your business.

Operations and security

When you migrate to Amazon OpenSearch Service, your operational activities will change. You will no longer be responsible for provisioning nodes, adding storage, installing and patching operating system, configuring and maintaining high availability, scaling, and other low-level activities. Instead, you can focus your attention on building your use cases and new user experiences.

Amazon OpenSearch Service offers logging, monitoring, and troubleshooting features that you will need to become familiar with to optimize your operational processes.

Runbooks and new processes

During the planning stage, identify existing processes that will need to be modified or eliminated. You can then add new operational processes that you might not have had bandwidth for in the past.

While Amazon OpenSearch Service takes away the undifferentiated heavy lifting, you will still need to ensure that your application is designed and monitored to deliver the best performance. You will need to configure monitoring and alerting for your domain so that you are fully aware of any health issues due to internal or external factors. You will need to schedule and initiate upgrades to the latest versions.

All such operational activities will require creating runbooks and modifying existing runbooks. To monitor infrastructure and to analyze operational metrics in Amazon OpenSearch Service, it's crucial to maintain runbooks. Runbooks ensure that you operate consistently according to your compliance and regulatory requirements. If you have not been using runbooks, it's a good time to consider doing so. Create processes to periodically run pre-planned steps to ensure remediation processes such as recovery from application crashes and unexpected failures are fully automated.

Support and ticketing system

To capture incidents associated with your deployments, we recommend planning and operating a ticketing system (you might already be doing so). You might need to train your support staff on how to create support tickets with AWS Support. We recommend streamlining the process of escalations during ticket triage.

The Operational excellence section later in this guide will provide you with links to a number of best practices and areas that you may need to consider in your runbooks and build processes around.

Security

At AWS, security is the top priority. Amazon OpenSearch Service provides multi-layer security. The service takes care of all security patches and offers network isolation through VPC, fine-grained access control, and multi-tenant support. Your data is encrypted at rest using keys that you create and control through AWS Key Management Service (AWS KMS). The node-to-node encryption capability provides Transport Layer Security (TLS) for all communications between instances in a domain. Amazon OpenSearch Service is also HIPAA eligible, and compliant with PCI DSS, SOC, ISO, and FedRAMP standards to help you meet industry-specific or regulatory requirements.

During the planning stage, identify the people and processes that interact with the domain, choose a network topology, and plan for authentication and authorization for each principal. Depending on your organizational security and compliance requirements, you can use multiple security features to create an environment that meets your business needs. In addition, consider the following factors:

- **VPC** – You can configure Amazon OpenSearch Service within a virtual private cloud (VPC) on AWS. This is the recommended configuration. We do not recommend creating a domain with a public endpoint. Plan to create the necessary network architecture to allow your client applications and users to access the target environment.
- **Authentication** – Amazon OpenSearch Service supports multiple ways to authenticate a user or software client. It supports Amazon Cognito or SAML authentication with your existing identity provider to access OpenSearch Dashboards. It also offers integration with IAM identities, and basic HTTP authentication using an internal user database. You should plan to configure and test an appropriate option for authentication. For more information, see the OpenSearch Service security documentation.
- **Authorization** – We recommend that you follow the principle of least privilege in configuring access to the service. Amazon OpenSearch Service provides fine-grained access control to help you configure access at document, row, and column levels.

Familiarize yourself with the security features and test them during the PoC stage.

Training

When you start your migration journey to AWS, your software development, operations, support, and security teams need to be equipped with knowledge of Amazon OpenSearch Service. Consider all teams that interact with your solution. When you are migrating from an Elasticsearch or an OpenSearch environment, most of the knowledge can be carried over. Provide training to the following teams:

- **Software development team** – Educate your software development team on the APIs and features, such as mechanisms for configuring data ingestion.
- **Operations team** – Train your operations team on how to interact with Amazon OpenSearch Service domains, monitor operational metrics, and access logs using Amazon CloudWatch. Team members should learn how to set up automated alarms to warn when OpenSearch Service domains need attention. If you are migrating from an existing toolset that you use on premises, such as Splunk, identify the monitoring options in Amazon OpenSearch Service that can provide similar visibility into your workloads.
- **Support team** – Educate your support team on how to implement runbooks that involve OpenSearch Service resources. You might want to update runbooks and event management procedures to use AWS Support services.
- **Security team** – Educate your security team on how to configure fine-grained access control and how to integrate with existing identity providers (IDPs).

Training options

AWS Training and Certification provides both digital and classroom training for beginner to professional level on cloud skills that are required to build and operate solutions on AWS. The content is created by experts at AWS and updated regularly. You have multiple training options.

You can work with your AWS account team to help you identify an appropriate resource. Following are some of the resources that you can use to help upskill your teams on Amazon OpenSearch Service:

- Immersion days – AWS Solutions Architects can deliver Immersion days, which are hands-on workshops that are tailored to address use cases, common implementation patterns, and roadmap items that might be specifically related to use cases.
- Hands-on workshops – Teams can follow self-service workshops built by AWS experts.

- Whitepapers and guides – AWS whitepapers are a great way to expand your knowledge of the cloud. Authored by AWS and the AWS community, they provide in-depth content that often addresses specific customer situations.
- Blog posts – Written by AWS experts and customers, these blog posts discuss latest announcements, best practices, solutions, service features, customer use cases, and other topics.
- Best practices – Participate in online or conference talks, or in sessions run by AWS experts that help you understand best practices for Amazon OpenSearch Service.
- AWS Professional Services – The AWS Professional Services team can provide best practices and prescriptive advice. The team offers a training program to help IT professionals understand and accomplish successful migrations.

Data flow

The data flow focus area includes the following three areas:

- Data ingestion
- Data retention
- Data migration approach

Data ingestion

Data ingestion focuses on how to get data into your Amazon OpenSearch Service domain. A thorough understanding of the data sources and formats is paramount when choosing the right ingestion framework for OpenSearch.

There are many different ways to create or modernize your ingestion design. There are many open-source tools for building a self-managed ingestion pipeline. OpenSearch Service supports integration with Fluentd, Logstash, or OpenSearch Data Prepper. These tools are popular with most log analytics solutions developers. You can deploy these tools on an Amazon EC2 instance, on Amazon Elastic Kubernetes Service (Amazon EKS), or on premises. Both Logstash and Fluentd support Amazon OpenSearch Service domains as an output destination. However, this will require you to maintain, patch, test, and keep the Fluentd or Logstash software versions up to date.

To reduce your operational overhead, you can use one of the AWS managed services that support integration with Amazon OpenSearch Service. For example, Amazon OpenSearch Ingestion is a fully managed, serverless data collector that delivers real-time log, metric, and trace data to

Amazon OpenSearch Service domains. With OpenSearch Ingestion, you no longer need to use third-party solutions such as Logstash or Jaeger to ingest data into your OpenSearch Service domains. You configure your data producers to send data to OpenSearch Ingestion. Then, it automatically delivers the data to the domain or collection that you specify. You can also configure OpenSearch Ingestion to transform your data before delivering it.

Another option is Amazon Data Firehose, which is a fully managed service that helps build a serverless ingestion pipeline. Firehose provides a secure way to ingest, transform, and deliver streaming data to Amazon OpenSearch Service domains. It can automatically scale to match the throughput of your data, and it requires no ongoing administration. Firehose can also transform incoming records by using AWS Lambda, compress, and batch the data before loading it into your OpenSearch Service domain.

With a managed service, you can retire your existing data ingestion pipeline, or you can augment your current setup to reduce operational overhead.

Migration planning is a good time to assess whether your current ingestion pipeline meets the needs of current and future use cases. If you are migrating from a self-managed Elasticsearch or OpenSearch cluster, your ingestion pipeline should support swapping the endpoints from the current cluster to the Amazon OpenSearch Service domain with minimal client library updates.

Data retention

When planning for data ingestion and storage, be sure to plan and agree on data retention. For log analytics use cases, it's critical that you have the right policies created within your domain to retire the historic data. When you are moving from an existing on-premises and cloud VM based architecture, you could be using a particular type of instance for all your data nodes. Data nodes have same CPU, memory, and storage profile. Most customers would configure high throughput storage to cater to their high-speed indexing requirement. This singular storage profile architecture is called *hot node only* architecture, or hot-only. Hot-only architecture couples storage with compute, which implies that you need to add compute nodes if your storage requirement increases.

To decouple storage from compute, Amazon OpenSearch Service offers the UltraWarm storage tier. UltraWarm provides a cost-effective way to store read-only data on Amazon OpenSearch Service by providing nodes that can accommodate a larger volume of data than traditional data nodes.

During planning, decide the data retention and processing requirement. To reduce the cost of your existing solution, take advantage of the UltraWarm tier. Identify the retention requirement for your data. Then create *Index state management policies* to move data from hot to warm or to delete the

data automatically from the domain when not needed. This also helps to ensure that your domain does not run out of storage.

Data migration approaches

During the planning stage, it's critical that you decide on a particular data migration approach. Your data migration approach dictates how you move the data that is in your current data store to the target store without any gaps. The procedural details for these approaches are covered in the Stage 4 – Data migration section, which is when you implement your approach.

This section covers different ways and patterns that you can use to migrate an Elasticsearch or OpenSearch cluster to Amazon OpenSearch Service. When choosing a pattern, consider the following list of factors (not exhaustive):

- Whether you want to copy data from an existing self-managed cluster or you are rebuilding from the original data source (log files, product catalog database)
- Version compatibility of the source Elasticsearch or OpenSearch cluster and target Amazon OpenSearch Service domain
- Applications and services dependent on the Elasticsearch or OpenSearch cluster
- The available window for the migration
- The volume of indexed data in your existing environment

Build from a snapshot

Snapshots are the most popular way to migrate from a self-managed Elasticsearch cluster to Amazon OpenSearch Service. Snapshots provide a way to back up your OpenSearch or Elasticsearch data by using a durable storage service such as Amazon S3. With this approach, you take snapshot of your current Elasticsearch or OpenSearch environment and restore it in the target Amazon OpenSearch Service environment. After restoring the snapshot, you can point your application to the new environment. This is a faster solution in the following situations:

- Your source and target are compatible.
- The existing cluster contains a large volume of indexed data, which can be time consuming to reindex.
- Your source data is not available for reindexing.

For additional considerations, see *Snapshot considerations* in the Stage 4 – Data migration section.

Build from the source

This approach implies that you are not going to move data from your current Elasticsearch or OpenSearch cluster. Instead, you reload the data directly from your log or product catalog source to the target Amazon OpenSearch Service domain. This is generally done with minor changes to existing data ingestion pipelines. In the log analytics use case, building from the source might also require reloading the historical logs from your sources to the new OpenSearch Service environment. For search use cases, it might require that you reload your full product catalog and content to the new Amazon OpenSearch Service domain. This approach works well in the following scenarios:

- Your source and target environment versions are not compatible for snapshot restore.
- You want to change your data model in the target environment as part of the migration.
- You want to jump to the most recent version of Amazon OpenSearch Service to avoid rolling upgrades, and you want to address the breaking changes in one go. This can be a good idea if you are self-managing a relatively older version (5.x or earlier) of Elasticsearch.
- You may want to change your indexing strategy. For example, instead of rolling over every day, you might roll over every month in the new environment.

For information about options for building from the source, see *2. Building from the source* in the Stage 4 – Data migration section.

Reindex remotely from an existing Elasticsearch or OpenSearch environment

This approach uses the remote reindex API from Amazon OpenSearch Service. Using remote reindex, you can copy data directly from your existing on-premises or cloud-based Elasticsearch or OpenSearch cluster to your Amazon OpenSearch Service domain. You can build automation that can keep the data synchronized between the two environment locations until you cut over to the target environment.

