AWS Well-Architected Framework

Amazon OpenSearch Service Lens

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon OpenSearch Service Lens: AWS Well-Architected Framework

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

Abstract and introduction ..... **i**
Introduction 1
Custom lens availability 1
Definitions **2**
Design principles **4**
Scenarios **6**
Log analytics workloads 6
Use case: Web application log analysis 6
Search workloads 7
Use case: E-commerce product search 7
Operational excellence **9**
Design principles 9
Operate 10
AOSOPS01-BP01 Employ Index State Management (ISM) to manage logs or time series data 11
AOSOPS01-BP02 Configure index templates to automate index configuration upon creation 12
AOSOPS01-BP03 Remove unused indexes 14
AOSOPS02-BP01 Establish a manual snapshot repository 15
AOSOPS02-BP02 Automate the process of taking snapshots 16
AOSOPS03-BP01 Establish alarms for OpenSearch Service domain 18
AOSOPS03-BP02 Configure notification services to receive monitoring alerts 19
AOSOPS03-BP03 Enable search and indexing slow log functionality 20
AOSOPS03-BP04 Enable audit logs for OpenSearch Service domains using fine-grained access control 21
Evolve 22
AOSOPS04-BP01 Train staff on common OpenSearch issues and how to remediate them 22
Key AWS services 23
Resources 24
Security **25**
Design principles 25
Security foundation 26

AOSSEC01-BP01 Launch your OpenSearch Service domains within a Virtual Private Cloud 26
AOSSEC01-BP02 Activate node-to-node encryption 28
AOSSEC01-BP03 Enable encryption at rest 29
AOSSEC01-BP04 Encrypt slow and error logs in Amazon CloudWatch to protect sensitive information 31
Detection 31
AOSSEC02-BP01 Set up audit logging for OpenSearch Service domains that use fine-grained access control 32
AOSSEC02-BP02 Track OpenSearch Service API calls 33
AOSSEC02-BP03 Monitor real-time events in your OpenSearch Service domains 34
AOSSEC02-BP04 Assess your OpenSearch Service domain's configuration with AWS Config 35
Data protection 36
AOSSEC03-BP01 Implement fine-grained access control to manage access to your data on Amazon OpenSearch Service 36
AOSSEC03-BP02 Secure your indices, documents, and fields using fine-grained access control 38
Key AWS services 39
Resources 39
Reliability 41
Design principles 41
Foundations 41
AOSREL01-BP01 Implement a system update notification strategy 42
AOSREL01-BP02 Regularly update your OpenSearch Service domain to the latest version 43
Workload architecture 44
AOSREL02-BP01 Create Multi-AZ with Standby OpenSearch Service domains 45
AOSREL03-BP01 Regularly review your OpenSearch Service quotas 46
AOSREL04-BP01 Implement a disaster recovery (DR) strategy for your OpenSearch Service for business continuity 47
AOSREL04-BP02 Implement an Index State Management (ISM) policy to generate snapshots for your crucial indices 48
AOSREL04-BP03 Enable index replication for your critical indices 50
AOSREL04-BP04 Employ cross-cluster replication to achieve higher availability 51
AOSREL05-BP01 Implement appropriate compute sizing for production workloads 52

Key AWS services ... 53
Resources ... 53
Performance efficiency ... 55
Design principles ... 55
Architecture selection ... 55
AOSPERF01-BP01 Maintain shard sizes at recommended ranges ... 56
AOSPERF01-BP02 Check shard-to-CPU ratio ... 57
AOSPERF01-BP03 Check the number of shards per GiB of heap memory ... 58
AOSPERF02-BP01 Implement processor utilization monitoring ... 59
AOSPERF02-BP02 Implement Java memory utilization monitoring ... 60
Data management ... 62
AOSPERF03-BP01 Establish storage utilization thresholds ... 63
AOSPERF03-BP02 Evenly distribute data across data nodes in your OpenSearch Service domain ... 64
AOSPERF03-BP03 Use storage types that provide higher IOPs and throughput baseline ... 66
AOSPERF04-BP01 Enable slow log functionality for search and indexing ... 67
AOSPERF04-BP02 Use static mapping for your index ... 68
AOSPERF04-BP03 Use the flat object type for nested objects ... 70
AOSPERF05-BP01 Enable a dedicated leader node for OpenSearch Service domain ... 71
AOSPERF05-BP02 Enable a dedicated coordinator node for OpenSearch Service domain ... 73
Process and culture ... 75
AOSPERF06-BP01 Identify index refresh controls for optimal ingestion performance ... 75
AOSPERF06-BP02 Evaluate bulk request size ... 76
AOSPERF06-BP03 Implement HTTP compression ... 77
AOSPERF06-BP04 Evaluate `filter_path` criteria ... 78
Key AWS services ... 79
Resources ... 79
Cost optimization ... 81
Design principles ... 81
Practice Cloud Financial Management (CFM) ... 82
AOSCOST01-BP01 Use the latest generation of instances for your OpenSearch Service domains ... 83
AOSCOST01-BP02 Employ the appropriate instance type and count ... 84
AOSCOST01-BP03 Evaluate manager nodes ... 85
Expenditure and usage awareness ... 86

AOSCOST02-BP01 Use the latest Amazon EBS gp3 volumes with your OpenSearch Service nodes ... 87
AOSCOST02-BP02 Use instances optimized for heavy indexing use cases ... 88
AOSCOST02-BP03 Use the warm storage tier to optimize storage for a significant amount of read-only data ... 89
AOSCOST02-BP04 Use the cold tier storage option to store and retrieve infrequently accessed or historical data ... 90
Cost-effective resources ... 92
AOSCOST03-BP01 Evaluate forecasting for your workloads ... 92
Optimize over time ... 93
AOSCOST04-BP01 Apply cost allocation tags to your OpenSearch resources for detailed cost tracking and analysis ... 94
AOSCOST05-BP01 Assess the pricing for instances and storage in Amazon OpenSearch Service ... 95
AOSCOST05-BP02 Examine the costs associated with Amazon S3 storage for manually creating snapshots of your OpenSearch Service domain ... 96
Key AWS services ... 97
Resources ... 97
Sustainability ... 99
Design principles ... 99
Region selection ... 100
AOSSUS01-BP01 Select the Region for your OpenSearch Service deployment based on a combination of business requirements and sustainability goals ... 101
Software and architecture ... 102
AOSSUS02-BP01 Evaluate instances in alignment to sustainability goals ... 103
AOSSUS02-BP02 Use the minimum number of instances necessary to meet your workload requirements ... 104
Data management ... 105
AOSSUS03-BP01 Use Index State Management to manage the lifecycle of your dataset ... 106
AOSSUS03-BP02 Reduce unnecessary or redundant data from your domain ... 107
AOSSUS03-BP03 Take manual snapshots of your indices only when it is difficult to recreate the dataset ... 108
Process and culture ... 108
AOSSUS04-BP01 Consolidate OpenSearch Service domain environments ... 109
Key AWS services ... 110
Resources ... 110

Conclusion **111**
Contributors **112**
Document revisions **113**
AWS Glossary **114**

Amazon OpenSearch Service Lens

Publication date: **June 23, 2025** (*Document revisions*)

The AWS Well-Architected Amazon OpenSearch Service Lens serves as a valuable resource for engineering and implementing secure, efficient, and high-performing OpenSearch Service workloads. This lens caters to various technology professionals such as Chief Technology Officers (CTOs), architects, developers, and operational teams. By using this lens, users can acquire best practices and effective strategies to optimize their OpenSearch Service design, thereby adhering to the AWS Well-Architected Framework.

Introduction

The Amazon OpenSearch Service Lens is a tool designed to help cloud architects and technology professionals create scalable, secure, and efficient Amazon OpenSearch Service workloads. The lens is based on the six pillars of the AWS Well-Architected Framework: Operational Excellence, Security, Reliability, Performance Efficiency, Cost Optimization, and Sustainability. These pillars provide a standardized approach to assessing architectures and deploying scalable designs that meet diverse application and workload requirements.

The lens offers best practices, design principles, and assessment questions specifically tailored for OpenSearch Service workloads. This guidance is informed by our extensive experience collaborating with customers across various industries, segments, sizes, and geographical locations.

By using the Amazon OpenSearch Service Lens, you can gain a comprehensive understanding of AWS best practices and strategies to design and operate optimal architectures for Amazon OpenSearch Service. The lens provides actionable advice on recommended design principles aligned with the AWS Well-Architected Framework, helping you create highly available, secure, and efficient workloads that meet business requirements.

Custom lens availability

Custom lenses extend the best practice guidance provided by AWS Well-Architected Tool. AWS WA Tool allows you to create your own custom lenses, or to use lenses created by others that have been shared with you.

To determine if a custom lens is available for the lens described in this whitepaper, reach out to your Technical Account Manager (TAM), Solutions Architect (SA), or Support.

Definitions

- **Domain:** The primary container for all OpenSearch resources. It represents the entire environment where your data is stored and indexed.
- **Node:** A single instance of OpenSearch running within your domain. Nodes work together to form a cluster and distribute the workloads and data among them.
- **Leader nodes:** Nodes that perform cluster management tasks but do not hold data or respond to data requests.
- **Dedicated leader nodes:** Nodes that serve exclusively as specialized nodes that are configured to perform the role of a leader.
- **Data nodes:** Nodes that are responsible for storing and managing data. Data nodes handle indexing and searching operations and store the actual index and document data.
- **UltraWarm and cold nodes:** UltraWarm nodes provide a cost-effective way to store large amounts of read-only data on Amazon OpenSearch Service. Rather than attached storage, UltraWarm nodes use Amazon S3 and a sophisticated caching solution to improve performance.
- **Multi-AZ with Standby:** A deployment option for Amazon OpenSearch Service domains providing 99.99% availability and consistent performance by spanning three Availability Zones, each with a complete data copy.
- **Index:** A data structure used to store, organize, and search documents, similar to a database table.
- **Shard:** A smaller, more manageable segment of an index. Each shard acts as a self-contained index, and in Amazon OpenSearch Service, shards are distributed across data nodes for load balancing, redundancy, and fault tolerance.
- **Replica:** A duplication of a shard, offering failover and load balancing capabilities. Amazon OpenSearch Service automatically distributes replica shards across various nodes within the domain.
- **Document:** A single unit of searchable data, like a row in a database table. Each document possesses a unique identification (ID) and encompasses a set of fields.
- **Field:** A named, typed value within a document, resembling a column in a database table. Fields may have various data types, including string, number, date, or Boolean.
- **Mapping:** Defines the structure of documents in an index, including field names, data types, and other metadata. It acts as a schema for the documents and helps Amazon OpenSearch Service understand the data and optimize search performance.

- **Query:** A request to search, filter, or aggregate data from an OpenSearch index. Amazon OpenSearch Service accommodates various query types, including term, match, range, and Boolean queries.
- **Aggregation:** A process of grouping and summarizing data based on specified criteria. OpenSearch supports various aggregation types, such as bucket aggregations, metric aggregations, and pipeline aggregations.
- **Snapshot:** Refers to a point-in-time copy of the data in one or more indices. It is essentially a backup mechanism that captures the state of your data at a specific moment.

Design principles

The AWS Well-Architected Framework identifies a set of general design principles to facilitate good design in the cloud for building and managing OpenSearch domains.

- **High availability:** Provide high availability and fault tolerance by implementing Multi-Availability Zone (AZ) deployments. Distribute nodes across different Availability Zones to mitigate the impact of potential failures.
- **Scalability:** Design the Amazon OpenSearch Service domain to handle increasing loads and data volumes through horizontal scaling.
- **Monitoring and logging:** Implement comprehensive monitoring using tools like Amazon CloudWatch. Set up alerts for critical metrics, track system performance, and establish logging practices to facilitate troubleshooting and auditing.
- **Security best practices:** Enforce secure communication using encryption protocols such as Transport Layer Security (TLS). Deploy your domain within a Virtual Private Cloud (VPC) to avoid it being publicly accessible. Implement access controls, such as fine-grained access control, to enhance authentication, and authorization mechanisms to safeguard data and control access to the OpenSearch cluster.
- **Cost efficiency:** Optimize resource utilization to control costs. Optimize Amazon OpenSearch Service costs by selecting the most suitable instance type and configurations that match workload demands. Monitor and adjust resources as needed to maintain cost-effectiveness.
- **Performance optimization:** Optimize query performance by fine-tuning the OpenSearch configuration. Use index optimization techniques, caching mechanisms, and efficient shard allocation to enhance overall system responsiveness.
- **Data lifecycle management:** Define policies for data retention, archival, and deletion based on business requirements. Implement data lifecycle management practices to manage storage and maintain Amazon OpenSearch Service performance efficiently over time.
- **Data backup and recovery:** Establish a robust backup and recovery strategy, including regular snapshots of the Amazon OpenSearch Service indices. Test the restoration process to ensure data integrity and minimize downtime in case of data loss.
- **Disaster recovery planning:** Develop and test a comprehensive disaster recovery plan. Implement failover mechanisms and practice disaster recovery drills to validate the effectiveness of the plan in real-world scenarios.

- **Upgradability and compatibility:** Plan for regular upgrades of Amazon OpenSearch Service versions to benefit from new features, improvements, and security patches. Ensure compatibility with your applications and dependencies during the upgrade process.

Scenarios

This section explores common Amazon OpenSearch Service use cases and their implications for configuring domains. While these examples are not exhaustive, they cover common and widely employed scenarios and use cases. We provide background info, design guidance, and reference architectures for implementing Amazon OpenSearch Service scenarios.

Scenarios

- Log analytics workloads
- Search workloads

Log analytics workloads

Log analytics focuses on analyzing machine-generated time series data for insights into operations, security, and user behavior.

One of the common use cases for log analytics is monitoring and troubleshooting application logs. Consider a scenario where you use Amazon OpenSearch Service to analyze and gain insights from logs of a web application.

Use case: Web application log analysis

Scenario

Imagine you have a web application that serves users and generates logs in a standard format, such as JSON or plaintext.

Objectives

You want to use the Amazon OpenSearch Service to:

1. Monitor application performance.

2. Identify and troubleshoot errors quickly.

3. Gain insights into user behavior and application usage patterns.

Needed actions

1. **Collect logs:** Configure your web application to log relevant information, such as HTTP requests, response times, errors, and user interactions. Collect logs in a central location using a log shipper like Fluentbit or Amazon OpenSearch Service Ingestion or by sending them directly to Amazon OpenSearch Service.
2. **Index your logs in Amazon OpenSearch Service:** Define an OpenSearch index mapping that corresponds to the structure of your logs. For example, if your logs are in JSON format, create an index with appropriate mappings for each field. This allows OpenSearch to efficiently index and search through the logs.
3. **Search and analyze:** Enhance user experience through powerful search capabilities by using Amazon OpenSearch Service's full-text search capabilities. For instance, you can use the search capabilities to:
    a. **Identify errors**: Search for log entries with specific error codes or keywords to find issues quickly.
    b. **Monitor performance**: Analyze response times and track performance metrics over time to identify trends or anomalies.
    c. **Analyze user behavior**: Explore logs to understand user interactions, popular features, or potential areas for improvement.
4. **Create visualizations and dashboards:** Create visualizations and dashboards using Amazon OpenSearch Service Dashboard. Dashboards provide a centralized view for monitoring various aspects of your application.
5. **Set up alerting:** Use the Amazon OpenSearch Service alerting plugin to receive notifications when specific log patterns or anomalies are detected. This proactive approach helps in identifying and addressing issues before they impact users.

Search workloads

Full-text search capabilities empower customer applications within internal networks, including content management systems and legal documents. Additionally, they are used in internet-facing applications such as catalog search on ecommerce websites and content search. Let's explore a scenario in which Amazon OpenSearch Service is used in the context of an ecommerce platform.

Use case: E-commerce product search

Scenario

Imagine you're managing the search functionality for an ecommerce website. Your goal is to leverage Amazon OpenSearch Service for efficient and relevant product searches, providing users with a seamless shopping experience.

Needed actions

1. **Index product data:** Index your product catalog in Amazon OpenSearch Service. Each product is represented by a document containing attributes such as name, description, category and price. Define an appropriate mapping for the product index.
2. **Use full-text search:** Provide users the ability to search for products efficiently by using term matching queries with Amazon OpenSearch Service's full-text search capabilities. You can also implement a search-as-you-type feature to provide real-time suggestions as users type in the search bar.
3. **Use bucket and aggregation search:** Use aggregation and bucket search so that users can narrow down results by attributes like brand, category, size, and color. This enhances the user experience by providing filters to refine their search results.
4. **Sort and rank:** You can use OpenSearch's scoring capabilities to verify that search results are relevant and displayed in an order that meets user expectations.
5. **Add synonym and typo handling:** You can add synonym support and handle typos with features like fuzziness to provide relevant results to users despite synonyms or minor spelling mistakes.

Operational excellence

The operational excellence pillar offers guidance on effectively running and monitoring systems to deliver business value, while continually improving supporting processes and procedures. This involves maintaining high-quality Amazon OpenSearch Service domains, gaining operational insights, and continually improving processes to achieve organizational outcomes and values.

The subsequent questions and best practices complement those outlined in the Operational Excellence Pillar whitepaper.

Focus areas

- Design principles
- Operate
- Evolve
- Key AWS services
- Resources

Design principles

In addition to the AWS Well-Architected Framework whitepaper principles, the following design principles can help achieve operational excellence for your Amazon OpenSearch Service workloads:

- **Implement monitoring and alerting:** Set up comprehensive monitoring and alerting within OpenSearch deployments, featuring log analysis, performance tracking, and notification systems that provide real-time alerts for slow queries, security events, and critical errors.
- **Optimize index management:** Employ efficient index management strategies, such as using Index templates and removing unused indexes, to maintain a clean and efficient index namespace.
- **Implement a regular snapshot:** Implement regular snapshots using Index State Management (ISM) to ensure consistent backups of OpenSearch data.

Operate

AOSOPS01: How do you manage indexes in your OpenSearch Service domain?

A well-defined index strategy boosts performance, efficiency, and scalability by optimizing queries, managing storage, and ensuring secure access to data.

AOSOPS02: How do you maintain your snapshots?

A snapshot strategy is essential for Amazon OpenSearch Service domains, offering data resilience through automatic backups and point-in-time recovery capabilities. This enables disaster recovery and efficient domain management, while also facilitating seamless migration and upgrades with minimal downtime.

AOSOPS03: How do you implement real-time monitoring and alerting for your OpenSearch Service domain?

Comprehensive monitoring and alerts are essential for maintaining optimal search domain health, enabling early issue detection, performance optimization, and swift security response.

Best practices

- AOSOPS01-BP01 Employ Index State Management (ISM) to manage logs or time series data
- AOSOPS01-BP02 Configure index templates to automate index configuration upon creation
- AOSOPS01-BP03 Remove unused indexes
- AOSOPS02-BP01 Establish a manual snapshot repository
- AOSOPS02-BP02 Automate the process of taking snapshots
- AOSOPS03-BP01 Establish alarms for OpenSearch Service domain
- AOSOPS03-BP02 Configure notification services to receive monitoring alerts
- AOSOPS03-BP03 Enable search and indexing slow log functionality

- AOSOPS03-BP04 Enable audit logs for OpenSearch Service domains using fine-grained access control

AOSOPS01-BP01 Employ Index State Management (ISM) to manage logs or time series data

Manage large volumes of log or time series data efficiently with automated index lifecycle tasks.

Level of risk exposed if this best practice is not established: High

Desired outcome: ISM is employed to manage the life-cycle of logs or time series data.

Benefits of establishing this best practice:

- **Automate index lifecycle tasks:** Employing ISM can automate index lifecycle tasks such as alias rollovers, snapshots, storage tier transitions, and deletion of old indices, which helps reduce manual effort and improves efficiency and reliability of your domain.
- **Manage log or time series data:** ISM is particularly useful for managing large volumes of log or time series data by automating the process of transitioning old indices to lower-cost storage tiers or deleting them when they are no longer needed.

Implementation guidance

ISM automates index lifecycle tasks, including alias rollovers, snapshots, storage tier transitions, and deletion of old indices.

It is recommended that you review AOSPERF01 and AOSPERF02 to familiarize yourself with sharding strategies before you implement ISM policies.

Additionally, consult Index State Management in Amazon OpenSearch Service, Tutorial: Automating Index State Management processes and Index State Management pages for samples and full details about implementing ISM policies.

Implementation steps

- Open OpenSearch Dashboards for your domain.
- From the left sidebar, select **Index Management**, then **Create policy**.
- Use the visual editor or JSON editor to create policies. We recommend using the visual editor as it offers a more structured way of defining policies.

- After you create a policy, attach it to one or more indexes:

```
POST _plugins/_ism/add/my-index
        {
        "policy_id": "my-policy-id"
        }
```

Resources

- Index State Management in Amazon OpenSearch Service
- Tutorial: Automating Index State Management processes
- Index State Management

AOSOPS01-BP02 Configure index templates to automate index configuration upon creation

Use templates to automate index creation and maintain consistent settings.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: Index templates are configured to inherit settings and mapping, allowing unified structure across similar indices.

Benefits of establishing this best practice:

- **Automate index configuration:** Configuring index templates allows you to automate the configuration of indexes upon creation, eliminating the need for manual configuration and reducing errors.
- **Consistent index settings:** By specifying settings and mappings in an index template, you can verify that newly created indexes inherit consistent settings and mappings, which improves data management.
- **Simplified index management:** With index templates, you can simplify the process of creating new indexes by automating configuration tasks, making it easier to manage your OpenSearch Service domain.

Implementation guidance

Index templates can instruct OpenSearch on configuring an index upon creation. Configure these templates before creating indexes, which verifies that newly created indexes inherit the specified settings and mappings.

For samples and full details about implementing templates, see Index templates.

Implementation steps

- Open OpenSearch Dashboards for your domain.
- From the left sidebar, select **Index Management**, then **Templates**, then **Create template**.
- Define basic template settings:
  - **Template name:** Enter a unique name for the template (like `log-index-template`).
  - **Template type:** Choose **Indexes**
  - **Index patterns:** Specify one or more patterns to match indexes that the template will apply to. For example, `logs-*` for all indexes that begin with `logs-<something>`.
  - **Priority:** Set the template priority. Higher priority templates override lower ones when multiple templates match an index pattern.
  - Under Choose a method to define your templates, choose **Simple template** for simpler creation options.
  - If you're using aliases for your indices, then select an alias or create a new one.
  - Under Index settings box, specify the Number of primary shards. If you don't specify any number, then the default of one is going to be used.
  - Configure Number of replicas and the Refresh interval values.
  - If you are using a static mapping for your indices, then you can configure that as well under the Index mapping section.
  - Review and create the template.

Resources

- Index templates

AOSOPS01-BP03 Remove unused indexes

Regularly review and remove unused indexes to optimize your OpenSearch Service domain's performance and reduce costs. Unused indexes, even when inactive, continue to consume domain resources including memory, CPU, and storage.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: Unused indexes are removed to optimize performance and reduce costs.

Benefits of establishing this best practice:

- **Improve query performance:** Removing unused indexes can help reduce the computational resources needed to manage your OpenSearch infrastructure, leading to improved query performance and faster search results.
- **Lower costs:** By reducing the number of indices that need to be managed, you can also lower costs associated with storing and processing data in Amazon OpenSearch Service.
- **Enhance domain efficiency:** Regularly reviewing and pruning unused indexes helps improve your OpenSearch Service domain efficiency, scalability, and cost-effectiveness, making it easier to manage and maintain your OpenSearch infrastructure.

Implementation guidance

To optimize the performance and cost-effectiveness of your OpenSearch Service domains, remove any unused indexes that are no longer required. This can help reduce the computational resources needed to manage your OpenSearch Service infrastructure, leading to improved query performance and lower costs. By regularly reviewing and pruning unused indexes, you improve your OpenSearch Service domain efficiency, scalability, and cost-effectiveness.

Implementation steps

- **Identify unused indexes:** Unused indexes are typically those that haven't been updated or queried for a specified period. To identify them, you can use OpenSearch Dashboards or the OpenSearch API.
- **Option 1:** Use OpenSearch Dashboards by going to Index Management, then Indexes, then Indexes.
- **Option 2:** Use the OpenSearch API:

- Use the `_cat/indices` API to list all indexes along with their name, status and size.

```
GET _cat/indices?v
```

- Once you identify unneeded indices, delete them using the Dashboard or with the API command:

```
DELETE /<index_name
```

- You can also create an ISM policy to delete indexes after they reach a certain age. For details on how to create an ISM policy, see AOSOPS01-BP01.

Resources

- Delete index

AOSOPS02-BP01 Establish a manual snapshot repository

Maintain control over your OpenSearch Service data by creating a centralized manual snapshot repository.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: Complete control and flexibility over OpenSearch Service domain data through centralized manual snapshot management.

Benefits of establishing this best practice: By establishing a manual snapshot repository, you maintain full control over your OpenSearch Service data, which helps you comply with regulatory requirements and data retention policies. Additionally, you can simplify disaster recovery processes by having a centralized repository for snapshots.

Implementation guidance

Establishing a manual snapshot repository for stored snapshots in OpenSearch Service enables full control over domain data, promotes manual backup and recovery of indices and documents, and provides a centralized location for storing copies of critical data, which can be restored into multiple OpenSearch Service domains.

Amazon OpenSearch Service automatically creates hourly snapshots of your indices within the same domain. However, these automatic snapshots are not restorable across different domains.

If you rely solely on the automatic hourly snapshot, there's no need to follow this recommendation. However, if you prefer to take control of your snapshot schedule and have the flexibility to restore them in other OpenSearch Service domains, then continue following this best practice.

To implement this best practice, use an S3 bucket that exists in the same AWS Region that hosts your OpenSearch Service domain. You also need the necessary permissions to manage the S3 bucket and run administrative commands in your OpenSearch Service domain.

Implementation steps

To create an ISM policy for snapshots:

- **Understand snapshot types:** Familiarize yourself with the two available snapshot options in Amazon OpenSearch Service: *automatic* and *manual*.
- **Set up manual snapshots:** If you choose to use manual snapshots:
  - Create an IAM role with the necessary permissions
  - Set up a custom Amazon S3 bucket for storing your snapshots
  - Register the S3 bucket in your OpenSearch Service domain
  - Test the repository by taking a manual snapshot and restoring it
- **Create an ISM policy:** Design and implement an ISM policy that captures snapshots of your most critical indices, storing them securely in your custom S3 bucket.