Use open-source data migration tools

There are multiple open source tools available to migrate data from your existing Elasticsearch environment to your target Amazon OpenSearch environment. One such example is the Logstash utility. You can use the Logstash utility to extract data from an Elasticsearch or OpenSearch cluster and copy it to the Amazon OpenSearch Service domain.

We recommend that you evaluate all your options and opt for the one that you are most comfortable with. To ensure that your selected approach is fool-proof, test all your tools and

automation during your PoC stage. For details and step-by-step guidance on how to implement these approaches, see the Stage 4 – Data migration section.

Deployment frameworks

Many modern teams use continuous integration and continuous delivery (CI/CD) practices and pipelines to automate the deployment of their solutions and infrastructure. If your team already uses CI/CD pipelines, you should be able to incorporate Amazon OpenSearch Service in your environment. If you are deploying manually in your current setup, consider building pipelines to automate repeatable work, reduce operational overhead, and reduce human errors.

You can deploy Amazon OpenSearch Service by using a variety of infrastructure as code (IaC) frameworks, including Terraform by HashiCorp, Chef, and Puppet. Terraform offers an OpenSearch module that you can use to create Amazon OpenSearch Service domains. In many cases, you can use your existing infrastructure deployment pipeline and point the search engine module to the Amazon OpenSearch Service module.

If you are thinking about building pipelines from the ground up, or if you want to use AWS native services, AWS provides several CI/CD tooling and service options. These include the following:

- AWS CodePipeline
- AWS CodeBuild
- AWS Cloud Development Kit (AWS CDK)
- AWS CloudFormation
- AWS CodeDeploy

You can use these services to automate infrastructure build, test, and deployment. Deploying your pipelines by using any of these cloud-native services has many advantages, including the following:

- Fully automated end-to-end (build, test, deployment) product releases
- Deployment to multiple environments (dev, test, pre-prod, prod)
- Integration with other AWS services
- The ability to modernize your deployment pipelines to automate deployments of Amazon OpenSearch Service across multiple environments

Stage 2 – Proof of concept

When performing a migration, it's critical to prove whether the target state solution will work as required. We strongly recommend running a proof-of-concept (PoC) exercise. This section focuses on the various aspects to factor in while running a PoC:

- Defining entry and exit criteria
- Securing funding
- Automating
- Thorough testing
- PoC stages
- Failure simulation

Defining entry and exit criteria

Having clear entry and exit criteria is key to a successful PoC exercise. When you define your entry criteria, consider the following:

- Use case definition
- Access to environments
- Familiarity with various services
- Associated training requirements

Similarly, define exit criteria that you can use to evaluate the PoC outcome, including the following:

- Functionality
- Performance requirements
- Security implementations PoC

Securing funding

Based on the PoC criteria definition, secure funding for the PoC. Ensure that you have performed the right sizing and considered all costs associated. If you are migrating from on premises to AWS, include the cost associated with migrating your frameworks over to the AWS Cloud from on

premises. If you're an existing AWS customer, work with your AWS account manager to understand whether you qualify for credits that can be used for the migration to Amazon OpenSearch Service.

Automating

Identify where automation can be done, and plan for a dedicated track to automate and time-box the testing. Automated deployment and testing helps you to rinse, repeat, test, and validate at a rapid pace and without human-introduced errors.

By time-boxing a test, you can ensure you deliver on time and can pivot to other activities if challenges arise. For example, if your performance tests are taking longer than the estimated time, you can pause that activity. You can then move to other tests and validation activities while your developers fix the issues. You can come back to the performance tests after the issues are resolved. Benchmark your existing solution performance, and create automated performance tests that can validate the effect of your configuration changes during the PoC.

Thorough testing

Test all portions of the stack by making sure that you perform the required validations for the different layers, such as ingestion pipelines and query mechanisms, that integrate with your Amazon OpenSearch Service domain. This will help you validate the end-to-end solution implementation.

Presentation layer

In the presentation layer, be sure to run a PoC exercise that includes the following activities:

- **Authenticate** – Validate the planned mechanisms for authenticating your users.
- **Authorize** – Identify the authorization mechanisms that you want to follow, and validate that they are working as expected.
- **Query** – What are the most common use cases that you will encounter in production? What are some edge-case scenarios that are critical to your business? Identify these patterns, and validate them during the PoC.
- **Render** – Is the data being rendered accurately and appropriately for various users across use cases? For log analytics use cases, you might want to build and test the dashboard on OpenSearch Dashboards or Kibana, depending on the target version, to confirm that it meets your requirements.

Ingestion layer

In the ingestion layer, be sure to evaluate various components such as collection, buffering, aggregation, and storage:

- **Collection** – For log analytics use cases, validate whether all the data that you are logging is being collected. For search use cases, identify the sources that feed the data and perform validations on completeness and correctness of data to make sure that the collection phase has been executed successfully.
- **Buffer** – If you have a spike in traffic, you might want to make sure that you are buffering the data that is getting ingested. There are various ways to create a buffering design. For example, you can collect data in Amazon Data Firehose, or you can use Amazon S3 storage as a buffer.
- **Aggregation** – Validate any aggregation of data, such as bulk API usage, that you perform during ingestion.
- **Storage** – Validate whether the storage is able to optimally handle the ingestion that you are performing.

PoC stages

We recommend that you use the following stages to implement your PoC and validate the outcome. Don't be afraid to iterate through these PoC phases and adjust the plan PoC even though you invested time in planning beforehand.

- **Functional testing and load testing** – Ensure that all levels are being thoroughly tested. Simulate failures in all portions of the stack. For example, if you have a cluster with two large nodes and one of them goes down, the other node must take up all the traffic on your cluster. In such a scenario, having a higher number of smaller nodes can result in a smoother recovery from a node failure. Test your workloads at peak loads and above to make sure that the performance is not impacted in such scenarios. During testing, raise issues early so that any potential issues are being evaluated by various stakeholders at the right time.
- **Verify KPIs and tune** – During the PoC, ensure that you are meeting the KPIs and business outcomes that you defined in your PoC exit criteria. Tune the configurations in such a way that they are meeting the KPIs.
- **Automate and deploy** – Automation and monitoring are the other key aspects to focus on during the PoC testing. Refine your automation steps, and validate them along with detailed monitoring to give all the stakeholders enough information to confidently evaluate the

outcomes of the PoC. Document all the steps, and create a runbook that you can reuse for the production migration.

Failure simulation

We highly recommend that you simulate a failure scenario and validate whether your design offers the resilience and fault tolerance required to meet your user requirements. You might want to simulate a failure of a data node to see if your cluster has enough resources to handle the recovery gracefully. To check whether your domain could be overwhelmed with large volume ingestion, you can test the buffering settings by simulating a sudden burst of logs from some of your sources. Validate that your design does not exceed any quotas when you scale to a production deployment. For more information, see the Amazon OpenSearch Service documentation on service quotas.

Stage 3 – Deployment

By the time you reach the deployment stage, you have completed your PoC, and you have a good idea of how to deploy your target environment to production. Keep in mind the following considerations:

- **Validate automation** – During the deployment, run the automation that you created during the PoC, and validate that it's working as expected. Also validate that your CI/CD automation is working as expected when you make configuration code changes.
- **Verify security** – It's critical to verify that all your security configurations are working as expected and that your data is secure. Confirm that the solution is vetted against your company's security standards, such as identity provider integration, and that your key users are able to log in and access data that they are authorized to access.
- **Monitoring** – Make sure that you have tested your monitoring configurations and have set up the recommended alerts. Monitor key metrics such as CPU, memory, disks, JVMs, and shard allocations. To give you the insights about the health of your Amazon OpenSearch Service domain and associated integrations, you can build a dashboard in Amazon CloudWatch. You can verify that your operations support team has access to the dashboard. The Operational excellence section provides links to useful tips for setting up a highly performant, resilient OpenSearch Service domain.
- **Exercise alarms** – Make sure that you test all your alarms. If you are using Amazon CloudWatch or an alerting plugin, validate that all integrations, such as Amazon Simple Notification Service (Amazon SNS) or Slack, work as expected. Simulate alerts to validate that the alerts are delivered correctly to the destination channel. Confirm that the alert text provides helpful information. For example, the alert could provide a link to the associated runbook for your support team to implement an associated remediation process.

Stage 4 – Data migration

Now that your target environment is ready, you can implement the data migration strategy that you chose during the planning stage.

This section covers implementation steps for the four different patterns:

- Building from a snapshot
- Building from the source
- Remote reindexing
- Using Logstash

1. Building from a snapshot

When you use the snapshot-restore approach, you copy data from the source Elasticsearch or OpenSearch cluster to target Amazon OpenSearch Service domain.

Broadly, the snapshot-restore process consists of the following steps:

1. Take a snapshot of the necessary data (indexes) from the existing cluster, and upload the snapshot to an S3 bucket.
2. Create an Amazon OpenSearch Service domain.
3. Give Amazon OpenSearch Service permissions to access the bucket, and give your user account permissions to work with snapshots. Create a snapshot repository and point that to your bucket.
4. Restore the snapshot on the Amazon OpenSearch Service domain.
5. Point your client applications to the Amazon OpenSearch Service domain.
6. Create Index State Management (ISM) policies for configuring retention (optional).

Snapshots are incremental. Therefore, a snapshot can be run and restored incrementally. By using snapshots, you can extract data in bulk as files on a storage system (for example, Amazon S3). You can then load these files in the target environment by using the `_restore` API operation. This eliminates the need for reindexing, which is time consuming, and it also reduces network traffic.

Snapshot considerations

When using the snapshot-restore approach, consider the following:

- You can't search or reindex while an index is being restored. However, you can search and reindex an index while the snapshot is being taken.
- The source and target Elasticsearch or OpenSearch versions must be compatible. A snapshot of an index that was created in:
  - 5.x can be restored to 6.x
  - 2.x can be restored to 5.x
  - 1.x can be restored to 2.x
- Because this is a point-in-time restoration of the Elasticsearch or OpenSearch snapshot, subsequent changes in the source cluster won't be replicated to the target Amazon OpenSearch Service domain. You can stop ingestion of the data into the source Elasticsearch or OpenSearch cluster until the restoration is done, or you can repeat the snapshot restore process a few times. Because the snapshot is incremental, only the changes will be copied and restored in the target environment in less time than the first restore. After restoration is successfully finished, you point the ingestion applications to the Amazon OpenSearch Service domain.
- Taking a snapshot includes, by default, a snapshot of the cluster state and all indexes. When migrating from Elasticsearch, you might need to create equivalent index lifecycle policies in the target environment using the ISM feature in OpenSearch. Elasticsearch Index Lifecycle Management (ILM) is not supported in Amazon OpenSearch Service.
- You can't restore a snapshot to an earlier version of Elasticsearch or OpenSearch. For example, you can't restore a snapshot of version 7.10 to 7.9. Similarly, you can't restore snapshots from Elasticsearch 7.11 or later to an Amazon OpenSearch Service domain. If you have migrated your self-managed Elasticsearch environment to version 7.11 or later, you can use Logstash to load data from the Elasticsearch cluster and write it to the OpenSearch domain.
- You export a snapshot to a designated storage location called a repository. Elasticsearch or OpenSearch creates a number of files in the repository. You can't modify or delete these files. Doing so might create inconsistencies or cause the restore process to fail.

2. Building from the source

As described earlier, building from the source is the approach where you do not migrate data from the current Elasticsearch or OpenSearch environment. Instead, you build indexes in the target domain directly from your log, or product-catalog data source or content source.

Two options are available for building from the source. The option you choose depends on the data type of your data:

- Using AWS Database Migration Service – If the source of your data is a relational database management system (RDBMS) and the source is supported by AWS Database Migration Service (AWS DMS), you can use AWS DMS to copy data from your data source to your target Amazon OpenSearch Service domain. AWS DMS supports full load and change data capture (CDC) options. In the full load option, the AWS DMS task copies all data from source database table to a target OpenSearch index. You can use default mapping or provide custom mapping configurations. In the CDC option, AWS DMS first makes a full copy of the source table records into a target OpenSearch index. Then it captures changed data (updates and inserts) and copies it to the OpenSearch index. For more information, see the blog posts Introducing Amazon Elasticsearch Service as a target in AWS Database Migration Service and Scale Amazon Elasticsearch Service for AWS Database Migration Service migrations.
- Building from the document source – If your data source is not an RDBMS or it is not supported by AWS DMS, you might have to create a custom solution using open-source tools or a combination of open-source tools and AWS services. You must convert your source data to JSON documents before it can be loaded in OpenSearch. If you already have pipelines set up from your source to your current Elasticsearch or OpenSearch environment, you can point those data pipelines to OpenSearch with appropriate changes in client libraries and (if required) data model changes in indexes in the Amazon OpenSearch Service domain. When building indexes from the source, keep in mind the following considerations:
  - The location of the documents – The documents could already be available in the AWS Cloud, in object storage such as Amazon S3, or they might be stored in an on-premises storage location such as a file system.
  - The format of the documents – The documents could already be in JSON format, ready to be ingested into the Amazon OpenSearch Service domain, or they might need to be cleansed, processed, and formatted into JSON before they can be ingested into the Amazon OpenSearch Service domain.

Building from the source involves the following high-level steps:

1. Define index mapping and settings in the Amazon OpenSearch Service domain.
2. Extract data from the document source and copy it into an object storage location such as Amazon S3. You can use an open source tool (for example, Logstash), an AWS service client (for example, Amazon Kinesis Agent), a third-party commercial tool, or a custom program.

3. Configure an open-source tool (for example, Logstash or Fluent Bit) or a native AWS service (for example, AWS Lambda or AWS DMS) to convert data into JSON documents and load it periodically or continuously from the object store to the Amazon OpenSearch Service domain.

For more information, see Loading streaming data into Amazon OpenSearch Service.

3. Remote reindexing

In this case, the indexes of the source self-managed Elasticsearch or OpenSearch cluster are migrated into the Amazon OpenSearch Service domain using the reindex document API operation. You can use the reindex document API operation to create an index from an existing Elasticsearch or OpenSearch index. The existing index can be in the same cluster where you run the reindex operation, or it can be in a remote cluster. Amazon OpenSearch Service supports using the reindex document API operation with remote clusters. You can reindex from an index in a self-managed Elasticsearch to an index in Amazon OpenSearch Service.

Remote reindex supports Elasticsearch 1.5 and later for the remote Elasticsearch cluster and Amazon OpenSearch Service 6.7 and later for the local domain. For more information, see the blog post Migrate data into Amazon ES using remote reindex. The blog post refers to Amazon Elasticsearch, but the guidance applies to Amazon OpenSearch Service domains equally.

4. Using Logstash

Logstash is an open-source data processing tool that can collect data from the source, perform transformation or filtering, and send data to one or more destinations. To write data to the Amazon OpenSearch Service domain, Logstash provides the following plugins:

- logstash-input-elasticsearch
- logstash-input-opensearch
- logstash-output-opensearch

For more information, see Loading data into Amazon OpenSearch Service with Logstash and the OpenSearch blog post Introducing logstash-input-opensearch plugin for OpenSearch.

Stage 5 – Cutover

This stage discusses various approaches that you can employ to cut over from your current Elasticsearch or OpenSearch environment to the target Amazon OpenSearch Service domain. Cutover can be done in two steps:

- Establish a data synchronization mechanism to keep the target environment synchronized with the source.
- Perform the swap from the current environment to the target environment with or without downtime.

Data synchronization

For any system receiving continuous data, data migration might require that you stop receiving new data during the migration and run the migration in a maintenance window (with possible downtime). If you can't afford downtime, you can capture changes after you have initiated the migration. You replay the changes on the target to keep it updated and synchronized with the source until you perform the cutover. The following sections discuss various ways that you can keep the source and target synchronized.

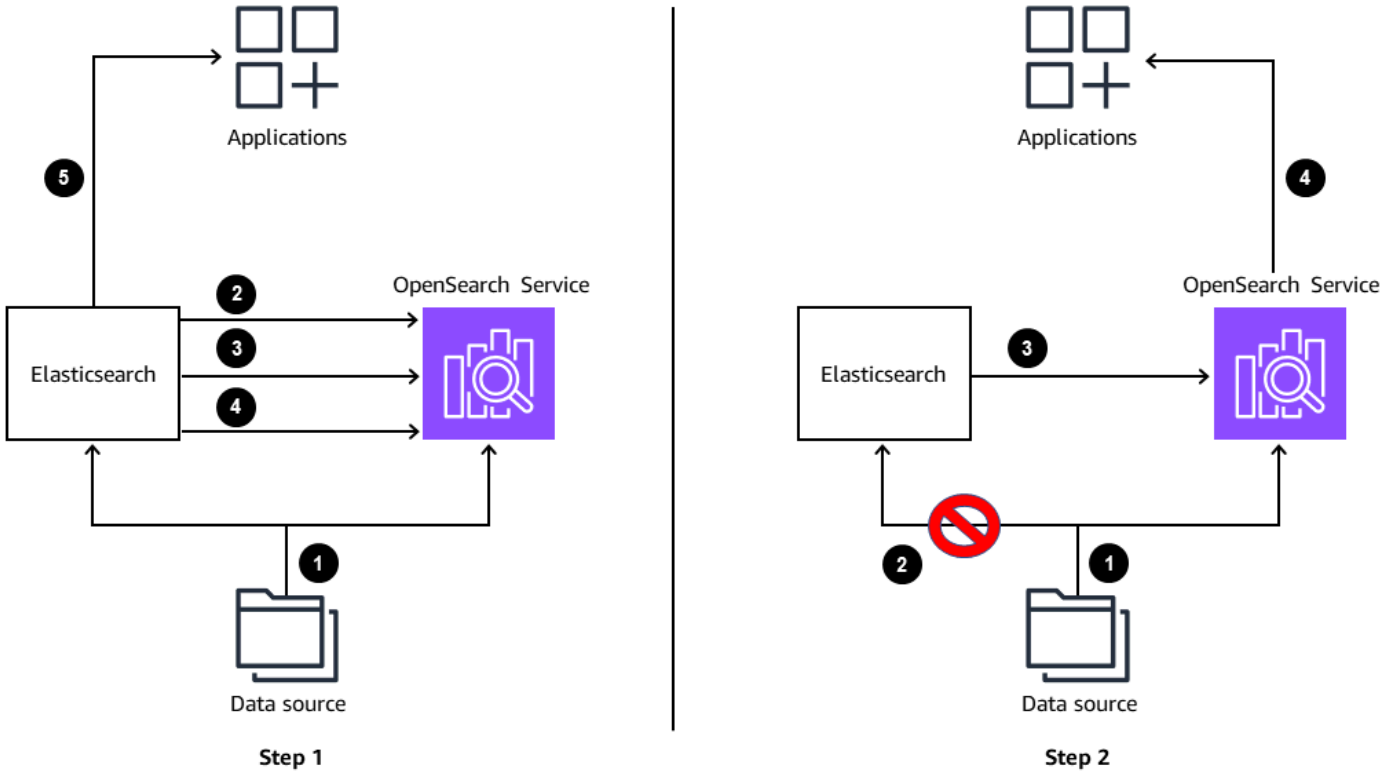
Log analytics workloads

For log analytics workloads, you can perform an update sync in the following ways:

- You can run two environments side by side until the retention period and run ingestion to both the current and target environments are complete. At some point in time, you decide to cut over and point your applications to the new environment. Sometimes, you can ingest new data from the log or document sources to both the existing cluster and the target OpenSearch Service environments. You can then backfill the older data in the target environment by copying it from the current environment. In all cases, you must make sure your data doesn't have any gaps that will impact your users.
- Before the data migration, you can decide to pause your ingestion to the existing environment. However, this approach means your users might not be able to search the latest or changed data from your existing environment until your data migration is complete. After your data migration is complete, you can point your data ingestion to the target environment and switch your applications and clients over to the target environment. This means no new data will be available until migration is complete. However, the system will remain available for search. You

should have the means to hold source logs and data in your source until the new environment is available.

- You can continue to use the current log analytics engine until your first pass of data is migrated. Then you backfill the remaining data that has been produced since the first pass was initiated. Assuming that the remaining data is much smaller than the first pass, you can pause ingestion while your remaining data is synchronized, because the synchronization might take only a few minutes or a few hours. You can also perform a few passes using this approach until your synchronization window becomes small enough to pause ingestion from the source to the target environment and cut over to the target environment without impacting your users. The following diagram shows using incremental snapshot and restore to update or sync data.



Step 1

1. Data flows from the source through the data ingestion pipeline to the current Elasticsearch environment and the Amazon OpenSearch Service domain.
2. The first pass takes the longest time to move from Elasticsearch to the Amazon OpenSearch Service domain.
3. The first update or sync pass takes less time.
4. The second update or sync pass takes the least amount of time.

5. Data continues to flow from Elasticsearch to the applications.

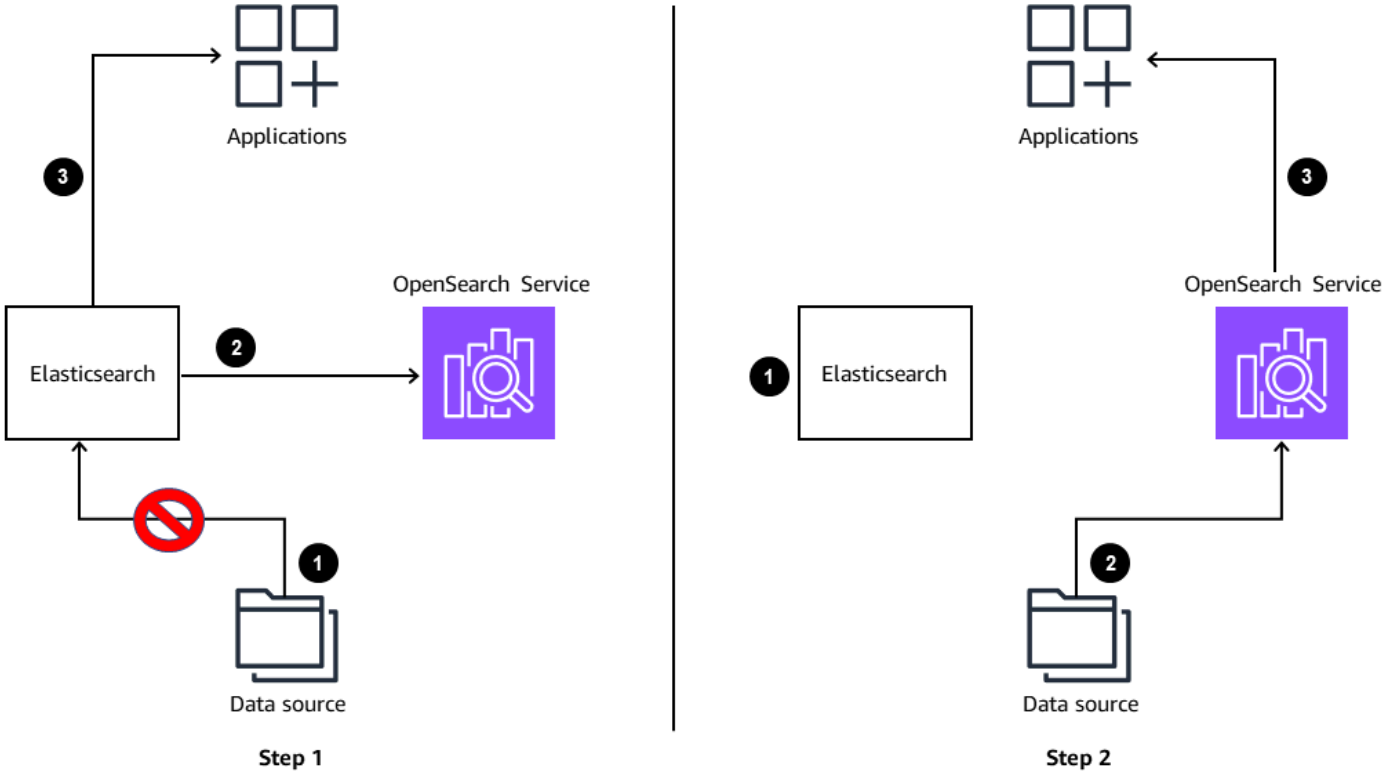
Step 2

1. Data flows from the source through the data ingestion pipeline to the OpenSearch Service domain.
2. Ingestion to the current Elasticsearch environment is stopped.
3. The final update or sync pass takes the least amount of time.
4. Data flows from OpenSearch Service to the applications.