Resources

- Creating index snapshots in Amazon OpenSearch Service
- Registering a manual snapshot repository
- Take manual snapshots and restore in a different domain spanning across various Regions and accounts in Amazon OpenSearch Service

AOSOPS02-BP02 Automate the process of taking snapshots

Schedule automatic snapshots to maintain consistent backups and data recoverability.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: Snapshots are automatically created.

Benefits of establishing this best practice: To streamline snapshot management, use these features to schedule automatic snapshots at regular intervals, which allows for consistent data backups and recoverability in case of issues. By automating this process, you can reduce the risk of human error and minimize the impact of any unexpected events on your OpenSearch Service domains.

Implementation guidance

Amazon OpenSearch Service automatically creates hourly snapshots of your indices within the same domain. However, these automatic snapshots are not restorable across different domains.

If you rely solely on the automatic hourly snapshot, there's no need to follow this recommendation. However, if you prefer to take control of your snapshot schedule and have the flexibility to restore them in other OpenSearch Service domains, then continue following this best practice.

Implementation steps

Create a Snapshot Management (SM) policy:

- Create a custom snapshot repository. For step-by-step instructions, see AOSOPS02-BP01.
- Navigate to your OpenSearch Service domain dashboard.
- In your OpenSearch Service dashboard, select **Snapshot Management** from the left-hand menu.
- Create a new **Snapshot Policy** by choosing **Create policy**.
- Configure policy details:
  - **Name**: Give your policy a unique name, for example `daily-snapshot-policy`.
  - Under Source and Destination boxes, select the indices that you want to snapshot, and choose the snapshot repository you registered (for example, `your-repository-name`).
- In the Snapshot schedule box:
  - **Snapshot frequency:** Enter how often you want snapshots taken. For example, choose daily if you want to back up every day.
  - **Start time and Time zone:** Enter the time of day and the time zone to take the snapshots.
- (Optional) Define a retention policy to automatically delete older snapshots after a specific time period (for example, keep the last seven days).
- Review the settings, then choose **Create policy** to save.

Once the policy is active, OpenSearch Service will automatically take snapshots based on your specified schedule, and they'll be saved in the designated Amazon S3 bucket.

Resources

- Automating snapshots with Snapshot Management
- Unleash the power of Snapshot Management to take automated snapshots using Amazon OpenSearch Service
- Take manual snapshots and restore in a different domain spanning across various regions and accounts in Amazon OpenSearch Service

AOSOPS03-BP01 Establish alarms for OpenSearch Service domain

Set up alerts to receive timely notifications about potential issues that may impact performance or availability.

Level of risk exposed if this best practice is not established: High

Desired outcome: Receive timely, relevant, and actionable alerts for rapid identification and mitigation of potential issues, especially when KPI outcomes are at risk.

Benefits of establishing this best practice:

- **Prompt notification:** Configure OpenSearch Service domains with recommended Amazon CloudWatch alarms to receive notifications when critical conditions occur, such as a cluster health status remaining in a critical state for more than one minute.
- **Enable corrective action:** By implementing automatic actions, such as sending email notifications, you can take corrective action to address issues before they impact your OpenSearch Service domain's performance or availability.
- **Improve domain reliability:** Regular monitoring and notification through CloudWatch alarms helps your OpenSearch Service domain to remain reliable, available, and performing optimally, meeting the requirements of your users and applications.

Implementation guidance

It's highly recommended to implement the Recommended Amazon CloudWatch alarms for your Amazon OpenSearch Service domains.

Use alarms to initiate automated actions when specific conditions are met, such as when a metric exceeds a predetermined threshold over a set duration. For example, if your cluster health status remains in a critical state (indicated by a red status) for more than one minute, you can configure your monitoring tool to send an email notification, which prepares you to take swift corrective action.

To set up essential Amazon CloudWatch alarms for your OpenSearch Service domain, follow a multi-step process that involves two key components:

- **Understanding alarms:** Familiarize yourself with the recommended alarms and their critical thresholds to stay aware of the most significant metrics to monitor. Recommended CloudWatch alarms for Amazon OpenSearch Service details the recommended alarms and their thresholds.
- **Creating alarms:** Follow the instructions in Amazon CloudWatch alarms and How do I use CloudWatch alarms to monitor my OpenSearch Service cluster to create alarms that meet your specific needs.

Resources

- Recommended CloudWatch alarms for Amazon OpenSearch Service
- Amazon CloudWatch alarms
- How do I use CloudWatch alarms to monitor my OpenSearch Service cluster

AOSOPS03-BP02 Configure notification services to receive monitoring alerts

Connect monitoring alerts to your messaging infrastructure to stay informed about domain changes and potential issues.

Level of risk exposed if this best practice is not established: High

Desired outcome: Alert notifications from monitoring systems are integrated with messaging infrastructure, enabling timely response to issues.

Benefits of establishing this best practice:

- Improved ability to stay on top of changes and potential issues
- Enhanced visibility into domain performance, security, and compliance

Implementation guidance

Amazon SNS provides a built-in integration with Amazon CloudWatch to receive notifications when an alarm breaches a threshold. You can receive notifications for events through multiple channels, including email, Amazon Q Developer in chat applications chat notifications, or AWS Management Console Mobile Application push notifications.

To set up essential Amazon CloudWatch alarms for your OpenSearch Service domain, follow AOSOPS03-BP01 and Notifying users on alarm changes.

Resources

- Notifying users on alarm changes

AOSOPS03-BP03 Enable search and indexing slow log functionality

Turn on slow log functionality to gain insights into query latency and optimize search and indexing operations.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You use slow logs for search and indexing operations, providing a detailed view of query latency and enabling optimization efforts.

Benefits of establishing this best practice:

- **Optimize queries:** Slow logs provide detailed information about slow or long-running queries, which helps you identify areas for optimization and make changes to improve performance.
- **Troubleshoot issues:** By capturing logs of slow searches, indexing operations, and other queries, slow logs help you troubleshoot issues more effectively, reducing downtime and improving overall efficiency in your OpenSearch Service domains.

Implementation guidance

Search slow logs, indexing slow logs, and error logs are valuable for diagnosing performance and stability issues. Audit logs record user activity for compliance purposes.

For a detailed guide on enabling logging slow index and search operations, see AOSPERF04-BP01.

Resources

- Monitoring OpenSearch logs with Amazon CloudWatch Logs

AOSOPS03-BP04 Enable audit logs for OpenSearch Service domains using fine-grained access control

Turn on audit logging with access control to gain visibility into domain operations, support issue resolution, and enhance security and compliance.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: Achieve clear visibility into your domain's operations, enabling quicker issue resolution and better security and compliance.

Benefits of establishing this best practice:

- Improved security and compliance
- Enhanced ability to monitor and audit user activity

Implementation guidance

Audit logs help monitor user activity on your OpenSearch clusters, capturing details such as authentication successes and failures, OpenSearch requests, index modifications, and search queries.

Implementation steps

Enabling audit logs for your domain involves a multi-step process that includes:

- Enabling audit logs in your OpenSearch Service domain.
- Creating a CloudWatch Log group or choosing an existing one.
- Creating an access policy.
- Activating the audit logs in your OpenSearch Dashboards.

For a detailed step-by-step guide on enabling audit logs for your OpenSearch Service Domain, see How do I activate audit logs in OpenSearch Service and Monitoring audit logs in Amazon OpenSearch Service.

Resources

- Monitoring audit logs in Amazon OpenSearch Service

Evolve

AOSOPS04: How do you proactively troubleshoot issues?

Familiarity with troubleshooting common issues in an OpenSearch domain is important because it helps you quickly identify and resolve problems, minimizing impact on domain availability. Rapid resolution is essential to reduce downtime and maintain service continuity for applications relying on the OpenSearch domain.

Understanding how to troubleshoot performance issues helps you optimize the overall performance of your OpenSearch domain. By identifying bottlenecks, slow queries, or resource-intensive operations, you can make informed decisions to enhance performance.

Best practices

- AOSOPS04-BP01 Train staff on common OpenSearch issues and how to remediate them

AOSOPS04-BP01 Train staff on common OpenSearch issues and how to remediate them

Educate staff on common OpenSearch Service issues and how to resolve them, which fosters proactive issue resolution and reduces downtime.

Level of risk exposed if this best practice is not established: High

Desired outcome: Your staff learns common issues in the OpenSearch Service domain and proactively resolves them.

Benefits of establishing this best practice:

- Improved availability and uptime of the domain
- Reduced downtime and impact on users

- Enhanced ability to identify and troubleshoot critical issues

Implementation guidance

Implement operational runbooks, as they can significantly decrease the time required to restore services when encountering issues.

Implementation steps

To resolve common issues within the OpenSearch Service domain, follow these steps:

- **Identify the issue:** Analyze logs and monitoring data to identify the specific issue or problem affecting your OpenSearch Service.
- **Check for known errors:** Review AWS documentation and OpenSearch release notes to see if there are any known issues or errors that may be causing the problem.
- **Verify configuration over time:** Conduct periodic reviews of your OpenSearch configuration to validate its integrity over time. This includes scrutinizing settings like index templates, mappings, IAM policies, user permissions, and query patterns to confirm that they remain accurate and free from errors.
- **Monitor performance:** Use Amazon CloudWatch metrics and logs to monitor OpenSearch Service performance and identify potential bottlenecks or issues.
- **Troubleshoot indexing:** If the issue is related to indexing, review the indexing process, including the number of shards, replicas, and indexing rate, to identify any potential causes.
- **Consult AWS documentation:** Refer to AWS documentation, like the OpenSearch Service User Guide, for guidance on troubleshooting common issues and resolving errors.
- **Reach out to support:** If you're unable to resolve the issue yourself, reach out to Support or your organization's internal support team for assistance.

Resources

- Troubleshooting Amazon OpenSearch Service

Key AWS services

- Amazon OpenSearch Service: A fully managed service that helps you search, ingest, and analyze data in real-time.

- Amazon CloudWatch: Monitoring and logging service that helps you monitor and troubleshoot your AWS resources.
- Amazon SNS: A fully managed messaging service that allows you to fan out messages to multiple subscribers.
- Amazon CloudWatch alarms: A feature of CloudWatch that allows you to set up alarms based on metrics and receive notifications when thresholds are breached.

Resources

- Learn how to add dedicated leader nodes in your OpenSearch Service domain
- Making configuration changes in Amazon OpenSearch Service
- OpenSearch Service domain and instance quotas
- Amazon OpenSearch Service prices
- Monitoring OpenSearch logs with Amazon CloudWatch Logs
- Monitoring audit logs in Amazon OpenSearch Service
- Recommended CloudWatch alarms for Amazon OpenSearch Service
- Notifying users on alarm changes
- Index State Management in Amazon OpenSearch Service
- Index templates
- Delete index
- Creating index snapshots in Amazon OpenSearch Service
- Automating snapshots with Snapshot Management
- Troubleshooting Amazon OpenSearch Service

Security

The security pillar includes the ability to protect information, systems, and assets while delivering business value. This section provides in-depth, best practice guidance for architecting secure OpenSearch domains.

Focus areas

- Design principles
- Security foundation
- Detection
- Data protection
- Key AWS services
- Resources

Design principles

In addition to the security principles of the AWS Well-Architected Framework, the following design principles can enhance the security posture of OpenSearch workloads:

- **Maintain traceability of your OpenSearch domains:** Monitor configuration changes to your domain, track user activity, and audit requests for data--including detailed connection attributes. Use AWS CloudTrail logging and OpenSearch audit logs to monitor the use of configuration APIs and requests to your data.
- **Maintain perimeter security for your domain:** Secure the perimeter to your domain by using AWS identity and resource policies to associate identities and resources to specific allow/deny actions. Create logically isolated networks using an Amazon Virtual Private Cloud (VPC), and Amazon VPC security groups to allow traffic only from known entities.
- **Protect access to sensitive data:** Secure access to your sensitive or confidential data using advanced security controls. Use index, document, or field-level security to limit access to specific indices, documents, or fields.
- **Implement least privilege access controls:** Manage user access and monitor cluster configuration by using access control features like IAM policies or fine-grained access control.

- **Apply security updates regularly:** Protect your data from security vulnerabilities. To minimize the need for version upgrades, OpenSearch Service provides backward compatible security patches and upgrades for all supported versions of OpenSearch and OpenSearch.
- **Maintain compliance requirements:** OpenSearch Service maintain compliance with several industry standards, including SOC, PIC and HIPAA. These validations can help you meet your organization's compliance and governance requirements.

Security foundation

AOSSEC01: How do you implement robust security measures to protect your OpenSearch Service domain?

As organizations increasingly rely on cloud-based search services, the importance of securing their data and keeping it private cannot be overstated. Given these services' powerful search and analytics capabilities, it is crucial for administrators to proactively safeguard their search domains from unauthorized access, tampering, and other malicious activities. Effective security measures can mitigate several risks, making it essential for organizations to implement a robust security approach for their search service infrastructure.

Best practices

- AOSSEC01-BP01 Launch your OpenSearch Service domains within a Virtual Private Cloud
- AOSSEC01-BP02 Activate node-to-node encryption
- AOSSEC01-BP03 Enable encryption at rest
- AOSSEC01-BP04 Encrypt slow and error logs in Amazon CloudWatch to protect sensitive information

AOSSEC01-BP01 Launch your OpenSearch Service domains within a Virtual Private Cloud

Host OpenSearch Service domains in a Virtual Private Cloud (VPC) for improved security, isolation, and network control.

Level of risk exposed if this best practice is not established: High

Desired outcome: Your organization launches its OpenSearch Service domain within a VPC.