Search workloads

In the three approaches previously discussed, you must ensure that all data on your target is up to date before you perform the cutover. For search workloads, you can consider the following suggestions for updating or syncing:

- For search workloads, typically you pause the ingestion from the source to the current environment. You copy all your data from the current environment to the target environment, and you put in place a change data capture (CDC) mechanism that can determine what data has changed since the start of the migration. You then copy the changed data to the Amazon OpenSearch environment. In most cases, the search application's data ingestion pipelines already have a CDC mechanism built in, and it usually is a matter of pointing your pipeline to the new environment after the data is migrated from the current environment. The following diagram shows building an index entirely from the source for search use cases.



Step 1

1. Ingestion to the current Elasticsearch environment is paused.
2. Data is copied from ElasticSearch to the OpenSearch Service domain.
3. Data continues to flow from ElasticSearch to the applications.

Step 2

1. The Elasticsearch environment is no longer connected to the data source or the applications.
2. Change data capture (CDC) data is ingested in the pipeline and flows to the OpenSearch Service domain.
3. Data flows from the OpenSearch Service domain to the applications.

- Some search workloads require loading only full data from the source database or data source to the new OpenSearch Service environment. After the load is complete, the client applications can cut over to the new environment. This is the simplest way to achieve migration for search workloads.

Swap or cut over

The final step in the migration journey is swapping, or cutting over, to the new environment. It is one of the critical phases. At this point, you are ready to go live. You have the data synchronized and up to date, you have monitoring and alerts configured, your runbooks are up to date, and you are ready to cut over to the new environment. You must make sure that your ingestion is flowing normally and that the metrics from your new environment are healthy. During this stage, you plan and perform the cutting over of the client connections from your existing Elasticsearch or OpenSearch cluster to the new Amazon OpenSearch Service domain. Be mindful of any client library changes that might be required. At this point, you should have tested all your client functionality with Amazon OpenSearch Service in your lower environments to verify compatibility and performance.

If you have a client application that needs to point to the new environment, update the DNS entry from the old environment to the new environment. Then closely monitor the application behavior to ensure that your users are getting the right experience.

Generally, if you have followed the guidelines in this document, you will have a safe switchover. However, we recommend that you keep your source environment up to date so that it can act as a fallback in case you encounter any problems with the new environment. Some AWS customers continue to operate both environments for a few weeks after the swap before decommissioning the older environment. We recommend that you choose a strategy that aligns with your business continuity requirements.

Stage 6 – Operational excellence

Amazon OpenSearch Service documentation has a dedicated section on Operational best practices. Topics include the following:

- Monitoring and alerting
- Shard strategy
- Stability
- Performance
- Security
- Cost optimization
- Sizing Amazon OpenSearch Service domains
- Petabyte scale in Amazon OpenSearch Service
- Dedicated master nodes in Amazon OpenSearch Service
- Recommended CloudWatch alarms for Amazon OpenSearch Service

We recommend that you follow guidance provided in the documentation to operate your newly migrated environment.

Conclusion

Amazon OpenSearch Service takes away the undifferentiated heavy lifting that is required to develop and operate self-managed Elasticsearch or OpenSearch clusters. If you are considering a migration to Amazon OpenSearch Service, you can use the process outlined in this guide to plan and choose a migration strategy that works for your situation.

Migrations can be as basic as taking a snapshot from self-managed cluster and restoring it in Amazon OpenSearch Service domain, or they can be as involved as testing all existing functionality and integrations. This guide provides information that can be used by migration project teams to make sure they have covered all aspects of a migration and to build a robust implementation strategy.

Amazon OpenSearch Service documentation has a dedicated section on Operational best practices. We recommend that you follow guidance provided in the documentation to operate your newly migrated environment.

Resources

- Creating index snapshots in Amazon OpenSearch Service
- Use Amazon S3 to Store a Single Amazon OpenSearch Service Index (blog post)
- Elasticsearch snapshot and restore (Elasticsearch documentation)
- S3 Repository Plugin (Elasticsearch documentation)
- Elasticsearch Repository Settings: Recommended S3 Permissions (Elasticsearch documentation)
- Elasticsearch Client Settings (Elasticsearch documentation)

Contributors

Contributors

Contributors to this document include:

- Muhammad Ali, Principal OpenSearch Solutions Architect
- Gene Alpert, Sr. Specialist Technical Account Manager – Analytics
- Jon Handler, Sr. Principal Solutions Architect
- Prashant Agrawal, Sr. OpenSearch Specialist Solutions Architect
- Ina Felsheim, Sr. Product Marketing Manager
- Sung-il Kim, Sr. Analytics Solutions Architect
- Hajer Bouafif, OpenSearch Solutions Architect
- Kevin Fallis, Principal OpenSearch Specialist Solutions Architect
- Muthu Pitchaimani, Sr. OpenSearch Specialist Solutions Architect
- Kunal Kusoorkar, Mgr., OpenSearch Solutions Architect
- Imtiaz Sayed, Pr. Analytics Solutions Architect Technical Leader
- Soujanya Konka, Sr. Solutions Architect
- Marc Clark, Mgr., OpenSearch Specialist
- Bob Taylor, Sr. OpenSearch Specialist
- Aneesh Chandra PN, Principal Analytics Solutions Architect, Health Care and Life Sciences

Document history

The following table describes significant changes to this guide. If you want to be notified about future updates, you can subscribe to an RSS feed.

Change	Description	Date
Initial publication	—	August 28, 2023

AWS Prescriptive Guidance glossary

The following are commonly used terms in strategies, guides, and patterns provided by AWS Prescriptive Guidance. To suggest entries, please use the **Provide feedback** link at the end of the glossary.

Numbers

7 Rs

Seven common migration strategies for moving applications to the cloud. These strategies build upon the 5 Rs that Gartner identified in 2011 and consist of the following:

- Refactor/re-architect – Move an application and modify its architecture by taking full advantage of cloud-native features to improve agility, performance, and scalability. This typically involves porting the operating system and database. Example: Migrate your on-premises Oracle database to the Amazon Aurora PostgreSQL-Compatible Edition.
- Replatform (lift and reshape) – Move an application to the cloud, and introduce some level of optimization to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Amazon Relational Database Service (Amazon RDS) for Oracle in the AWS Cloud.
- Repurchase (drop and shop) – Switch to a different product, typically by moving from a traditional license to a SaaS model. Example: Migrate your customer relationship management (CRM) system to Salesforce.com.
- Rehost (lift and shift) – Move an application to the cloud without making any changes to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Oracle on an EC2 instance in the AWS Cloud.
- Relocate (hypervisor-level lift and shift) – Move infrastructure to the cloud without purchasing new hardware, rewriting applications, or modifying your existing operations. You migrate servers from an on-premises platform to a cloud service for the same platform. Example: Migrate a Microsoft Hyper-V application to AWS.
- Retain (revisit) – Keep applications in your source environment. These might include applications that require major refactoring, and you want to postpone that work until a later time, and legacy applications that you want to retain, because there's no business justification for migrating them.

- Retire – Decommission or remove applications that are no longer needed in your source environment.

A

ABAC

See attribute-based access control.

abstracted services

See managed services.

ACID

See atomicity, consistency, isolation, durability.

active-active migration

A database migration method in which the source and target databases are kept in sync (by using a bidirectional replication tool or dual write operations), and both databases handle transactions from connecting applications during migration. This method supports migration in small, controlled batches instead of requiring a one-time cutover. It's more flexible but requires more work than active-passive migration.

active-passive migration

A database migration method in which the source and target databases are kept in sync, but only the source database handles transactions from connecting applications while data is replicated to the target database. The target database doesn't accept any transactions during migration.

aggregate function

A SQL function that operates on a group of rows and calculates a single return value for the group. Examples of aggregate functions include `SUM` and `MAX`.

AI

See artificial intelligence.

AIOps

See artificial intelligence operations.

anonymization

The process of permanently deleting personal information in a dataset. Anonymization can help protect personal privacy. Anonymized data is no longer considered to be personal data.

anti-pattern

A frequently used solution for a recurring issue where the solution is counter-productive, ineffective, or less effective than an alternative.

application control

A security approach that allows the use of only approved applications in order to help protect a system from malware.

application portfolio

A collection of detailed information about each application used by an organization, including the cost to build and maintain the application, and its business value. This information is key to the portfolio discovery and analysis process and helps identify and prioritize the applications to be migrated, modernized, and optimized.

artificial intelligence (AI)

The field of computer science that is dedicated to using computing technologies to perform cognitive functions that are typically associated with humans, such as learning, solving problems, and recognizing patterns. For more information, see What is Artificial Intelligence?

artificial intelligence operations (AIOps)

The process of using machine learning techniques to solve operational problems, reduce operational incidents and human intervention, and increase service quality. For more information about how AIOps is used in the AWS migration strategy, see the operations integration guide.

asymmetric encryption

An encryption algorithm that uses a pair of keys, a public key for encryption and a private key for decryption. You can share the public key because it isn't used for decryption, but access to the private key should be highly restricted.

atomicity, consistency, isolation, durability (ACID)

A set of software properties that guarantee the data validity and operational reliability of a database, even in the case of errors, power failures, or other problems.

attribute-based access control (ABAC)

The practice of creating fine-grained permissions based on user attributes, such as department, job role, and team name. For more information, see ABAC for AWS in the AWS Identity and Access Management (IAM) documentation.

authoritative data source

A location where you store the primary version of data, which is considered to be the most reliable source of information. You can copy data from the authoritative data source to other locations for the purposes of processing or modifying the data, such as anonymizing, redacting, or pseudonymizing it.

Availability Zone

A distinct location within an AWS Region that is insulated from failures in other Availability Zones and provides inexpensive, low-latency network connectivity to other Availability Zones in the same Region.

AWS Cloud Adoption Framework (AWS CAF)

A framework of guidelines and best practices from AWS to help organizations develop an efficient and effective plan to move successfully to the cloud. AWS CAF organizes guidance into six focus areas called perspectives: business, people, governance, platform, security, and operations. The business, people, and governance perspectives focus on business skills and processes; the platform, security, and operations perspectives focus on technical skills and processes. For example, the people perspective targets stakeholders who handle human resources (HR), staffing functions, and people management. For this perspective, AWS CAF provides guidance for people development, training, and communications to help ready the organization for successful cloud adoption. For more information, see the AWS CAF website and the AWS CAF whitepaper.