Benefits of establishing this best practice:

- Improved security and isolation of the domain
- Protect network perimeter of your OpenSearch Service domain reducing risks of public exposures
- Enhanced ability to control and monitor network traffic

Implementation guidance

You can launch an OpenSearch Service domain in a VPC. A VPC is a dedicated virtual network that is logically isolated from other virtual networks in the AWS Cloud and is specific to your AWS account. By placing an OpenSearch Service domain within a VPC, you enable secure communication between OpenSearch Service and other services within the VPC without requiring an internet gateway, NAT device, or VPN connection. All traffic remains securely within the AWS Cloud, with no need for external connectivity.

When creating an OpenSearch Service domain, you must choose whether it will have a public endpoint or reside within a VPC. This decision cannot be changed afterwards, as doing so would require that you create a new domain and either manually reindex or migrate your data. However, using a custom snapshot repository can simplify this process, as you can set up an S3 bucket as a snapshot repository on both domains and then transfer snapshots between them. For detailed instructions on implementing custom repositories, see Creating index snapshots in Amazon OpenSearch Service, Registering a manual snapshot repository, and Take manual snapshots and restore in a different domain spanning across various Regions and accounts in Amazon OpenSearch Service.

Implementation steps

To create a domain inside a VPC, do the following:

- **Create a VPC:** See Working with VPCs in the Amazon VPC User Guide. If you already have a VPC, you can skip this step.
- **Reserve IP addresses:** OpenSearch Service enables the connection of a VPC to a domain by placing network interfaces in a subnet of the VPC. Each network interface is associated with an IP address. You must reserve a sufficient number of IP addresses in the subnet for the network interfaces. For more information, see Reserving IP addresses in a VPC subnet.

- Navigate to the Amazon OpenSearch Service console.
- To create a new domain, choose **Create domain**.
- In the Network box, select **VPC access - recommended**.
- The IP address type – new option provides a choice between having a dual-stack mode, in which IPv6 and IPv4 are enabled and your resources can communicate using any of them, or an IPv4-only option. We recommend you to use Dual-stack mode - recommended option.
- Under the VPC option, choose the VPC you want your OpenSearch Service domain to reside in.
- Continue with other options, then review and choose **Create**.

Resources

- Launching your OpenSearch Service domains within a VPC
- Creating index snapshots in Amazon OpenSearch Service
- Registering a manual snapshot repository
- Take manual snapshots and restore in a different domain spanning across various Regions and accounts in Amazon OpenSearch Service
- Working with VPCs
- Reserving IP addresses in a VPC subnet

AOSSEC01-BP02 Activate node-to-node encryption

Activate encryption for secure transmission of data between nodes, protecting it from unauthorized access.

Level of risk exposed if this best practice is not established: High

Desired outcome: Node-to-node encryption is activated for OpenSearch Service domains, providing full data protection in transit.

Benefits of establishing this best practice: Improved security and confidentiality of data in transit.

Implementation guidance

Node-to-node encryption add an extra layer of security to the inherent security features of OpenSearch Service. It incorporates Transport Layer Security (TLS) to secure all communications between nodes.

Each OpenSearch Service domain, regardless of whether the domain uses VPC access, resides within its own dedicated VPC. This architecture protects traffic between OpenSearch nodes from public access. However, traffic within the VPC is unencrypted by default. Node-to-node encryption enables TLS 1.2 encryption for all communications within the VPC.

If you send data to OpenSearch Service over HTTPS, node-to-node encryption keeps your data encrypted as OpenSearch distributes (and redistributes) it throughout the cluster. If data arrives unencrypted over HTTP, OpenSearch Service encrypts it after it reaches the cluster. You can require that all traffic to the domain arrive over HTTPS using the console, AWS CLI, or configuration API.

If you enable fine-grained access control, node-to-node encryption is required.

Implementation steps

- Navigate to the Amazon OpenSearch Service console.
- Create a new domain, or modify an existing domain:
  - For a new domain, choose **Create domain**. For an existing domain, select the domain name and choose **Actions**, then **Edit security configuration**.
  - If fine-grained access control is enabled, all encryption options under the Encryption box will be enabled by default and cannot be disabled.
  - If you don't have fine-grained access control enabled, then you can enable Node-to-node encryption located in the Encryption box.
  - Choose **Save changes**.

Resources

- Node-to-node encryption for Amazon OpenSearch Service

AOSSEC01-BP03 Enable encryption at rest

Protect data stored in OpenSearch by enabling encryption, keeping your data confidential even when not in transit.

Level of risk exposed if this best practice is not established: High

Desired outcome: Improving data protection at rest and helping you to meet compliance requirements for data protection.

Benefits of establishing this best practice: Improved security and confidentiality of data at rest.

Implementation guidance

OpenSearch Service domains offer encryption of data at rest, a security feature that helps prevent unauthorized access to your data. The feature uses AWS Key Management Service to store and manage your encryption keys and the Advanced Encryption Standard algorithm with 256-bit keys (AES-256) to perform the encryption. If enabled, the feature encrypts the following aspects of a domain:

- All indexes (including those in UltraWarm storage)
- OpenSearch logs
- Swap files
- All other data in the application directory
- Automated snapshots

Implementation steps

- Navigate to the Amazon OpenSearch Service console.
- Create a new domain or modify an existing domain:
  - For a new domain, choose **Create domain**. For an existing domain, select the domain name and choose **Actions**, then **Edit security configuration**.
  - If fine-grained access control is enabled, all encryption options under the Encryption box will be enabled by default and cannot be disabled.
  - If you don't have fine-grained access control enabled, then you can enable Enable encryption of data at rest located in the Encryption box.
  - Choose **Save changes**.

Resources

- Encryption of data at rest for Amazon OpenSearch Service

AOSSEC01-BP04 Encrypt slow and error logs in Amazon CloudWatch to protect sensitive information

Protect sensitive information in slow and error logs by encrypting them, keeping them confidential even when stored in Amazon CloudWatch.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: Slow and error logs are encrypted. This protects your logs from exposing information such as document fields and queries.

Benefits of establishing this best practice: Improved security and confidentiality of sensitive information.

Implementation guidance

If you're publishing slow and error logs to Amazon CloudWatch, you can protect them with encryption by using an AWS KMS key to secure your CloudWatch Logs log group. To achieve this, create an AWS KMS key, set permissions on the key, associate it with a log group. After this is completed, you can use the same AWS KMS key with your OpenSearch Service domain.

For step-by-step instructions on implementing this encryption process, see Encrypt log data in CloudWatch Logs using AWS Key Management Service and How do I use AWS KMS to encrypt log data in CloudWatch Logs?

Resources

- Encrypt log data in CloudWatch Logs using AWS Key Management Service

Detection

AOSSEC02: How do you track and audit user and API access to your OpenSearch Service domains?

To protect the security and integrity of your data, it's essential to monitor and control access to your OpenSearch Service domains. This includes tracking and auditing access to prevent

unauthorized changes or data breaches. Implement robust access controls and monitor usage patterns to quickly detect and respond to potential threats, securing your search infrastructure and sensitive data.

Best practices

- AOSSEC02-BP01 Set up audit logging for OpenSearch Service domains that use fine-grained access control
- AOSSEC02-BP02 Track OpenSearch Service API calls
- AOSSEC02-BP03 Monitor real-time events in your OpenSearch Service domains
- AOSSEC02-BP04 Assess your OpenSearch Service domain's configuration with AWS Config

AOSSEC02-BP01 Set up audit logging for OpenSearch Service domains that use fine-grained access control

Turn on audit logging for OpenSearch Service domains using fine-grained access control, enhancing security monitoring and compliance.

Level of risk exposed if this best practice is not established: High

Desired outcome: Audit logs are enabled on OpenSearch Service domains with fine-grained access control enabled.

Benefits of establishing this best practice:

- **Improved security monitoring:** Enabling audit logs on OpenSearch Service domains with fine-grained access control enabled allows for improved security monitoring and tracking of user activity.
- **Enhanced compliance:** By customizing the audit log settings to meet specific needs, organizations can enhance compliance with regulatory requirements by maintaining a detailed record of user actions and activities within their OpenSearch clusters.

Implementation guidance

If your OpenSearch Service domain uses fine-grained access control, you can enable audit logs for your data. Audit logs are highly customizable and let you track user activity on your OpenSearch clusters, including authentication success and failures, requests to OpenSearch, index changes,

and incoming search queries. The default configuration tracks a popular set of user actions, but we recommend tailoring the settings to your exact needs.

For details on how to enable audit logs for your OpenSearch Service domain, see AOSOPS03-BP04.

Resources

- Monitoring audit logs in Amazon OpenSearch Service

AOSSEC02-BP02 Track OpenSearch Service API calls

Monitor and log all API calls made to your OpenSearch Service, which provides visibility into access and empowers your team to take swift action against unauthorized activity.

Level of risk exposed if this best practice is not established: High

Desired outcome: OpenSearch Service API calls are tracked and logged.

Benefits of establishing this best practice:

- Improved visibility and control of access to sensitive data
- Enhanced ability to detect and respond to unauthorized access

Implementation guidance

Amazon OpenSearch Service seamlessly integrates with AWS CloudTrail, which logs actions performed by users, roles, or AWS services within OpenSearch Service. The captured calls include calls from the OpenSearch Service console, AWS CLI, or an AWS SDK. If you create a trail, you can enable continuous delivery of CloudTrail events to an S3 bucket, including events for OpenSearch Service. If you don't configure a trail, you can still view the most recent events on the CloudTrail console in Event history.

Using the information collected by CloudTrail, you can determine the request that was made to OpenSearch Service, the IP address from which the request was made, who made the request, when it was made, and other details.

All OpenSearch Service configuration API actions are logged by CloudTrail and are documented in the Amazon OpenSearch Service API Reference. For detail on Amazon OpenSearch Service

log entries in AWS CloudTrail, see Monitoring Amazon OpenSearch Service API calls with AWS CloudTrail.

AOSSEC02-BP03 Monitor real-time events in your OpenSearch Service domains

Track real-time events in your OpenSearch Service domains using Amazon EventBridge, enabling automated actions and notifications to improve visibility and response times.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: Real-time events in OpenSearch Service domains are tracked and monitored according to your requirements. Automated actions and notifications are configured for relevant events if required.

Benefits of establishing this best practice:

- **Enhanced monitoring:** You can use Amazon EventBridge to monitor real-time events in OpenSearch Service domains to receive notifications about specific events that affect your domains, improving overall visibility and awareness.
- **Automated actions:** By creating rules to determine which events are relevant and defining automated actions to take when a rule initiates, organizations can streamline responses to domain-related events and reduce the risk of manual errors or delays.

Implementation guidance

Amazon OpenSearch Service integrates with Amazon EventBridge, which provides notifications about specific events that affect your domains in near real-time. This integration also provides a transition from Amazon CloudWatch Events.

With this setup, you can create simple rules to determine which events are relevant to you and define automated actions to take when a rule occurs. Amazon EventBridge can notify you of various events related to your OpenSearch Service domain, such as software updates, Auto-Tune activities, cluster health changes, VPC endpoint modifications, node retirements, and domain errors.

To create a Lambda function that listens to these events, see Tutorial: Listening for Amazon OpenSearch Service EventBridge events and Tutorial: Sending Amazon SNS alerts for available software updates.

Resources

- Monitoring OpenSearch Service events with Amazon EventBridge

AOSSEC02-BP04 Assess your OpenSearch Service domain's configuration with AWS Config

Use AWS Config to evaluate your OpenSearch Service domain's configuration settings, gaining visibility into potential issues or areas for improvement and enhancing compliance and security.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You assess OpenSearch Service domains' configuration settings using AWS Config.

Benefits of establishing this best practice:

- **Improved configuration visibility**: Assessing OpenSearch Service domains' configuration settings using AWS Config provides visibility into domain configurations, helping you identify potential issues or areas for improvement.
- **Enhanced compliance and security**: By using AWS Config to detect security risks or compliance issues, organizations can take corrective actions to meet their requirements, which creates a more secure and compliant setup for their OpenSearch Service domains.

Implementation guidance

To maintain optimal configuration and alignment with your organization's security and compliance requirements, it's essential to regularly assess the settings of your OpenSearch Service domains. To do this, use AWS Config, which provides a service that automatically evaluates and reports on the configuration of your AWS resources, including your OpenSearch Service domains.

With AWS Config, you can gain visibility into your domain configurations, identify potential security risks or compliance issues, and take corrective actions to meet your organization's requirements. To set up AWS Config for compliance notifications, see Getting Started with AWS Config, List of AWS Config Managed Rules and Security Best Practices for Amazon OpenSearch Service. These resources provide step-by-step instructions on how to configure AWS Config to send compliance notifications and meet your organization's security and compliance standards.

Data protection

AOSSEC03: How do you protect your data, indices, and documents?

To protect the security and integrity of your data, indices, and documents on OpenSearch Service, it's essential to implement robust security measures. Fine-grained access control in OpenSearch Service provides additional control over data access. For instance, you can tailor search results based on the requester, displaying results from specific indexes or concealing certain fields in documents.

Best practices

- AOSSEC03-BP01 Implement fine-grained access control to manage access to your data on Amazon OpenSearch Service
- AOSSEC03-BP02 Secure your indices, documents, and fields using fine-grained access control

AOSSEC03-BP01 Implement fine-grained access control to manage access to your data on Amazon OpenSearch Service

Control user access to OpenSearch Service domains and dashboards using fine-grained access control, and only provide sensitive data access to authorized users.

Level of risk exposed if this best practice is not established: High

Desired outcome: User access to OpenSearch Service domains and Dashboards is controlled using fine-grained access control.

Benefits of establishing this best practice:

- **Enhanced data security**: Controlling user access to OpenSearch Service domains and Dashboards using fine-grained access control verifies that sensitive data is only accessible to authorized users, maintaining a high level of security.
- **Improved compliance**: By implementing fine-grained access control, organizations can meet regulatory requirements by providing precise control over who has access to specific indexes, documents, or fields within OpenSearch Service, reducing the risk of non-compliance.

Implementation guidance

For added control over who can access your data on OpenSearch Service, fine-grained access control provides several options. For instance, you might need to restrict search results to only one index based on the user making the request or hide specific fields in documents or exclude certain documents entirely.

Fine-grained access control offers a range of benefits, including:

- **Role-based access control:** Provides tailored permissions based on individual roles
- **Security at multiple levels:** Index, document, and field, giving you precise control over data access
- **Multi-tenancy support in OpenSearch Dashboards:** Create separate, secure environments for different users or organizations
- **HTTP basic authentication for both OpenSearch and OpenSearch Dashboards:** Provides an additional layer of security.

To understand key concepts and features, see Fine-grained access control in Amazon OpenSearch Service.

Implementation steps

To enable fine-grained access control on your domain:

- Navigate to the Amazon OpenSearch Service console.
- Create a new domain or modify an existing domain:
  - For a new domain, choose **Create domain**. For an existing domain, select the domain name and choose **Actions**, then **Edit security configuration**.
  - **For new domains:** If you choose Easy create under Domain creation method box, then fine-grained access control will be enabled by default, and you can't change it. However, if you choose Standard create, you have more options available, such as Enable fine-grained access control and selecting a master user. You can choose the master user to be an IAM ARN or a normal user with a username and password. For a simple setup, choose **Create master user**.
  - **For existing domains:** Choose **Enable fine-grained access control** located under Fine-grained access control box. You can choose the master user to be an IAM ARN or a normal user with a username and password. For a simple setup, choose **Create master user**.
- Continue with other desired options

- Choose **Create** or **Save changes**.

AOSSEC03-BP02 Secure your indices, documents, and fields using fine-grained access control

Protect sensitive data in Amazon OpenSearch Service by implementing fine-grained access control to secure indices, documents, and fields.

Level of risk exposed if this best practice is not established: High

Desired outcome: Indices, documents, and fields on Amazon OpenSearch Service are secured using fine-grained access control.

Benefits of establishing this best practice:

- **Improved data security**: Implementing fine-grained access control to secure indices, documents, and fields in Amazon OpenSearch Service verifies that sensitive data is only accessible to authorized users.
- **Enhanced compliance**: By setting up strict access controls on specific data using fine-grained access control, organizations can meet regulatory requirements and reduce the risk of non-compliance.

Implementation guidance

With fine-grained access control, you can implement strict access controls on your indices, documents, and fields in Amazon OpenSearch Service. This verifies that only authorized users have access to specific data.

To set up these controls, navigate through OpenSearch Dashboards to create roles, map users, configure permissions, and define filter queries. For a detailed step-by-step guide on how to implement field-level security or document-level security, see Field-level security in Amazon OpenSearch Service, Document-level security and Field-level security.

Resources

- Document-level security (DLS)

Key AWS services

- Amazon OpenSearch Service: A fully managed service that helps you search, ingest, and analyze data in real-time.
- Amazon Virtual Private Cloud (Amazon VPC): A virtual network dedicated to your AWS account, which you can use to launch AWS resources in a virtual environment.
- AWS Key Management Service (AWS KMS): A service that helps you manage the encryption and decryption of your data.
- Amazon CloudWatch: Monitoring and logging service that helps you monitor and troubleshoot your AWS resources.
- AWS CloudTrail: Provides a record of all API calls made within your AWS account.
- Amazon EventBridge: A service that helps you respond to events in your AWS resources and applications.
- AWS Config: Provides a detailed view of your AWS resources and helps you ensure they are configured in compliance with your organization's security and compliance requirements.

Resources

Blogs and documentation:

- Node-to-node encryption for Amazon OpenSearch Service

- Encryption of data at rest for Amazon OpenSearch Service
- Encrypt log data in CloudWatch Logs using AWS Key Management Service
- Launching your Amazon OpenSearch Service domains within a VPC
- Fine-grained access control in Amazon OpenSearch Service
- Document-level security (DLS)
- Monitoring audit logs in Amazon OpenSearch Service
- Monitoring Amazon OpenSearch Service API calls with AWS CloudTrail
- Monitoring OpenSearch Service events with Amazon EventBridge
- List of AWS Config Managed Rules

Videos:

- Securing Your Log Analytics and Search Data with Amazon OpenSearch Service

Reliability

The reliability pillar focuses on verifying that your Amazon OpenSearch Service workload performs correctly and consistently when needed. This includes operating and testing the workload throughout its entire lifecycle. This paper provides best practices for building and maintaining reliable OpenSearch Service workloads.

Focus areas

- Design principles
- Foundations
- Workload architecture
- Key AWS services
- Resources

Design principles

- **Implement monitoring and notifications:** Set up monitoring and notifications for OpenSearch domain failures to ensure timely detection and response to potential issues.
- **Provide continuous availability:** Implement multi-AZ with standby deployment, Index State Management (ISM) policy, index replication, cross-cluster replication, and regular quota monitoring to ensure OpenSearch Service domain availability.
- **Implement a disaster recovery strategy:** Develop a Disaster Recovery (DR) strategy for OpenSearch Service, including regular updates to the latest version, and monitoring of software updates.
- **Stay informed about updates:** Stay informed about OpenSearch Service update notifications, keep domains updated to the latest version, and monitor software updates for reliability and security.

Foundations

AOSREL01: How do you keep your OpenSearch version up to date?

OpenSearch Service consistently adds support for community-maintained OpenSearch versions. It is recommended to always upgrade to the latest available OpenSearch versions to benefit from performance improvements, security upgrades, and bug fixes.

Best practices

- AOSREL01-BP01 Implement a system update notification strategy
- AOSREL01-BP02 Regularly update your OpenSearch Service domain to the latest version

AOSREL01-BP01 Implement a system update notification strategy

Stay informed about notifications regarding updates to Amazon OpenSearch Service, which prepares you for changes or new features.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You stay informed about notifications regarding updates to Amazon OpenSearch Service.

Benefits of establishing this best practice: Staying informed about updates to Amazon OpenSearch Service, can help you anticipate and prepare for any changes or new features in the service.

Implementation guidance

We recommend setting up alarms receive notifications on updates to the OpenSearch Service domains. Subscribe to AWS Region availability, new features, security patches, bug fixes, and other improvements about OpenSearch Service.

Implementation steps

- Subscribe to AWS Newsletters and Blogs
- Setup CloudWatch alarms. For detailed implementation steps, see AOSOPS03-BP01.

Resources

- What's New with Analytics?
- Amazon OpenSearch Service Blog

AOSREL01-BP02 Regularly update your OpenSearch Service domain to the latest version

Keep your OpenSearch Service domain current by regularly updating it to the latest available version, which provides improved performance, issue resolution, and access to new features.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: Your OpenSearch Service domain is updated to the latest available version.

Benefits of establishing this best practice: Keeping your OpenSearch Service domain up to date with the latest version can improve performance, resolve known issues with bug fixes, and unlock new features and capabilities.

Implementation guidance

OpenSearch releases new versions regularly. Always keep your domains updated to apply improvements to your domains.

Implementation steps

- Open the Amazon OpenSearch Service console.
- Select the domain name to open its configuration.
- Choose **Actions**, then **Update**, and select one of the following options:
  - **Apply update now**: Immediately schedules the action to happen in the current hour if there's capacity available. If capacity isn't available, we provide other available time slots to choose from.
  - **Schedule it in off-peak window**: Only available if the off-peak window is enabled for the domain. Schedules the update to take place during the domain's configured off-peak window. There's no guarantee that the update will happen during the next immediate window. Depending on capacity, it might happen in subsequent days. For more information, see Scheduling software updates during off-peak windows.
  - **Schedule for specific date and time**: Schedules the update to take place at a specific date and time. If the time that you specify is unavailable for capacity reasons, you can select a different time slot.
- If you schedule the update for a later date (within or outside the domain's off-peak window), you can reschedule it at any time. For instructions, see Rescheduling actions.

- Choose **Confirm**.

Resources

- Service software updates in Amazon OpenSearch Service

Workload architecture

AOSREL02: How do you provide high availability for your OpenSearch Service domain?

To protect your OpenSearch Service domain during a service disruption, you can deploy nodes across two or three Availability Zones within the same Region. This configuration, known as Multi-AZ, leverages isolated locations called Availability Zones within each AWS Region. You can also choose to deploy OpenSearch with Multi-AZ with Standby deployment to protect against infrastructure failures. Enabling Multi-AZ with Standby deployment increases the availability of your domain to 99.99%.

AOSREL03: How do you monitor quotas and limits in your OpenSearch Service domain?

Cloud providers have built-in usage constraints that prevent excessive resource utilization. Understanding and monitoring these quotas is essential to prevent scaling issues and ensure uninterrupted service operations.

AOSREL04: How do you recover your OpenSearch Service domains from outages?

Disaster recovery involves plans and processes to resume operations and restore data in the aftermath of a disruptive event or catastrophe. Having a disaster recovery plan is important to recover from any disruptions to the OpenSearch domain.

AOSREL05: How do you use reliable instances for your production search workloads?

Business-critical workloads demand enough resources in production environments to handle reliability, stability, and high performance. In contrast, dev environments can function on minimal resources without impacting critical operations.

Best practices

- AOSREL02-BP01 Create Multi-AZ with Standby OpenSearch Service domains
- AOSREL03-BP01 Regularly review your OpenSearch Service quotas
- AOSREL04-BP01 Implement a disaster recovery (DR) strategy for your OpenSearch Service for business continuity
- AOSREL04-BP02 Implement an Index State Management (ISM) policy to generate snapshots for your crucial indices
- AOSREL04-BP03 Enable index replication for your critical indices
- AOSREL04-BP04 Employ cross-cluster replication to achieve higher availability
- AOSREL05-BP01 Implement appropriate compute sizing for production workloads

AOSREL02-BP01 Create Multi-AZ with Standby OpenSearch Service domains

Provide high availability and reliability in Amazon OpenSearch Service by creating Multi-AZ with Standby domains, minimizing downtime and simplifying domain management.

Level of risk exposed if this best practice is not established: High

Desired outcome: A highly available OpenSearch Service domain is created using Multi-AZ with Standby.

Benefits of establishing this best practice:

- **Improved availability:** Creating Multi-AZ with Standby OpenSearch Service domains can provide improved high availability, minimizing downtime and improving overall system reliability.
- **Simplified domain management:** This deployment option simplifies domain management by using multiple Availability Zones and best practices, making it easier to manage and maintain your OpenSearch Service domain.

Implementation guidance

Multi-AZ with Standby is a deployment option for Amazon OpenSearch Service that offers 99.99% availability, consistent performance for production workloads, and simplified domain management by using multiple Availability Zones and best practices.

Implementation steps

- Log in to AWS Management Console.
- Navigate to the Amazon OpenSearch Service console.
- Create a new domain or modify an existing domain:
  - For a new domain, choose **Create domain**.
    - You can use the Easy create method available under Domain creation method, which creates a new domain with Multi-AZ with Standby enabled by default.
    - If you want to have more control over different creation options, then you can choose **Standard create** and select **Domain with standby** in the Deployment option(s) box.
  - For an existing domain, select the domain name and choose **Actions**, then select **Edit cluster configuration**.
  - For an existing domain, choose **Domain with standby** under Deployment option(s) box.
- You will need to have a minimum of three data nodes to use Multi-AZ with Standby.
- Proceed with other options.

Resources

- AWS announces Multi-AZ with Standby for Amazon OpenSearch Service
- Amazon OpenSearch Service Under the Hood: Multi-AZ with Standby
- Configuring a Multi-AZ domain in Amazon OpenSearch Service

AOSREL03-BP01 Regularly review your OpenSearch Service quotas

Prevent workload limitations in Amazon OpenSearch Service by regularly reviewing and updating your domain quotas for smooth operation.

Level of risk exposed if this best practice is not established: Low

Desired outcome: Your OpenSearch Service domain quotas are regularly reviewed to prevent limitations on your OpenSearch workloads.

Benefits of establishing this best practice: Regularly reviewing OpenSearch Service quotas helps avoid limitations on your workloads, and you can continue to operate without restrictions.

Implementation guidance

Familiarize yourself with the domain and instance quotas for Amazon OpenSearch Service in your specific Region.

Implementation steps

- Understand Amazon OpenSearch Service's quota and limit policies by visiting Amazon OpenSearch Service quotas and Amazon OpenSearch Service endpoints and quotas, which provide a comprehensive list of all applicable quotas and restrictions.

- Monitor your quotas regularly:
  - Set a schedule to review your OpenSearch Service quotas regularly (like monthly or quarterly).
  - Make this a part of your operational procedures to maintain visibility into your resource limits and usage.
- To increase a soft limit, you can open a support ticket and use Service Quotas.
- Log in to AWS Management Console.
- To use Service Quotas:
  - Choose **AWS Services** in the Service Quotas left navigation.
  - Locate OpenSearch Service Quotas.
  - Search for Amazon OpenSearch Service.
  - Select the desired quota you want to increase, and choose **Request increase at account level**.