AWS Workload Qualification Framework (AWS WQF)

A tool that evaluates database migration workloads, recommends migration strategies, and provides work estimates. AWS WQF is included with AWS Schema Conversion Tool (AWS SCT). It analyzes database schemas and code objects, application code, dependencies, and performance characteristics, and provides assessment reports.

B

bad bot

A bot that is intended to disrupt or cause harm to individuals or organizations.

BCP

See business continuity planning.

behavior graph

A unified, interactive view of resource behavior and interactions over time. You can use a behavior graph with Amazon Detective to examine failed logon attempts, suspicious API calls, and similar actions. For more information, see Data in a behavior graph in the Detective documentation.

big-endian system

A system that stores the most significant byte first. See also endianness.

binary classification

A process that predicts a binary outcome (one of two possible classes). For example, your ML model might need to predict problems such as “Is this email spam or not spam?" or "Is this product a book or a car?"

bloom filter

A probabilistic, memory-efficient data structure that is used to test whether an element is a member of a set.

blue/green deployment

A deployment strategy where you create two separate but identical environments. You run the current application version in one environment (blue) and the new application version in the other environment (green). This strategy helps you quickly roll back with minimal impact.

bot

A software application that runs automated tasks over the internet and simulates human activity or interaction. Some bots are useful or beneficial, such as web crawlers that index information on the internet. Some other bots, known as *bad bots*, are intended to disrupt or cause harm to individuals or organizations.

botnet

Networks of bots that are infected by malware and are under the control of a single party, known as a *bot herder* or *bot operator*. Botnets are the best-known mechanism to scale bots and their impact.

branch

A contained area of a code repository. The first branch created in a repository is the *main branch*. You can create a new branch from an existing branch, and you can then develop features or fix bugs in the new branch. A branch you create to build a feature is commonly referred to as a *feature branch*. When the feature is ready for release, you merge the feature branch back into the main branch. For more information, see About branches (GitHub documentation).

break-glass access

In exceptional circumstances and through an approved process, a quick means for a user to gain access to an AWS account that they don't typically have permissions to access. For more information, see the Implement break-glass procedures indicator in the AWS Well-Architected guidance.

brownfield strategy

The existing infrastructure in your environment. When adopting a brownfield strategy for a system architecture, you design the architecture around the constraints of the current systems and infrastructure. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

buffer cache

The memory area where the most frequently accessed data is stored.

business capability

What a business does to generate value (for example, sales, customer service, or marketing). Microservices architectures and development decisions can be driven by business capabilities. For more information, see the Organized around business capabilities section of the Running containerized microservices on AWS whitepaper.

business continuity planning (BCP)

A plan that addresses the potential impact of a disruptive event, such as a large-scale migration, on operations and enables a business to resume operations quickly.

C

CAF

See AWS Cloud Adoption Framework.

canary deployment

The slow and incremental release of a version to end users. When you are confident, you deploy the new version and replace the current version in its entirety.

CCoE

See Cloud Center of Excellence.

CDC

See change data capture.

change data capture (CDC)

The process of tracking changes to a data source, such as a database table, and recording metadata about the change. You can use CDC for various purposes, such as auditing or replicating changes in a target system to maintain synchronization.

chaos engineering

Intentionally introducing failures or disruptive events to test a system's resilience. You can use AWS Fault Injection Service (AWS FIS) to perform experiments that stress your AWS workloads and evaluate their response.

CI/CD

See continuous integration and continuous delivery.

classification

A categorization process that helps generate predictions. ML models for classification problems predict a discrete value. Discrete values are always distinct from one another. For example, a model might need to evaluate whether or not there is a car in an image.

client-side encryption

Encryption of data locally, before the target AWS service receives it.

Cloud Center of Excellence (CCoE)

A multi-disciplinary team that drives cloud adoption efforts across an organization, including developing cloud best practices, mobilizing resources, establishing migration timelines, and leading the organization through large-scale transformations. For more information, see the CCoE posts on the AWS Cloud Enterprise Strategy Blog.

cloud computing

The cloud technology that is typically used for remote data storage and IoT device management. Cloud computing is commonly connected to edge computing technology.

cloud operating model

In an IT organization, the operating model that is used to build, mature, and optimize one or more cloud environments. For more information, see Building your Cloud Operating Model.

cloud stages of adoption

The four phases that organizations typically go through when they migrate to the AWS Cloud:

- Project – Running a few cloud-related projects for proof of concept and learning purposes
- Foundation – Making foundational investments to scale your cloud adoption (e.g., creating a landing zone, defining a CCoE, establishing an operations model)
- Migration – Migrating individual applications
- Re-invention – Optimizing products and services, and innovating in the cloud

These stages were defined by Stephen Orban in the blog post The Journey Toward Cloud-First & the Stages of Adoption on the AWS Cloud Enterprise Strategy blog. For information about how they relate to the AWS migration strategy, see the migration readiness guide.

CMDB

See configuration management database.

code repository

A location where source code and other assets, such as documentation, samples, and scripts, are stored and updated through version control processes. Common cloud repositories include GitHub or Bitbucket Cloud. Each version of the code is called a *branch*. In a microservice structure, each repository is devoted to a single piece of functionality. A single CI/CD pipeline can use multiple repositories.

cold cache

A buffer cache that is empty, not well populated, or contains stale or irrelevant data. This affects performance because the database instance must read from the main memory or disk, which is slower than reading from the buffer cache.

cold data

Data that is rarely accessed and is typically historical. When querying this kind of data, slow queries are typically acceptable. Moving this data to lower-performing and less expensive storage tiers or classes can reduce costs.

computer vision (CV)

A field of AI that uses machine learning to analyze and extract information from visual formats such as digital images and videos. For example, Amazon SageMaker AI provides image processing algorithms for CV.

configuration drift

For a workload, a configuration change from the expected state. It might cause the workload to become noncompliant, and it's typically gradual and unintentional.

configuration management database (CMDB)

A repository that stores and manages information about a database and its IT environment, including both hardware and software components and their configurations. You typically use data from a CMDB in the portfolio discovery and analysis stage of migration.

conformance pack

A collection of AWS Config rules and remediation actions that you can assemble to customize your compliance and security checks. You can deploy a conformance pack as a single entity in an AWS account and Region, or across an organization, by using a YAML template. For more information, see Conformance packs in the AWS Config documentation.

continuous integration and continuous delivery (CI/CD)

The process of automating the source, build, test, staging, and production stages of the software release process. CI/CD is commonly described as a pipeline. CI/CD can help you automate processes, improve productivity, improve code quality, and deliver faster. For more information, see Benefits of continuous delivery. CD can also stand for *continuous deployment*. For more information, see Continuous Delivery vs. Continuous Deployment.

CV

See computer vision.

D

data at rest

Data that is stationary in your network, such as data that is in storage.

data classification

A process for identifying and categorizing the data in your network based on its criticality and sensitivity. It is a critical component of any cybersecurity risk management strategy because it helps you determine the appropriate protection and retention controls for the data. Data classification is a component of the security pillar in the AWS Well-Architected Framework. For more information, see Data classification.

data drift

A meaningful variation between the production data and the data that was used to train an ML model, or a meaningful change in the input data over time. Data drift can reduce the overall quality, accuracy, and fairness in ML model predictions.

data in transit

Data that is actively moving through your network, such as between network resources.

data mesh

An architectural framework that provides distributed, decentralized data ownership with centralized management and governance.

data minimization

The principle of collecting and processing only the data that is strictly necessary. Practicing data minimization in the AWS Cloud can reduce privacy risks, costs, and your analytics carbon footprint.

data perimeter

A set of preventive guardrails in your AWS environment that help make sure that only trusted identities are accessing trusted resources from expected networks. For more information, see Building a data perimeter on AWS.

data preprocessing

To transform raw data into a format that is easily parsed by your ML model. Preprocessing data can mean removing certain columns or rows and addressing missing, inconsistent, or duplicate values.

data provenance

The process of tracking the origin and history of data throughout its lifecycle, such as how the data was generated, transmitted, and stored.

data subject

An individual whose data is being collected and processed.

data warehouse

A data management system that supports business intelligence, such as analytics. Data warehouses commonly contain large amounts of historical data, and they are typically used for queries and analysis.

database definition language (DDL)

Statements or commands for creating or modifying the structure of tables and objects in a database.

database manipulation language (DML)

Statements or commands for modifying (inserting, updating, and deleting) information in a database.

DDL

See database definition language.

deep ensemble

To combine multiple deep learning models for prediction. You can use deep ensembles to obtain a more accurate prediction or for estimating uncertainty in predictions.

deep learning

An ML subfield that uses multiple layers of artificial neural networks to identify mapping between input data and target variables of interest.

defense-in-depth

An information security approach in which a series of security mechanisms and controls are thoughtfully layered throughout a computer network to protect the confidentiality, integrity, and availability of the network and the data within. When you adopt this strategy on AWS, you add multiple controls at different layers of the AWS Organizations structure to help secure resources. For example, a defense-in-depth approach might combine multi-factor authentication, network segmentation, and encryption.

delegated administrator

In AWS Organizations, a compatible service can register an AWS member account to administer the organization's accounts and manage permissions for that service. This account is called the *delegated administrator* for that service. For more information and a list of compatible services, see Services that work with AWS Organizations in the AWS Organizations documentation.

deployment

The process of making an application, new features, or code fixes available in the target environment. Deployment involves implementing changes in a code base and then building and running that code base in the application's environments.

development environment

See environment.

detective control

A security control that is designed to detect, log, and alert after an event has occurred. These controls are a second line of defense, alerting you to security events that bypassed the preventative controls in place. For more information, see Detective controls in *Implementing security controls on AWS*.

development value stream mapping (DVSM)

A process used to identify and prioritize constraints that adversely affect speed and quality in a software development lifecycle. DVSM extends the value stream mapping process originally designed for lean manufacturing practices. It focuses on the steps and teams required to create and move value through the software development process.

digital twin

A virtual representation of a real-world system, such as a building, factory, industrial equipment, or production line. Digital twins support predictive maintenance, remote monitoring, and production optimization.

dimension table

In a star schema, a smaller table that contains data attributes about quantitative data in a fact table. Dimension table attributes are typically text fields or discrete numbers that behave like text. These attributes are commonly used for query constraining, filtering, and result set labeling.

disaster

An event that prevents a workload or system from fulfilling its business objectives in its primary deployed location. These events can be natural disasters, technical failures, or the result of human actions, such as unintentional misconfiguration or a malware attack.

disaster recovery (DR)

The strategy and process you use to minimize downtime and data loss caused by a disaster. For more information, see Disaster Recovery of Workloads on AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML

See database manipulation language.

domain-driven design

An approach to developing a complex software system by connecting its components to evolving domains, or core business goals, that each component serves. This concept was introduced by Eric Evans in his book, *Domain-Driven Design: Tackling Complexity in the Heart of Software* (Boston: Addison-Wesley Professional, 2003). For information about how you can use domain-driven design with the strangler fig pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

DR

See disaster recovery.

drift detection

Tracking deviations from a baselined configuration. For example, you can use AWS CloudFormation to detect drift in system resources, or you can use AWS Control Tower to detect changes in your landing zone that might affect compliance with governance requirements.

DVSM

See development value stream mapping.

E

EDA

See exploratory data analysis.