AOSREL04-BP01 Implement a disaster recovery (DR) strategy for your OpenSearch Service for business continuity

Protect your Amazon OpenSearch Service with a disaster recovery plan, which improves business continuity in case of disruptions by minimizing downtime, data loss, and risk.

Level of risk exposed if this best practice is not established: High

Desired outcome: You have a disaster recovery plan for your OpenSearch Service to maintain business continuity in case of disruptions.

Benefits of establishing this best practice: Implementing a disaster recovery strategy for your Amazon OpenSearch Service minimizes downtime and data loss in the event of an outage or disaster. This proactive approach protects sensitive data, reduces risk, and improves business continuity, enabling high availability and rapid service restoration.

Implementation guidance

To maintain business operations during emergencies, create an effective disaster recovery (DR). This plan should outline steps to recover the system within defined Recovery Time Objective (RTO) and Recovery Point Objective (RPO) timeframes.

- Collaborate with business stakeholders to determine the internal and external service-level agreements (SLAs) for your workloads, taking into account their criticality. This will help inform the development of disaster recovery (DR) plans that match the severity of potential outages.
- Design a tailored DR solution for each layer of your workloads, considering the specific requirements of each. For example, if you're ingesting logs from multiple systems directly into OpenSearch, storing them in Amazon S3 before processing can keep your data consistent during failures. Similarly, implementing a queue system like Amazon Managed Streaming for Apache Kafka or Amazon SQS between data sources and OpenSearch can facilitate the replaying of changes that occurred during downtime.
- Regularly implement and test your backup and restoration tools to guarantee they're functioning correctly, ensuring business continuity in the event of an outage.

Resources

- Disaster Recovery (DR) Architecture on AWS, Part I: Strategies for Recovery in the Cloud

AOSREL04-BP02 Implement an Index State Management (ISM) policy to generate snapshots for your crucial indices

Protect your critical Amazon OpenSearch Service indices by implementing an ISM policy that generates snapshots for disaster recovery, which helps you comply with regulatory requirements and simplify the recovery process.

Level of risk exposed if this best practice is not established: High

Desired outcome: Your crucial indices are protected by an ISM policy that generates snapshots for disaster recovery.

Benefits of establishing this best practice: By establishing a manual snapshot repository and using ISM and SM, you maintain full control over your Amazon OpenSearch Service data, and you can comply with regulatory requirements and data retention policies. Additionally, it can simplify disaster recovery processes by having a centralized repository for snapshots.

Implementation guidance

ISM automates index lifecycle tasks, including alias rollovers, snapshots, storage tier transitions, and deletion of old indices.

It is recommended that you review AOSPERF01 and AOSPERF02 to familiarize yourself with sharding strategies before you implement ISM policies.

Additionally, consult Index State Management in Amazon OpenSearch Service, Tutorial: Automating Index State Management processes and Index State Management pages for samples and full details about implementing ISM policies.

Implementation steps

- Open OpenSearch Dashboards for your domain.
- From the left sidebar, select **Index Management**, then **Create policy**.
- Use the visual editor or JSON editor to create policies. We recommend using the visual editor as it offers a more structured way of defining policies.
- After you create a policy, attach it to one or more indexes:

```
POST _plugins/_ism/add/my-index
        {
        "policy_id": "my-policy-id"
        }
```

Resources

- Creating index snapshots in Amazon OpenSearch Service

- Registering a manual snapshot repository
- Take manual snapshots and restore in a different domain spanning across various Regions and accounts in Amazon OpenSearch Service

AOSREL04-BP03 Enable index replication for your critical indices

Provide high availability and improved performance in Amazon OpenSearch Service by enabling index replication for critical indices, allowing replicas to handle read operations alongside primaries. This setup maintains access to data and minimizes downtime in case of primary shard unavailability.

Level of risk exposed if this best practice is not established: High

Desired outcome: Your critical indices are enabled with replication for improved availability.

Benefits of establishing this best practice:

- **Improved performance:** Index replication for critical indices distributes read requests across the OpenSearch Service domain, improving performance and throughput by allowing replicas to handle read operations alongside primaries.
- **Enhanced data availability:** If a primary shard becomes unavailable, replicas can automatically take over, providing continued access to your data and minimizing downtime.

Implementation guidance

For high-priority use cases, you can replicate critical indices in real-time using index replication, which maintains an up to date copy of your data. By doing this, you can distribute read requests across your OpenSearch Service domain, improving performance and throughput by allowing replicas to handle read operations alongside primaries. Additionally, if a primary shard becomes unavailable, replicas can automatically take over, providing continued access to your data.

Implementation steps

- To update the number of replicas for an existing index run the following:

```
PUT /my-index/_settings
          {
```

```
          "index" : {
          "number_of_replicas" : 2
          }
          }
```

- You can also use the auto_expand_replicas feature to dynamically adjust the number of replicas in your OpenSearch Service domain based on changes to the cluster, such as adding or removing data nodes. As a result, your replica count will match the number of data nodes you have at any specific moment.

> **Note**
>
> Adding multiple replicas may not necessarily enhance performance; in some cases, it can cause performance degradation. Therefore, use this feature with caution, and thoroughly test its effects before implementing it in your OpenSearch Service domain.

```
PUT /my-index/_settings
          {
          "index" : {
          "auto_expand_replicas" : "0-all"
          }
          }
```

Resources

- Update index settings
- Create Index

AOSREL04-BP04 Employ cross-cluster replication to achieve higher availability

Enhance OpenSearch Service domain availability by employing cross-cluster replication, which replicates user indexes, mappings, and metadata between domains in different AWS accounts or Regions, improving disaster recovery capabilities.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: Your Amazon OpenSearch Service is enabled with cross-cluster replication to enhance domain availability.

Benefits of establishing this best practice:

- **Enhanced domain availability**: Employing cross-cluster replication for Amazon OpenSearch Service can enhance domain availability by replicating user indexes, mappings, and metadata from one OpenSearch Service domain to another in another AWS account or Region.
- **Improved disaster recovery**: Cross-cluster replication helps with disaster recovery planning by allowing you to replicate data across geographically distant AWS Regions (data centers) and reduce latency in the event of an outage.

Implementation guidance

With cross-cluster replication in Amazon OpenSearch Service, you can replicate user indexes, mappings, and metadata from one OpenSearch Service domain to another domain in another AWS account or AWS Region. If there is an outage, cross-cluster replication assists in disaster recovery and replicates data across geographically distant AWS Regions to reduce latency.

Enabling cross-cluster replication in Amazon OpenSearch Service involves a multi-step process that includes setting up IAM roles and configuring both OpenSearch clusters. For a detailed guide on how to achieve this, see Ensure availability of your data using cross-cluster replication with Amazon OpenSearch Service.

Resources

- Cross-cluster replication for Amazon OpenSearch Service
- Ensure availability of your data using cross-cluster replication with Amazon OpenSearch Service

AOSREL05-BP01 Implement appropriate compute sizing for production workloads

Improve OpenSearch Service domain performance by implementing compute sizing that meets production workload requirements. This practice helps you avoid CPU throttling due to depleted burst credits and minimize risks.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: Your OpenSearch Service domain is running on instance families that meet the required performance and resource needs.

Benefits of establishing this best practice:

- Avoid CPU throttling if burst credits are depleted
- Improve your ability to maintain performance and minimize risks

Implementation guidance

Avoid using t2 or t3.small instances for production domains, as they can become unstable under sustained heavy load. t3.medium instances are an option for small production workloads (both as data nodes and as dedicated leader nodes).

Resources

- Operational best practices for Amazon OpenSearch Service

Key AWS services

- Amazon OpenSearch Service

Resources

- Recommended CloudWatch alarms for Amazon OpenSearch Service
- Notifying users on alarm changes
- Configuring a Multi-AZ domain in Amazon OpenSearch Service
- Disaster Recovery (DR) Architecture on AWS, Part I: Strategies for Recovery in the Cloud
- Creating index snapshots in Amazon OpenSearch Service
- Update index settings
- Cross-cluster replication for Amazon OpenSearch Service
- Amazon OpenSearch Service quotas
- Amazon OpenSearch Service endpoints and quotas
- Operational best practices for Amazon OpenSearch Service

- What's New with Analytics?
- Service software updates in Amazon OpenSearch Service
- Sending Amazon SNS alerts for available software updates

Performance efficiency

The performance efficiency pillar refers to the ability to deliver optimal performance while efficiently using system resources. Amazon OpenSearch Service is designed to handle large-scale distributed search and analytics workloads, and achieving performance efficiency is crucial for delivering timely and responsive results. The following questions and best practices are designed to complement the best practices in the Well-Architected Performance Efficiency Pillar whitepaper.

Focus areas

- Design principles
- Architecture selection
- Data management
- Process and culture
- Key AWS services
- Resources

Design principles

- **Optimize resource utilization:** Use resources efficiently by optimizing resource utilization through monitoring.
- **Sharding effectiveness:** Use an effective sharding strategy based on your workload type.
- **Optimize storage resources:** Optimize storage resources by distributing data evenly across data nodes.
- **Optimize query efficiency:** Optimize query efficiency by monitoring search and indexing logs.
- **Decide on optimal bulk request size:** Optimize bulk request size to improve data ingestion.

Architecture selection

AOSPERF01: How do you plan, monitor, and optimize your sharding strategy?

Providing the effectiveness of your sharding strategy in OpenSearch Service involves planning, monitoring, and ongoing optimization. Sharding is a critical aspect of OpenSearch Service's distributed architecture, and a well-designed strategy can significantly impact the performance, scalability, and reliability of your domain.

AOSPERF02: How do you monitor and optimize resource allocation for your OpenSearch instances?

Optimizing resource utilization in your OpenSearch domain instances is important for several reasons, as it contributes to the efficiency, cost-effectiveness, stability, reliability, efficient data storage and overall performance of your search and analytics environment.

Best practices

- AOSPERF01-BP01 Maintain shard sizes at recommended ranges
- AOSPERF01-BP02 Check shard-to-CPU ratio
- AOSPERF01-BP03 Check the number of shards per GiB of heap memory
- AOSPERF02-BP01 Implement processor utilization monitoring
- AOSPERF02-BP02 Implement Java memory utilization monitoring

AOSPERF01-BP01 Maintain shard sizes at recommended ranges

Shard size is recommended in a range of 10 GiB to 50 GiB.

Level of risk exposed if this best practice is not established: High

Desired outcome: Shards fall within the recommended size range of 10 GiB to 50 GiB, for efficient indexing, querying and relocation performance.

Benefits of establishing this best practice:

- Reduced CPU and memory utilization
- Enhanced domain stability by avoiding shard contention and node overload

Implementation guidance

Shard sizes depend on the workload, and the distribution of shards to data nodes significantly influences a domain's performance.

- Verify that the ideal shard size range is between 10GB and 30GB for *search-intensive* workloads, and 30GB to 50GB for *log analytics* and *time-series* data processing. Use `GET _cat/shards` to view shard size.
- If your indices have excessively small shards (less than 10GB in size), consider reindexing the data with a reduced number shards for that index to potentially boost performance and reduce CPU utilization. You can reindex the data from source into a new index or use the `_reindex` API to copy data from an existing index to a new one within the same domain.
- Verify shard to Java heap memory ratio. Verify that you have no more than 25 shards per GiB of Java heap.
- Verify that you don't have more than 1,000 shards per data node.
- You can implement ISM policies to roll over your indices when the shards reach a certain size.
- Consider OpenSearch's aliasing capability, which you can use to quickly update your indices without requiring modifications to your application.

Resources

- Sharding strategy
- Index aliases

AOSPERF01-BP02 Check shard-to-CPU ratio

Shards are allocated a minimum of 1.5 vCPUs per shard, supporting efficient processing and scalability.

Level of risk exposed if this best practice is not established: High

Desired outcome: The Shard-to-CPU ratio is maintained at a minimum level of 1.5 vCPUs per shard. This may vary depending on the use case. Search use cases might require a higher ratio, while log analytics use cases may work with a smaller ratio.

Benefits of establishing this best practice:

- A minimum shard-to-CPU ratio of 1.5 helps support each shard with sufficient processing power, reducing the risk of bottlenecks and promoting performance during indexing or search operations.
- By provisioning at least 1.5 vCPUs per shard, you can scale your cluster more efficiently, either by scaling up (increasing instance size) or out (adding more data nodes), to meet changing workload demands without sacrificing performance.

Implementation guidance

During indexing or search operations, each shard should use at least one vCPU to process requests. To guarantee sufficient processing capacity, maintain a minimum shard-to-CPU ratio of 1:1.5. This means that if you have 10 shards in your cluster, for example, you should provision at least 15 vCPUs (10 x 1.5 = 15). You can scale up or out to meet the needs of your cluster based on the total number of shards their density in your data nodes.

In addition to the recommended shard-to-CPU ratio, for more detail, see AOSPERF01-BP01. This helps to align shard sizes with your workload requirements.

Resources

- Sharding strategy

AOSPERF01-BP03 Check the number of shards per GiB of heap memory

Prevent inefficient resource utilization by keeping each data node's heap under a shard load of 25.

Level of risk exposed if this best practice is not established: High

Desired outcome: Each data node's heap memory should support no more than 25 shards, thereby optimizing resource utilization and enhancing query performance.

Benefits of establishing this best practice:

- Improved query performance
- Reduce risks of running out of memory while executing the queries

Implementation guidance

Amazon OpenSearch Service allocates approximately half of each data node's physical memory, up to 32 GB, for the Java Virtual Machine (JVM) and OpenSearch. In a system with 16 GB of memory, this amounts to roughly 8 GB reserved for the JVM.

Additionally, the total number of shards a node can handle is directly related to its JVM heap memory size. To maintain an optimal balance, strive for a shard-to-JVM-heap-memory ratio of approximately 25:1. For example, with 16 GB of memory (and thus 8 GB allocated to the JVM), you should aim for no more than 200 shards per node (25 x 8 = 200).

For further guidance on optimal shard sizing, see AOSPERF04-BP01 and AOSPERF04-BP02.

Resources

- Sharding strategy
- Choosing the number of shards
- Sizing OpenSearch Service domains

AOSPERF02-BP01 Implement processor utilization monitoring

Keep CPU usage under 75% to maintain efficient resource utilization and prevent potential performance issues.

Level of risk exposed if this best practice is not established: High

Desired outcome: CPU utilization is below 75% for efficient resource utilization and minimizing potential performance issues.

Benefits of establishing this best practice:

- **Prevent performance issues:** Keeping CPU utilization below 75% helps identify and address potential issues related to high CPU usage, minimizing the risk of indexing or query processing bottlenecks.
- **Maintain scalability:** By keeping CPU utilization within a safe threshold, you can prevent scalability issues and ensure that your OpenSearch Service domains can handle increased traffic or workloads without degradation.

Implementation guidance

The performance and efficiency in your OpenSearch Service domain can be maintained by closely monitoring CPU utilization. This involves tracking CPU usage over the past 14 days and verifying that the average rate remains below 75 percent. By doing so, you can proactively identify and address potential issues related to high CPU utilization, such as indexing or query processing bottlenecks, before they affect your domain's performance or scalability.

Implementation steps

- Log in to the AWS Management Console.
- Navigate to the Amazon OpenSearch Service console.
- Select your OpenSearch Service domain name.
- Choose the **Cluster health** tab.
- Navigate to the Data nodes box.
- Choose the **CPU utilization** graph.
- Adjust time range to `2w` and Statistic to `Average`.

If you notice that your average CPU utilization exceeds 75% over a 14-day period, it's recommended to investigate further by checking for dedicated leader nodes and reviewing the best practices outlined in AOSPERF01.

Resources

- Monitoring OpenSearch cluster metrics with Amazon CloudWatch
- Choosing the number of shards
- Sizing OpenSearch Service domains

AOSPERF02-BP02 Implement Java memory utilization monitoring

Keep `JVMMemoryPressure` below 85% to maintain efficient memory utilization and prevent potential performance issues.

Level of risk exposed if this best practice is not established: High

Desired outcome: The JVMMemoryPressure metric remains below 85% for efficient memory utilization and minimizing potential performance issues or domain blocks.

Benefits of establishing this best practice: Closely monitoring `JVMMemoryPressure` helps you identify and address any potential issues related to high memory pressure, such as excessive garbage collection or memory leaks, before they impact your domain's performance, availability and scalability.

Implementation guidance

To maintain optimal performance and memory utilization for your OpenSearch Service domains, closely monitor the `JVMMemoryPressure` metric. Specifically, track this metric to verify that it remains below the recommended threshold of 85 percent. This helps you identify and address any potential issues related to high memory pressure, such as excessive garbage collection or out of memory problems, before they impact your domain's performance or scalability.

Implementation steps

- Log in to the AWS Management Console.
- Navigate to the Amazon OpenSearch Service console.
- Select your OpenSearch Service domain name.
- Choose the **Cluster health tab**.
- Navigate to the Data nodes box.
- Choose the `JVMMemoryPressure` graph.
- Adjust time range to `2w` and Statistic to `Maximum`.

If you notice that your `JVMMemoryPressure` exceeds 85% over a 14-day period, investigate further by reviewing the best practices outlined in AOSPERF01 as well as reviewing some of the common issues outlined in How do I troubleshoot high JVM memory pressure on my OpenSearch Service cluster.

Resources

- How do I troubleshoot high JVM memory pressure on my OpenSearch Service cluster
- Troubleshooting Amazon OpenSearch Service
- Choosing the number of shards
- Sizing OpenSearch Service domains

Data management

AOSPERF03: How do you monitor and optimize storage resource utilization in your OpenSearch Service domain?

Optimizing the efficient use of storage resources in OpenSearch is crucial for cost-effectiveness, performance, and overall cluster management.

AOSPERF04: How do you monitor and optimize query performance?

Efficient querying in an OpenSearch domain requires considerations related to query optimization, index design, and overall system configuration. However, you can achieve performance by carefully designing your indices and map your data to optimize query performance. Consider using appropriate and correct data types, reduce unnecessary fields, and choose the right analyzer for your text fields. A well-designed index structure is the foundation for efficient queries.

AOSPERF05: How do you separate cluster management from data processing functions?

In an OpenSearch Service deployment, two specialized node types play important roles in maintaining efficient service operation: *dedicated leader nodes* and *dedicated coordinator nodes*.

- The dedicated leader node is responsible for domain coordination and management tasks such as tracking all nodes within the cluster, monitoring index numbers, keeping track of shard distribution, and updating the cluster state following any changes.
- The function of a coordinator node role is to forward requests to the data nodes that hold the relevant data, aggregating each node's results into a single global result set.

Best practices

- AOSPERF03-BP01 Establish storage utilization thresholds
- AOSPERF03-BP02 Evenly distribute data across data nodes in your OpenSearch Service domain

- AOSPERF03-BP03 Use storage types that provide higher IOPs and throughput baseline
- AOSPERF04-BP01 Enable slow log functionality for search and indexing
- AOSPERF04-BP02 Use static mapping for your index
- AOSPERF04-BP03 Use the flat object type for nested objects
- AOSPERF05-BP01 Enable a dedicated leader node for OpenSearch Service domain
- AOSPERF05-BP02 Enable a dedicated coordinator node for OpenSearch Service domain

AOSPERF03-BP01 Establish storage utilization thresholds

Maintain efficient resource utilization by keeping domain storage usage under 75% to prevent potential performance issues or cluster write blocks.

Level of risk exposed if this best practice is not established: High

Desired outcome: Domain storage usage remains below 75%, which improves resource utilization and minimizes potential performance issues due to storage constraints.

Benefits of establishing this best practice: You can prevent write operations such as adding documents or creating indices to fail.

Implementation guidance

If one or more data nodes in your cluster have storage space less than the minimum value of either 20% of available storage space or 20 GB, basic write operations such as adding documents and creating indexes may fail.

- The Amazon CloudWatch `FreeStorageSpace` metric is used to monitor available storage in the data nodes.
- Adjusting your indexing rate or deleting unnecessary data can help keep your storage usage below the set threshold.
- If you are using UltraWarm nodes, consider migrating logs and time-series indices from the hot storage to the UltraWarm nodes.

Implementation steps

- Log in to the AWS Management Console.

- Navigate to the Amazon OpenSearch Service console.
- Select your OpenSearch Service domain name.
- Choose the **Cluster health** tab.
- Navigate to the Data nodes box.
- Choose the `FreeStorageSpace` graph.
- Adjust time range to `2w` and Statistic to `Maximum`.
- Remove unnecessary or redundant data from your domain. Use the `GET _cat/indices` API to list and investigate your indices and the `DELETE /<index_name>` API to remove unnecessary or redundant data.
- Consider adding more storage to your domain by increasing EBS storage size or add more data nodes to your domain. Keep this step consistent with shard-to-CPU ratio, shard-to-java-heap ratio, and shards distribution across your domain. For more information, see AOSPERF01-BP01, AOSPERF01-BP02, and AOSPERF03-BP02.
- For more detail, see Lack of available storage space.

Resources

- Monitoring OpenSearch cluster metrics with Amazon CloudWatch
- Choosing the number of shards
- Sizing OpenSearch Service domains
- Lack of available storage

AOSPERF03-BP02 Evenly distribute data across data nodes in your OpenSearch Service domain

Maintain efficient resource utilization and optimal query performance by distributing data evenly across data nodes.

Level of risk exposed if this best practice is not established: High

Desired outcome: Data is evenly distributed across data nodes in the OpenSearch Service domain, ensuring efficient resource utilization and optimal query performance.

Benefits of establishing this best practice:

- **Prevent bottlenecks and hotspots:** Evenly distributing data across data nodes in your OpenSearch Service domain helps prevent bottlenecks and hotspots, reducing the risk of node overload.
- **Improves availability:** When shards are distributed evenly across data nodes, can also help maintain high availability within your domain, as no single node will be overwhelmed with shards or storage, minimizing the risk of full node failure.

Implementation guidance

Node shard skew is when one or more nodes within a domain has significantly more shards than the other nodes. *Node storage skew* is when one or more nodes within a cluster has significantly more storage (`disk.indices`) than the other nodes. While both conditions can occur temporarily, like when a domain has replaced a node and is still allocating shards to it, you should address them if they persist because uneven distribution can lead to bottlenecks and hotspots, causing queries to slow down or even fail. In some cases, you can have a full node failure when it has a very high density of shards compared to other nodes in the domain.

You need to identify node shard and index shard skewness, and use shard counts that are multiples of the data node count to distribute each index evenly across data nodes.

Implementation steps

- Run the `GET _cat/allocation` API operation in OpenSearch Service to retrieve shard allocation information.
- Compare the `shards` and `disk.indices` entries in the response to identify skew.
- Note the average storage usage across all shards and nodes.
- Determine if storage skew is normal (within 10% of the average) or significant (over 10% above the average).
- Use shard counts that are multiples of the data node count to evenly distribute each index across data nodes. If you still see index storage or shard skew, you might need to force a shard reallocation, which occurs with every blue/green deployment of your OpenSearch Service domain.

Resources

- Node shard and storage skew

- Blue/Green Deployment

AOSPERF03-BP03 Use storage types that provide higher IOPs and throughput baseline

Improve storage performance with higher baseline IOPs and throughput using more scalable and efficient volume types.

Level of risk exposed if this best practice is not established: Low

Desired outcome: Improved storage performance, with increased input and output operations Per second (IOPs) and higher throughput.

Benefits of establishing this best practice:

- **Improved baseline performance:** Using gp3 EBS volumes provide higher baseline performance compared to gp2 volumes.
- **Scalable high performance:** With gp3 volumes, you can provision higher performance independently of the volume size, allowing for more scalable and efficient performance in your OpenSearch Service domains.

Implementation guidance

gp3 is the successor to the general-purpose SSD gp2 volume, offering higher baseline performance and the capability to provision higher performance independent of volume size.

Implementation steps

- Log in to the AWS Management Console.
- Navigate to the Amazon OpenSearch Service console.
- For an existing domain, select the domain name and ch**oose Actions**, then **Edit cluster configuration**.
- Navigate to Number of data nodes box, and locate EBS volume type.
- Change the volume type from General Purpose (SSD) – gp2 to General Purpose (SSD) – gp3.

Resources

- Making configuration changes in Amazon OpenSearch Service

AOSPERF04-BP01 Enable slow log functionality for search and indexing

Gain visibility into query latency by enabling slow logs for search and indexing. This visibility helps you optimize queries and troubleshoot issues.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You have enabled slow logs for search and indexing operations, which provides you a detailed view of query latency and enabling optimization efforts.

Benefits of establishing this best practice:

- **Optimize queries:** Slow logs provide detailed information about slow or long-running queries, which you can use to identify areas for optimization and make changes to improve performance.
- **Troubleshoot issues:** By capturing logs of slow searches, indexing operations, and other queries, slow logs help you troubleshoot issues more effectively. This process reduces downtime and improves overall efficiency in your OpenSearch Service domains.

Implementation guidance

To optimize the logging and troubleshooting process for your OpenSearch Service domains, enable slow log functionality for both search and indexing operations. Capture detailed logs of slow or long-running queries and indexing operations, which provide insights into performance bottlenecks and optimization areas. By enabling slow log functionality, you can better understand how your OpenSearch Service domains are performing under various loads and conditions and make data-driven decisions to improve overall efficiency and scalability.

Understand the benefits of slow logs. Slow logs help you identify performance bottlenecks in your OpenSearch instance by capturing detailed information about slow searches, indexing operations, and other queries. This information can be used to optimize your queries, improve performance, and troubleshoot issues.

Implementation steps

- Enable log publishing to CloudWatch (as described in Monitoring OpenSearch logs with Amazon CloudWatch Logs).
- Set up the search and index (shard) slow log thresholds. Search slow logs are configured with cluster settings, while shard slow logs are configured at index level.
- Search slow log:

```
PUT domain-endpoint/_cluster/settings
          {
          "transient": {
          "cluster.search.request.slowlog.threshold.warn":
          "5s",
          "cluster.search.request.slowlog.threshold.info":
          "2s"
          }
          }
```

- Shard level slow logs:

```
PUT domain-endpoint/index/_settings
          {
          "index.search.slowlog.threshold.query.warn":
          "5s",
          "index.search.slowlog.threshold.query.info":
          "2s"
          }
```

Resources

- Monitoring OpenSearch logs with Amazon CloudWatch Logs
- Shards slow logs

AOSPERF04-BP02 Use static mapping for your index

Improve indexing performance and query efficiency by using fixed, static mappings instead of dynamic ones.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You use static mappings instead of dynamic mappings, employing efficient and optimized indexing and querying strategies.