EDI

See electronic data interchange.

edge computing

The technology that increases the computing power for smart devices at the edges of an IoT network. When compared with cloud computing, edge computing can reduce communication latency and improve response time.

electronic data interchange (EDI)

The automated exchange of business documents between organizations. For more information, see What is Electronic Data Interchange.

encryption

A computing process that transforms plaintext data, which is human-readable, into ciphertext.

encryption key

A cryptographic string of randomized bits that is generated by an encryption algorithm. Keys can vary in length, and each key is designed to be unpredictable and unique.

endianness

The order in which bytes are stored in computer memory. Big-endian systems store the most significant byte first. Little-endian systems store the least significant byte first.

endpoint

See service endpoint.

endpoint service

A service that you can host in a virtual private cloud (VPC) to share with other users. You can create an endpoint service with AWS PrivateLink and grant permissions to other AWS accounts or to AWS Identity and Access Management (IAM) principals. These accounts or principals can connect to your endpoint service privately by creating interface VPC endpoints. For more

information, see Create an endpoint service in the Amazon Virtual Private Cloud (Amazon VPC) documentation.

enterprise resource planning (ERP)

A system that automates and manages key business processes (such as accounting, MES, and project management) for an enterprise.

envelope encryption

The process of encrypting an encryption key with another encryption key. For more information, see Envelope encryption in the AWS Key Management Service (AWS KMS) documentation.

environment

An instance of a running application. The following are common types of environments in cloud computing:

- development environment – An instance of a running application that is available only to the core team responsible for maintaining the application. Development environments are used to test changes before promoting them to upper environments. This type of environment is sometimes referred to as a *test environment*.
- lower environments – All development environments for an application, such as those used for initial builds and tests.
- production environment – An instance of a running application that end users can access. In a CI/CD pipeline, the production environment is the last deployment environment.
- upper environments – All environments that can be accessed by users other than the core development team. This can include a production environment, preproduction environments, and environments for user acceptance testing.

epic

In agile methodologies, functional categories that help organize and prioritize your work. Epics provide a high-level description of requirements and implementation tasks. For example, AWS CAF security epics include identity and access management, detective controls, infrastructure security, data protection, and incident response. For more information about epics in the AWS migration strategy, see the program implementation guide.

ERP

See enterprise resource planning.

exploratory data analysis (EDA)

The process of analyzing a dataset to understand its main characteristics. You collect or aggregate data and then perform initial investigations to find patterns, detect anomalies, and check assumptions. EDA is performed by calculating summary statistics and creating data visualizations.

F

fact table

The central table in a star schema. It stores quantitative data about business operations. Typically, a fact table contains two types of columns: those that contain measures and those that contain a foreign key to a dimension table.

fail fast

A philosophy that uses frequent and incremental testing to reduce the development lifecycle. It is a critical part of an agile approach.

fault isolation boundary

In the AWS Cloud, a boundary such as an Availability Zone, AWS Region, control plane, or data plane that limits the effect of a failure and helps improve the resilience of workloads. For more information, see AWS Fault Isolation Boundaries.

feature branch

See branch.

features

The input data that you use to make a prediction. For example, in a manufacturing context, features could be images that are periodically captured from the manufacturing line.

feature importance

How significant a feature is for a model's predictions. This is usually expressed as a numerical score that can be calculated through various techniques, such as Shapley Additive Explanations (SHAP) and integrated gradients. For more information, see Machine learning model interpretability with AWS.

feature transformation

To optimize data for the ML process, including enriching data with additional sources, scaling values, or extracting multiple sets of information from a single data field. This enables the ML model to benefit from the data. For example, if you break down the “2021-05-27 00:15:37” date into “2021”, “May”, “Thu”, and “15”, you can help the learning algorithm learn nuanced patterns associated with different data components.

few-shot prompting

Providing an LLM with a small number of examples that demonstrate the task and desired output before asking it to perform a similar task. This technique is an application of in-context learning, where models learn from examples (*shots*) that are embedded in prompts. Few-shot prompting can be effective for tasks that require specific formatting, reasoning, or domain knowledge. See also zero-shot prompting.

FGAC

See fine-grained access control.

fine-grained access control (FGAC)

The use of multiple conditions to allow or deny an access request.

flash-cut migration

A database migration method that uses continuous data replication through change data capture to migrate data in the shortest time possible, instead of using a phased approach. The objective is to keep downtime to a minimum.

FM

See foundation model.

foundation model (FM)

A large deep-learning neural network that has been training on massive datasets of generalized and unlabeled data. FMs are capable of performing a wide variety of general tasks, such as understanding language, generating text and images, and conversing in natural language. For more information, see What are Foundation Models.

G

generative AI

A subset of AI models that have been trained on large amounts of data and that can use a simple text prompt to create new content and artifacts, such as images, videos, text, and audio. For more information, see What is Generative AI.

geo blocking

See geographic restrictions.

geographic restrictions (geo blocking)

In Amazon CloudFront, an option to prevent users in specific countries from accessing content distributions. You can use an allow list or block list to specify approved and banned countries. For more information, see Restricting the geographic distribution of your content in the CloudFront documentation.

Gitflow workflow

An approach in which lower and upper environments use different branches in a source code repository. The Gitflow workflow is considered legacy, and the trunk-based workflow is the modern, preferred approach.

golden image

A snapshot of a system or software that is used as a template to deploy new instances of that system or software. For example, in manufacturing, a golden image can be used to provision software on multiple devices and helps improve speed, scalability, and productivity in device manufacturing operations.

greenfield strategy

The absence of existing infrastructure in a new environment. When adopting a greenfield strategy for a system architecture, you can select all new technologies without the restriction of compatibility with existing infrastructure, also known as brownfield. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

guardrail

A high-level rule that helps govern resources, policies, and compliance across organizational units (OUs). *Preventive guardrails* enforce policies to ensure alignment to compliance standards. They are implemented by using service control policies and IAM permissions boundaries.

Detective guardrails detect policy violations and compliance issues, and generate alerts for remediation. They are implemented by using AWS Config, AWS Security Hub CSPM, Amazon GuardDuty, AWS Trusted Advisor, Amazon Inspector, and custom AWS Lambda checks.

H

HA

See high availability.

heterogeneous database migration

Migrating your source database to a target database that uses a different database engine (for example, Oracle to Amazon Aurora). Heterogeneous migration is typically part of a re-architecting effort, and converting the schema can be a complex task. AWS provides AWS SCT that helps with schema conversions.

high availability (HA)

The ability of a workload to operate continuously, without intervention, in the event of challenges or disasters. HA systems are designed to automatically fail over, consistently deliver high-quality performance, and handle different loads and failures with minimal performance impact.

historian modernization

An approach used to modernize and upgrade operational technology (OT) systems to better serve the needs of the manufacturing industry. A *historian* is a type of database that is used to collect and store data from various sources in a factory.

holdout data

A portion of historical, labeled data that is withheld from a dataset that is used to train a machine learning model. You can use holdout data to evaluate the model performance by comparing the model predictions against the holdout data.

homogeneous database migration

Migrating your source database to a target database that shares the same database engine (for example, Microsoft SQL Server to Amazon RDS for SQL Server). Homogeneous migration is typically part of a rehosting or replatforming effort. You can use native database utilities to migrate the schema.

hot data

Data that is frequently accessed, such as real-time data or recent translational data. This data typically requires a high-performance storage tier or class to provide fast query responses.

hotfix

An urgent fix for a critical issue in a production environment. Due to its urgency, a hotfix is usually made outside of the typical DevOps release workflow.

hypercare period

Immediately following cutover, the period of time when a migration team manages and monitors the migrated applications in the cloud in order to address any issues. Typically, this period is 1–4 days in length. At the end of the hypercare period, the migration team typically transfers responsibility for the applications to the cloud operations team.

I

IaC

See infrastructure as code.

identity-based policy

A policy attached to one or more IAM principals that defines their permissions within the AWS Cloud environment.

idle application

An application that has an average CPU and memory usage between 5 and 20 percent over a period of 90 days. In a migration project, it is common to retire these applications or retain them on premises.

IIoT

See industrial Internet of Things.

immutable infrastructure

A model that deploys new infrastructure for production workloads instead of updating, patching, or modifying the existing infrastructure. Immutable infrastructures are inherently more consistent, reliable, and predictable than mutable infrastructure. For more information, see the Deploy using immutable infrastructure best practice in the AWS Well-Architected Framework.

inbound (ingress) VPC

In an AWS multi-account architecture, a VPC that accepts, inspects, and routes network connections from outside an application. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

incremental migration

A cutover strategy in which you migrate your application in small parts instead of performing a single, full cutover. For example, you might move only a few microservices or users to the new system initially. After you verify that everything is working properly, you can incrementally move additional microservices or users until you can decommission your legacy system. This strategy reduces the risks associated with large migrations.

Industry 4.0

A term that was introduced by Klaus Schwab in 2016 to refer to the modernization of manufacturing processes through advances in connectivity, real-time data, automation, analytics, and AI/ML.

infrastructure

All of the resources and assets contained within an application's environment.

infrastructure as code (IaC)

The process of provisioning and managing an application's infrastructure through a set of configuration files. IaC is designed to help you centralize infrastructure management, standardize resources, and scale quickly so that new environments are repeatable, reliable, and consistent.

industrial Internet of Things (IIoT)

The use of internet-connected sensors and devices in the industrial sectors, such as manufacturing, energy, automotive, healthcare, life sciences, and agriculture. For more information, see Building an industrial Internet of Things (IIoT) digital transformation strategy.

inspection VPC

In an AWS multi-account architecture, a centralized VPC that manages inspections of network traffic between VPCs (in the same or different AWS Regions), the internet, and on-premises networks. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

Internet of Things (IoT)

The network of connected physical objects with embedded sensors or processors that communicate with other devices and systems through the internet or over a local communication network. For more information, see What is IoT?

interpretability

A characteristic of a machine learning model that describes the degree to which a human can understand how the model's predictions depend on its inputs. For more information, see Machine learning model interpretability with AWS.

IoT

See Internet of Things.

IT information library (ITIL)

A set of best practices for delivering IT services and aligning these services with business requirements. ITIL provides the foundation for ITSM.

IT service management (ITSM)

Activities associated with designing, implementing, managing, and supporting IT services for an organization. For information about integrating cloud operations with ITSM tools, see the operations integration guide.

ITIL

See IT information library.

ITSM

See IT service management.

L

label-based access control (LBAC)

An implementation of mandatory access control (MAC) where the users and the data itself are each explicitly assigned a security label value. The intersection between the user security label and data security label determines which rows and columns can be seen by the user.

landing zone

A landing zone is a well-architected, multi-account AWS environment that is scalable and secure. This is a starting point from which your organizations can quickly launch and deploy workloads and applications with confidence in their security and infrastructure environment. For more information about landing zones, see Setting up a secure and scalable multi-account AWS environment.

large language model (LLM)

A deep learning AI model that is pretrained on a vast amount of data. An LLM can perform multiple tasks, such as answering questions, summarizing documents, translating text into other languages, and completing sentences. For more information, see What are LLMs.

large migration

A migration of 300 or more servers.

LBAC

See label-based access control.

least privilege

The security best practice of granting the minimum permissions required to perform a task. For more information, see Apply least-privilege permissions in the IAM documentation.

lift and shift

See 7 Rs.

little-endian system

A system that stores the least significant byte first. See also endianness.

LLM

See large language model.

lower environments

See environment.

M

machine learning (ML)

A type of artificial intelligence that uses algorithms and techniques for pattern recognition and learning. ML analyzes and learns from recorded data, such as Internet of Things (IoT) data, to generate a statistical model based on patterns. For more information, see Machine Learning.

main branch

See branch.

malware

Software that is designed to compromise computer security or privacy. Malware might disrupt computer systems, leak sensitive information, or gain unauthorized access. Examples of malware include viruses, worms, ransomware, Trojan horses, spyware, and keyloggers.

managed services

AWS services for which AWS operates the infrastructure layer, the operating system, and platforms, and you access the endpoints to store and retrieve data. Amazon Simple Storage Service (Amazon S3) and Amazon DynamoDB are examples of managed services. These are also known as *abstracted services*.

manufacturing execution system (MES)

A software system for tracking, monitoring, documenting, and controlling production processes that convert raw materials to finished products on the shop floor.

MAP

See Migration Acceleration Program.

mechanism

A complete process in which you create a tool, drive adoption of the tool, and then inspect the results in order to make adjustments. A mechanism is a cycle that reinforces and improves itself as it operates. For more information, see Building mechanisms in the AWS Well-Architected Framework.

member account

All AWS accounts other than the management account that are part of an organization in AWS Organizations. An account can be a member of only one organization at a time.

MES

See manufacturing execution system.

Message Queuing Telemetry Transport (MQTT)

A lightweight, machine-to-machine (M2M) communication protocol, based on the publish/subscribe pattern, for resource-constrained IoT devices.

microservice

A small, independent service that communicates over well-defined APIs and is typically owned by small, self-contained teams. For example, an insurance system might include microservices that map to business capabilities, such as sales or marketing, or subdomains, such as purchasing, claims, or analytics. The benefits of microservices include agility, flexible scaling, easy deployment, reusable code, and resilience. For more information, see Integrating microservices by using AWS serverless services.

microservices architecture

An approach to building an application with independent components that run each application process as a microservice. These microservices communicate through a well-defined interface by using lightweight APIs. Each microservice in this architecture can be updated, deployed, and scaled to meet demand for specific functions of an application. For more information, see Implementing microservices on AWS.

Migration Acceleration Program (MAP)

An AWS program that provides consulting support, training, and services to help organizations build a strong operational foundation for moving to the cloud, and to help offset the initial cost of migrations. MAP includes a migration methodology for executing legacy migrations in a methodical way and a set of tools to automate and accelerate common migration scenarios.

migration at scale

The process of moving the majority of the application portfolio to the cloud in waves, with more applications moved at a faster rate in each wave. This phase uses the best practices and lessons learned from the earlier phases to implement a *migration factory* of teams, tools, and processes to streamline the migration of workloads through automation and agile delivery. This is the third phase of the AWS migration strategy.

migration factory

Cross-functional teams that streamline the migration of workloads through automated, agile approaches. Migration factory teams typically include operations, business analysts and owners,

migration engineers, developers, and DevOps professionals working in sprints. Between 20 and 50 percent of an enterprise application portfolio consists of repeated patterns that can be optimized by a factory approach. For more information, see the discussion of migration factories and the Cloud Migration Factory guide in this content set.

migration metadata

The information about the application and server that is needed to complete the migration. Each migration pattern requires a different set of migration metadata. Examples of migration metadata include the target subnet, security group, and AWS account.

migration pattern

A repeatable migration task that details the migration strategy, the migration destination, and the migration application or service used. Example: Rehost migration to Amazon EC2 with AWS Application Migration Service.

Migration Portfolio Assessment (MPA)

An online tool that provides information for validating the business case for migrating to the AWS Cloud. MPA provides detailed portfolio assessment (server right-sizing, pricing, TCO comparisons, migration cost analysis) as well as migration planning (application data analysis and data collection, application grouping, migration prioritization, and wave planning). The MPA tool (requires login) is available free of charge to all AWS consultants and APN Partner consultants.

Migration Readiness Assessment (MRA)

The process of gaining insights about an organization's cloud readiness status, identifying strengths and weaknesses, and building an action plan to close identified gaps, using the AWS CAF. For more information, see the migration readiness guide. MRA is the first phase of the AWS migration strategy.

migration strategy

The approach used to migrate a workload to the AWS Cloud. For more information, see the 7 Rs entry in this glossary and see Mobilize your organization to accelerate large-scale migrations.

ML

See machine learning.

modernization

Transforming an outdated (legacy or monolithic) application and its infrastructure into an agile, elastic, and highly available system in the cloud to reduce costs, gain efficiencies, and take advantage of innovations. For more information, see Strategy for modernizing applications in the AWS Cloud.

modernization readiness assessment

An evaluation that helps determine the modernization readiness of an organization's applications; identifies benefits, risks, and dependencies; and determines how well the organization can support the future state of those applications. The outcome of the assessment is a blueprint of the target architecture, a roadmap that details development phases and milestones for the modernization process, and an action plan for addressing identified gaps. For more information, see Evaluating modernization readiness for applications in the AWS Cloud.

monolithic applications (monoliths)

Applications that run as a single service with tightly coupled processes. Monolithic applications have several drawbacks. If one application feature experiences a spike in demand, the entire architecture must be scaled. Adding or improving a monolithic application's features also becomes more complex when the code base grows. To address these issues, you can use a microservices architecture. For more information, see Decomposing monoliths into microservices.

MPA

See Migration Portfolio Assessment.

MQTT

See Message Queuing Telemetry Transport.

multiclass classification

A process that helps generate predictions for multiple classes (predicting one of more than two outcomes). For example, an ML model might ask "Is this product a book, car, or phone?" or "Which product category is most interesting to this customer?"

mutable infrastructure

A model that updates and modifies the existing infrastructure for production workloads. For improved consistency, reliability, and predictability, the AWS Well-Architected Framework recommends the use of immutable infrastructure as a best practice.

O

OAC

See origin access control.

OAI

See origin access identity.

OCM

See organizational change management.

offline migration

A migration method in which the source workload is taken down during the migration process. This method involves extended downtime and is typically used for small, non-critical workloads.

OI

See operations integration.

OLA

See operational-level agreement.

online migration

A migration method in which the source workload is copied to the target system without being taken offline. Applications that are connected to the workload can continue to function during the migration. This method involves zero to minimal downtime and is typically used for critical production workloads.

OPC-UA

See Open Process Communications - Unified Architecture.

Open Process Communications - Unified Architecture (OPC-UA)

A machine-to-machine (M2M) communication protocol for industrial automation. OPC-UA provides an interoperability standard with data encryption, authentication, and authorization schemes.

operational-level agreement (OLA)

An agreement that clarifies what functional IT groups promise to deliver to each other, to support a service-level agreement (SLA).

operational readiness review (ORR)

A checklist of questions and associated best practices that help you understand, evaluate, prevent, or reduce the scope of incidents and possible failures. For more information, see Operational Readiness Reviews (ORR) in the AWS Well-Architected Framework.

operational technology (OT)

Hardware and software systems that work with the physical environment to control industrial operations, equipment, and infrastructure. In manufacturing, the integration of OT and information technology (IT) systems is a key focus for Industry 4.0 transformations.

operations integration (OI)

The process of modernizing operations in the cloud, which involves readiness planning, automation, and integration. For more information, see the operations integration guide.

organization trail

A trail that's created by AWS CloudTrail that logs all events for all AWS accounts in an organization in AWS Organizations. This trail is created in each AWS account that's part of the organization and tracks the activity in each account. For more information, see Creating a trail for an organization in the CloudTrail documentation.

organizational change management (OCM)

A framework for managing major, disruptive business transformations from a people, culture, and leadership perspective. OCM helps organizations prepare for, and transition to, new systems and strategies by accelerating change adoption, addressing transitional issues, and driving cultural and organizational changes. In the AWS migration strategy, this framework is called *people acceleration*, because of the speed of change required in cloud adoption projects. For more information, see the OCM guide.

origin access control (OAC)

In CloudFront, an enhanced option for restricting access to secure your Amazon Simple Storage Service (Amazon S3) content. OAC supports all S3 buckets in all AWS Regions, server-side encryption with AWS KMS (SSE-KMS), and dynamic PUT and DELETE requests to the S3 bucket.

origin access identity (OAI)

In CloudFront, an option for restricting access to secure your Amazon S3 content. When you use OAI, CloudFront creates a principal that Amazon S3 can authenticate with. Authenticated principals can access content in an S3 bucket only through a specific CloudFront distribution. See also OAC, which provides more granular and enhanced access control.

ORR

See operational readiness review.

OT

See operational technology.

outbound (egress) VPC

In an AWS multi-account architecture, a VPC that handles network connections that are initiated from within an application. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

P

permissions boundary

An IAM management policy that is attached to IAM principals to set the maximum permissions that the user or role can have. For more information, see Permissions boundaries in the IAM documentation.

personally identifiable information (PII)

Information that, when viewed directly or paired with other related data, can be used to reasonably infer the identity of an individual. Examples of PII include names, addresses, and contact information.

PII

See personally identifiable information.

playbook

A set of predefined steps that capture the work associated with migrations, such as delivering core operations functions in the cloud. A playbook can take the form of scripts, automated runbooks, or a summary of processes or steps required to operate your modernized environment.

PLC

See programmable logic controller.

PLM

See product lifecycle management.

policy

An object that can define permissions (see identity-based policy), specify access conditions (see resource-based policy), or define the maximum permissions for all accounts in an organization in AWS Organizations (see service control policy).

polyglot persistence

Independently choosing a microservice's data storage technology based on data access patterns and other requirements. If your microservices have the same data storage technology, they can encounter implementation challenges or experience poor performance. Microservices are more easily implemented and achieve better performance and scalability if they use the data store best adapted to their requirements.

portfolio assessment

A process of discovering, analyzing, and prioritizing the application portfolio in order to plan the migration. For more information, see Evaluating migration readiness.

predicate

A query condition that returns `true` or `false`, commonly located in a `WHERE` clause.

predicate pushdown

A database query optimization technique that filters the data in the query before transfer. This reduces the amount of data that must be retrieved and processed from the relational database, and it improves query performance.

preventative control

A security control that is designed to prevent an event from occurring. These controls are a first line of defense to help prevent unauthorized access or unwanted changes to your network. For more information, see Preventative controls in *Implementing security controls on AWS*.

principal

An entity in AWS that can perform actions and access resources. This entity is typically a root user for an AWS account, an IAM role, or a user. For more information, see *Principal* in Roles terms and concepts in the IAM documentation.

privacy by design

A system engineering approach that takes privacy into account through the whole development process.

private hosted zones

A container that holds information about how you want Amazon Route 53 to respond to DNS queries for a domain and its subdomains within one or more VPCs. For more information, see Working with private hosted zones in the Route 53 documentation.

proactive control

A security control designed to prevent the deployment of noncompliant resources. These controls scan resources before they are provisioned. If the resource is not compliant with the control, then it isn't provisioned. For more information, see the Controls reference guide in the AWS Control Tower documentation and see Proactive controls in *Implementing security controls on AWS*.

product lifecycle management (PLM)

The management of data and processes for a product throughout its entire lifecycle, from design, development, and launch, through growth and maturity, to decline and removal.

production environment

See environment.

programmable logic controller (PLC)

In manufacturing, a highly reliable, adaptable computer that monitors machines and automates manufacturing processes.

prompt chaining

Using the output of one LLM prompt as the input for the next prompt to generate better responses. This technique is used to break down a complex task into subtasks, or to iteratively refine or expand a preliminary response. It helps improve the accuracy and relevance of a model's responses and allows for more granular, personalized results.

pseudonymization

The process of replacing personal identifiers in a dataset with placeholder values. Pseudonymization can help protect personal privacy. Pseudonymized data is still considered to be personal data.

publish/subscribe (pub/sub)

A pattern that enables asynchronous communications among microservices to improve scalability and responsiveness. For example, in a microservices-based MES, a microservice can publish event messages to a channel that other microservices can subscribe to. The system can add new microservices without changing the publishing service.

Q

query plan

A series of steps, like instructions, that are used to access the data in a SQL relational database system.

query plan regression

When a database service optimizer chooses a less optimal plan than it did before a given change to the database environment. This can be caused by changes to statistics, constraints, environment settings, query parameter bindings, and updates to the database engine.

R

RACI matrix

See responsible, accountable, consulted, informed (RACI).

RAG

See Retrieval Augmented Generation.

ransomware

A malicious software that is designed to block access to a computer system or data until a payment is made.

RASCI matrix

See responsible, accountable, consulted, informed (RACI).

RCAC

See row and column access control.

read replica

A copy of a database that's used for read-only purposes. You can route queries to the read replica to reduce the load on your primary database.

re-architect

See 7 Rs.

recovery point objective (RPO)

The maximum acceptable amount of time since the last data recovery point. This determines what is considered an acceptable loss of data between the last recovery point and the interruption of service.

recovery time objective (RTO)

The maximum acceptable delay between the interruption of service and restoration of service.

refactor

See 7 Rs.

Region

A collection of AWS resources in a geographic area. Each AWS Region is isolated and independent of the others to provide fault tolerance, stability, and resilience. For more information, see Specify which AWS Regions your account can use.

regression

An ML technique that predicts a numeric value. For example, to solve the problem of "What price will this house sell for?" an ML model could use a linear regression model to predict a house's sale price based on known facts about the house (for example, the square footage).

rehost

See 7 Rs.

release

In a deployment process, the act of promoting changes to a production environment.

relocate

See 7 Rs.

replatform

See 7 Rs.

repurchase

See 7 Rs.

resiliency

An application's ability to resist or recover from disruptions. High availability and disaster recovery are common considerations when planning for resiliency in the AWS Cloud. For more information, see AWS Cloud Resilience.

resource-based policy

A policy attached to a resource, such as an Amazon S3 bucket, an endpoint, or an encryption key. This type of policy specifies which principals are allowed access, supported actions, and any other conditions that must be met.

responsible, accountable, consulted, informed (RACI) matrix

A matrix that defines the roles and responsibilities for all parties involved in migration activities and cloud operations. The matrix name is derived from the responsibility types defined in the matrix: responsible (R), accountable (A), consulted (C), and informed (I). The support (S) type is optional. If you include support, the matrix is called a *RASCI matrix*, and if you exclude it, it's called a *RACI matrix*.

responsive control

A security control that is designed to drive remediation of adverse events or deviations from your security baseline. For more information, see Responsive controls in *Implementing security controls on AWS*.

retain

See 7 Rs.

retire

See 7 Rs.

Retrieval Augmented Generation (RAG)

A generative AI technology in which an LLM references an authoritative data source that is outside of its training data sources before generating a response. For example, a RAG model might perform a semantic search of an organization's knowledge base or custom data. For more information, see What is RAG.

rotation

The process of periodically updating a secret to make it more difficult for an attacker to access the credentials.

row and column access control (RCAC)

The use of basic, flexible SQL expressions that have defined access rules. RCAC consists of row permissions and column masks.

RPO

See recovery point objective.

RTO

See recovery time objective.

runbook

A set of manual or automated procedures required to perform a specific task. These are typically built to streamline repetitive operations or procedures with high error rates.

S

SAML 2.0

An open standard that many identity providers (IdPs) use. This feature enables federated single sign-on (SSO), so users can log into the AWS Management Console or call the AWS API operations without you having to create user in IAM for everyone in your organization. For more information about SAML 2.0-based federation, see About SAML 2.0-based federation in the IAM documentation.

SCADA

See supervisory control and data acquisition.

SCP

See service control policy.

secret

In AWS Secrets Manager, confidential or restricted information, such as a password or user credentials, that you store in encrypted form. It consists of the secret value and its metadata.

The secret value can be binary, a single string, or multiple strings. For more information, see What's in a Secrets Manager secret? in the Secrets Manager documentation.

security by design

A system engineering approach that takes security into account through the whole development process.

security control

A technical or administrative guardrail that prevents, detects, or reduces the ability of a threat actor to exploit a security vulnerability. There are four primary types of security controls: preventative, detective, responsive, and proactive.

security hardening

The process of reducing the attack surface to make it more resistant to attacks. This can include actions such as removing resources that are no longer needed, implementing the security best practice of granting least privilege, or deactivating unnecessary features in configuration files.

security information and event management (SIEM) system

Tools and services that combine security information management (SIM) and security event management (SEM) systems. A SIEM system collects, monitors, and analyzes data from servers, networks, devices, and other sources to detect threats and security breaches, and to generate alerts.

security response automation

A predefined and programmed action that is designed to automatically respond to or remediate a security event. These automations serve as detective or responsive security controls that help you implement AWS security best practices. Examples of automated response actions include modifying a VPC security group, patching an Amazon EC2 instance, or rotating credentials.

server-side encryption

Encryption of data at its destination, by the AWS service that receives it.

service control policy (SCP)

A policy that provides centralized control over permissions for all accounts in an organization in AWS Organizations. SCPs define guardrails or set limits on actions that an administrator can delegate to users or roles. You can use SCPs as allow lists or deny lists, to specify which services or actions are permitted or prohibited. For more information, see Service control policies in the AWS Organizations documentation.

service endpoint

The URL of the entry point for an AWS service. You can use the endpoint to connect programmatically to the target service. For more information, see AWS service endpoints in *AWS General Reference*.

service-level agreement (SLA)

An agreement that clarifies what an IT team promises to deliver to their customers, such as service uptime and performance.

service-level indicator (SLI)

A measurement of a performance aspect of a service, such as its error rate, availability, or throughput.

service-level objective (SLO)

A target metric that represents the health of a service, as measured by a service-level indicator.

shared responsibility model

A model describing the responsibility you share with AWS for cloud security and compliance. AWS is responsible for security *of* the cloud, whereas you are responsible for security *in* the cloud. For more information, see Shared responsibility model.

SIEM

See security information and event management system.

single point of failure (SPOF)

A failure in a single, critical component of an application that can disrupt the system.

SLA

See service-level agreement.

SLI

See service-level indicator.

SLO

See service-level objective.

split-and-seed model

A pattern for scaling and accelerating modernization projects. As new features and product releases are defined, the core team splits up to create new product teams. This helps scale your

organization's capabilities and services, improves developer productivity, and supports rapid innovation. For more information, see Phased approach to modernizing applications in the AWS Cloud.

SPOF

See single point of failure.

star schema

A database organizational structure that uses one large fact table to store transactional or measured data and uses one or more smaller dimensional tables to store data attributes. This structure is designed for use in a data warehouse or for business intelligence purposes.

strangler fig pattern

An approach to modernizing monolithic systems by incrementally rewriting and replacing system functionality until the legacy system can be decommissioned. This pattern uses the analogy of a fig vine that grows into an established tree and eventually overcomes and replaces its host. The pattern was introduced by Martin Fowler as a way to manage risk when rewriting monolithic systems. For an example of how to apply this pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

subnet

A range of IP addresses in your VPC. A subnet must reside in a single Availability Zone.

supervisory control and data acquisition (SCADA)

In manufacturing, a system that uses hardware and software to monitor physical assets and production operations.

symmetric encryption

An encryption algorithm that uses the same key to encrypt and decrypt the data.

synthetic testing

Testing a system in a way that simulates user interactions to detect potential issues or to monitor performance. You can use Amazon CloudWatch Synthetics to create these tests.

system prompt

A technique for providing context, instructions, or guidelines to an LLM to direct its behavior. System prompts help set context and establish rules for interactions with users.

T

tags

Key-value pairs that act as metadata for organizing your AWS resources. Tags can help you manage, identify, organize, search for, and filter resources. For more information, see Tagging your AWS resources.

target variable

The value that you are trying to predict in supervised ML. This is also referred to as an *outcome variable*. For example, in a manufacturing setting the target variable could be a product defect.

task list

A tool that is used to track progress through a runbook. A task list contains an overview of the runbook and a list of general tasks to be completed. For each general task, it includes the estimated amount of time required, the owner, and the progress.

test environment

See environment.

training

To provide data for your ML model to learn from. The training data must contain the correct answer. The learning algorithm finds patterns in the training data that map the input data attributes to the target (the answer that you want to predict). It outputs an ML model that captures these patterns. You can then use the ML model to make predictions on new data for which you don't know the target.

transit gateway

A network transit hub that you can use to interconnect your VPCs and on-premises networks. For more information, see What is a transit gateway in the AWS Transit Gateway documentation.

trunk-based workflow

An approach in which developers build and test features locally in a feature branch and then merge those changes into the main branch. The main branch is then built to the development, preproduction, and production environments, sequentially.

trusted access

Granting permissions to a service that you specify to perform tasks in your organization in AWS Organizations and in its accounts on your behalf. The trusted service creates a service-linked role in each account, when that role is needed, to perform management tasks for you. For more information, see Using AWS Organizations with other AWS services in the AWS Organizations documentation.

tuning

To change aspects of your training process to improve the ML model's accuracy. For example, you can train the ML model by generating a labeling set, adding labels, and then repeating these steps several times under different settings to optimize the model.

two-pizza team

A small DevOps team that you can feed with two pizzas. A two-pizza team size ensures the best possible opportunity for collaboration in software development.

U

uncertainty

A concept that refers to imprecise, incomplete, or unknown information that can undermine the reliability of predictive ML models. There are two types of uncertainty: *Epistemic uncertainty* is caused by limited, incomplete data, whereas *aleatoric uncertainty* is caused by the noise and randomness inherent in the data. For more information, see the Quantifying uncertainty in deep learning systems guide.

undifferentiated tasks

Also known as *heavy lifting*, work that is necessary to create and operate an application but that doesn't provide direct value to the end user or provide competitive advantage. Examples of undifferentiated tasks include procurement, maintenance, and capacity planning.

upper environments

See environment.

V

vacuuming

A database maintenance operation that involves cleaning up after incremental updates to reclaim storage and improve performance.

version control

Processes and tools that track changes, such as changes to source code in a repository.

VPC peering

A connection between two VPCs that allows you to route traffic by using private IP addresses. For more information, see What is VPC peering in the Amazon VPC documentation.

vulnerability

A software or hardware flaw that compromises the security of the system.

W

warm cache

A buffer cache that contains current, relevant data that is frequently accessed. The database instance can read from the buffer cache, which is faster than reading from the main memory or disk.

warm data

Data that is infrequently accessed. When querying this kind of data, moderately slow queries are typically acceptable.

window function

A SQL function that performs a calculation on a group of rows that relate in some way to the current record. Window functions are useful for processing tasks, such as calculating a moving average or accessing the value of rows based on the relative position of the current row.

workload

A collection of resources and code that delivers business value, such as a customer-facing application or backend process.

workstream

Functional groups in a migration project that are responsible for a specific set of tasks. Each workstream is independent but supports the other workstreams in the project. For example, the portfolio workstream is responsible for prioritizing applications, wave planning, and collecting migration metadata. The portfolio workstream delivers these assets to the migration workstream, which then migrates the servers and applications.

WORM

See write once, read many.

WQF

See AWS Workload Qualification Framework.

write once, read many (WORM)

A storage model that writes data a single time and prevents the data from being deleted or modified. Authorized users can read the data as many times as needed, but they cannot change it. This data storage infrastructure is considered immutable.

Z

zero-day exploit

An attack, typically malware, that takes advantage of a zero-day vulnerability.

zero-day vulnerability

An unmitigated flaw or vulnerability in a production system. Threat actors can use this type of vulnerability to attack the system. Developers frequently become aware of the vulnerability as a result of the attack.

zero-shot prompting

Providing an LLM with instructions for performing a task but no examples (*shots*) that can help guide it. The LLM must use its pre-trained knowledge to handle the task. The effectiveness of zero-shot prompting depends on the complexity of the task and the quality of the prompt. See also few-shot prompting.

zombie application

An application that has an average CPU and memory usage below 5 percent. In a migration project, it is common to retire these applications.