Benefits of establishing this best practice:

- **Improve indexing performance:** Using static mappings for your index in OpenSearch Service domains can improve indexing performance by defining a fixed mapping at creation time, which is more efficient than dynamic mappings.
- **Enhance query efficiency:** By using static mappings, you can also enhance query efficiency and reduce the risk of slower query performance associated with dynamic mappings.

Implementation guidance

To improve indexing and query performance in OpenSearch Service domains, consider switching from dynamic to static mappings. Understand the difference between static and dynamic mappings to choose the best approach for your data.

- **Static mappings:** Define a fixed mapping for an index at creation time. This approach is more efficient and can provide better performance.
- **Dynamic mappings:** Allow OpenSearch to detect and create mappings automatically based on the data being indexed. While convenient, this approach can lead to slower query performance and increased storage and resources, such as CPU and memory, usage.

Implementation steps

- Mappings are defined at index level. The following is an example of a static mapping:

```
PUT sample-index1
        {
        "mappings": {
        "properties": {
        "year": { "type" : "text" },
        "age": { "type" : "integer" },
        "director":{ "type" : "text" }
        }
        }
```

```
        }
```

- Mappings can be either created using `PUT HTTP verb` or added to an existing mapping using `POST HTTP`. But you cannot change the mapping of an existing field. If you need to change existing field types, consider using the `_reindex` API to copy the data to a new index.

Resources

- Mappings and field types

AOSPERF04-BP03 Use the flat object type for nested objects

Improve efficiency by using flat object types for indexing and querying complex data structures, reducing storage and retrieval overhead.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You use flat object types for indexing and querying complex data structures, which improves the efficiency of your storage and retrieval of nested objects.

Benefits of establishing this best practice:

- **Efficient reads:** Fetching performance is similar to that of a keyword field.
- **Memory efficiency:** Storing the entire complex JSON object in one field without indexing all of its subfields reduces the number of fields in an index.
- **Space efficiency:** OpenSearch does not create an inverted index for subfields in flat objects, which saves space.
- **Compatibility for migration:** You can migrate your data from systems that support similar flat types to OpenSearch.

Implementation guidance

The flat object field type treats the JSON object as a string, allowing subfield access without indexing for faster lookup and addressing performance concerns.

Flat objects are defined in the mapping of an index, either at creation time or by adding a new field that is flat object type. You cannot change an existing field to use flat object.

Implementation steps

- Create a mapping for your index where issue is of type `flat_object`:

```
PUT /test-index/
        {
        "mappings": {
        "properties": {
        "issue": {
        "type": "flat_object"
        }
        }
        }
        }
```

Resources

- Flat object field type

AOSPERF05-BP01 Enable a dedicated leader node for OpenSearch Service domain

Configure dedicated leader nodes with appropriate instance types to improve cluster stability and performance. This separation of cluster management tasks helps prevent split-brain scenarios and optimizes resource utilization as your OpenSearch Service domain grows.

Level of risk exposed if this best practice is not established: High

Desired outcome: Your OpenSearch Service domain has dedicated leader nodes with appropriately sized instances to handle cluster management tasks. This configuration improves cluster stability, prevents split-brain scenarios, and maintains performance as your workload grows.

Benefits of establishing this best practice:

- Improved cluster stability through isolated cluster management tasks
- Enhanced reliability by preventing split-brain scenarios and reducing election times
- Better resource utilization with workload separation from data nodes
- Increased scalability and performance as your domain grows

- Reduced risk of operational disruptions

Implementation guidance

Dedicated manager nodes' size is directly tied to instance size, number of instances, indexes, and shards they manage. For production clusters, we recommend, at a minimum, the following instance types for dedicated managing nodes.

Instance count	Manager node RAM Size	Maximum supported shard count	Recommended minimum dedicated manager instance type
1–10	8 GiB	10K	m5.large.search or m6g.large.search
11–30	16 GiB	30K	c5.2xlarge.search or c6g.2xlarge.search
31–75	32 GiB	40K	r5.xlarge.search or r6g.xlarge.search
76–125	64 GiB	75K	r5.2xlarge.search or r6g.2xlarge.search
126–200	128 GiB	75K	r5.4xlarge.search or r6g.4xlarge.search

Implementation steps

- Log in to the AWS Management Console.
- Navigate to the Amazon OpenSearch Service console.
- Create a new domain or modify an existing domain:
  - For a new domain, choose **Create domain**. For an existing domain, select the domain name and choose **Actions**, then **Edit cluster configuration**.
- Enable dedicated leader nodes:

  - In the Cluster configuration section, under Dedicated master nodes, toggle on the option to enable dedicated leader nodes.
- Choose leader node instance type and count:
  - **Instance type:** Select an instance type that fits the workload needs. For detailed information on choosing the right instance type for leader roles, see AOSPERF05-BP02.
  - **Instance count:** Select three or five. Three is usually more than enough for the majority of workloads.
- After configuring the settings, review your setup and choose **Create** (for a new domain) or **Save changes** (for an existing domain).

Resources

- For information about how certain configuration changes can affect dedicated leader nodes, see Making configuration changes in Amazon OpenSearch Service.
- Learn how to add dedicated manager nodes in your OpenSearch Service domain
- For clarification on instance count limits, see OpenSearch Service domain and instance quotas.
- For more information about specific instance types, including vCPU, memory, and pricing, see Amazon OpenSearch Service prices.

AOSPERF05-BP02 Enable a dedicated coordinator node for OpenSearch Service domain

Improve resource utilization and resilience by enabling a dedicated coordinator node, reducing private IP address reservations

Level of risk exposed if this best practice is not established: High

Desired outcome: You use a dedicated coordinator node for an OpenSearch Service domain to improve resource utilization, enhance resilience, and reduce private IP address reservations.

Benefits of establishing this best practice:

- **Improve resource utilization**: Enabling a dedicated coordinator node in your OpenSearch Service domain can improve resource utilization by relieving data nodes from coordination responsibilities, allowing them to focus on core tasks like search, data storage, and indexing.

- **Enhance domain resiliency**: With a dedicated coordinator node, you can also enhance the resiliency of your OpenSearch Service domain by reducing the risk of resource (memory and CPU) strain on data nodes that are hosting OpenSearch Dashboards.
- **Reduce private IP addresses reservation**: Provisioning a dedicated coordinator node can help reduce the reservation of private IP addresses required for virtual private cloud (VPC) domains, making it easier to manage your domain's resources.

Implementation guidance

The following are general guidelines that apply to many use cases. Each workload is unique, with unique characteristics, so no generic recommendation is exactly right for every use case.

- General-purpose instances are enough for most of the use cases.
- Keep the instance family the same for a dedicated coordinating node and data node.
- It is recommended to provision 5% to 10% of your domain's total data nodes as dedicated coordinator nodes. For example, if your domain has 90 R6g.large data nodes, you can plan for coordinator nodes to make up anywhere between 5 and 9 of the R6g.large instance type.
- As a starting point, the instance size of your dedicated coordinator nodes should be the same as your domain's data nodes. However, in certain cases, it may be advisable to use a smaller instance type with a higher count to ensure availability. For example, if you have R6g.8xlarge as the instance size for a data node, you can try 3 instances of R6g.2xlarge size instead of one R6g.8xlarge size to achieve better availability.
- Source domains typically do not perform coordination tasks. If you are using cross region search, it is recommended you provision a dedicated coordinator node on the destination domains.

Implementation steps

- Log in to the AWS Management Console.
- Navigate to the Amazon OpenSearch Service console.
- Create a new domain or modify an existing domain:
  - For a new domain, choose **Create domain**. For an existing domain, select the domain name and choose **Actions**, then **Edit cluster configuration**..
- Enable dedicated coordinator nodes:
  - In the Cluster configuration section, under Dedicated coordinator nodes, toggle on the option to enable dedicated coordinator nodes.

- Choose coordinator node instance type and count. Follow the recommendations mentioned in the implementation guidance for this best practice.
- After configuring the settings, review your setup and choose **Create** (for a new domain) or **Save changes** (for an existing domain).

Process and culture

AOSPERF06: How do you balance throughput and latency through bulk request size?

Determining the optimal bulk request size for data ingestion in OpenSearch involves balancing factors such as network latency, indexing rate, document size, batching frequency, refresh intervals, and buffer size.

Best practices

- AOSPERF06-BP01 Identify index refresh controls for optimal ingestion performance
- AOSPERF06-BP02 Evaluate bulk request size
- AOSPERF06-BP03 Implement HTTP compression
- AOSPERF06-BP04 Evaluate filter_path criteria

AOSPERF06-BP01 Identify index refresh controls for optimal ingestion performance

Improve indexing throughput and speed by adjusting the refresh_interval value to more than 30 seconds.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: The refresh_interval value is set to more than 30 seconds, which could potentially lead to increased indexing throughput and faster indexing speeds.

Benefits of establishing this best practice: By adjusting the `refresh_interval`, you can optimize index write performance, as less frequent refreshes allow for more efficient ongoing writes which usually results as faster indexing speeds.

Implementation guidance

A refresh operation makes all updates to an index accessible for search. The default refresh interval is one second, indicating that OpenSearch Service performs a refresh every second during ongoing index writes.

Implementation steps

- Check the current `refresh_interval` value for your index.

```
GET /<index-name>/_settings/index.refresh_interval
```

- Change the `refresh_interval` value to 30s or more

```
PUT /sample_data/_settings
        {
        "index" : {
        "refresh_interval" : "30s"
        }
        }
```

- It is also possible to disable the automatic refreshes by setting `refresh_interval": "-1"`
- If the `refresh_interval` is disabled, you can manually refresh an index running `POST <index-name>/_refresh`.
- If you're loading new data into your domain through a batch process, it might be beneficial to disable the automatic refresh just before the batch process begins, and re-enable it after the process concludes.

Resources

- Optimize OpenSearch Refresh Interval

AOSPERF06-BP02 Evaluate bulk request size

Improve indexing performance and domain stability by adjusting bulk request size to a recommended range of 3-5 MiB.

Level of risk exposed if this best practice is not established: Low

Desired outcome: The bulk request size is within the recommended starting range of 3 – 5 MiB, ensuring efficient data ingestion and query performance.

Benefits of establishing this best practice:

- Improved indexing performance
- Enhanced domain stability by avoiding unnecessary resource utilization

Implementation guidance

Determining the optimal bulk size depends on your data and domain configuration. However, a recommended starting point is a bulk size of three to five MiB per request.

To optimize bulk requests, define the target document batch size within your application. Instead of trying to determine an exact number of documents per batch, focus on reaching a total batch size that falls within this range.

If you have multiple systems that are sending batches to OpenSearch, consider either using a service like Amazon OpenSearch Service Ingestion, which can queue the batches for more effective ingestion, or ensure that the size of the batches across all systems do not overpass the recommended range if they simultaneously ingest data into your OpenSearch Service domain.

Resources

- Optimize bulk request size and compression

AOSPERF06-BP03 Implement HTTP compression

Reduce request and response payload sizes by enabling GZIP compression.

Level of risk exposed if this best practice is not established: Low

Desired outcome: GZIP compression is enabled in the OpenSearch Service domain, reducing the size of data being ingested or requested.

Benefits of establishing this best practice: Reduce the payload size of requests and responses.

Implementation guidance

In OpenSearch Service domains, GZIP compression can be used to compress HTTP requests and responses. This compression helps reduce document size, lowering bandwidth usage and latency, resulting in improved transfer speeds.

Implementation steps

- To use compression, include the following headers in the HTTP requests of your application:

```
'Accept-Encoding': 'gzip'
```

```
'Content-Encoding': 'gzip'
```

Resources

- Compressing HTTP requests in Amazon OpenSearch Service

AOSPERF06-BP04 Evaluate `filter_path` criteria

Reduce response and request sizes by optimizing the filter_path criteria, minimizing download traffic.

Level of risk exposed if this best practice is not established: Low

Desired outcome: Reduced response and request size.

Benefits of establishing this best practice: Reduced download traffic.

Implementation guidance

Responses from the `_index` and `_bulk` APIs carry extensive information, which is valuable for troubleshooting or implementing retry logic. However, due to bandwidth considerations, indexing a 32-byte document, for instance, results in a 339-byte response (including headers).

This response size might seem minimal, but if you index 1,000,000 documents per day (or approximately 11.5 documents per second), 339 bytes per response works out to 10.17 GB of download traffic per month.

Implementation steps

- Use `filter_path` with the APIs that you call frequently, such as the `_index` and `_bulk` APIs. For example:

```
PUT opensearch-domain/<index-name>
        /_doc/1?filter_path=result,_shards.total
        POST
        opensearch-domain/_bulk?filter_path=-took,-items.index._*
```

Resources

- Reducing response size

Key AWS services

- Amazon OpenSearch Service: A fully managed service that helps you search, ingest, and analyze data in real-time.
- Amazon CloudWatch: Monitoring and logging service that helps you monitor and troubleshoot your AWS resources.
- Amazon Elastic Block Store (EBS): A service that provides block-level storage volumes for use with Amazon EC2 instances.

Resources

- Monitoring OpenSearch cluster metrics with Amazon CloudWatch
- Troubleshooting Amazon OpenSearch Service
- Sharding strategy
- Demystifying OpenSearch shard allocation
- Sharding strategy
- Lack of available storage
- Node shard and storage skew
- Making configuration changes in Amazon OpenSearch Service
- Monitoring OpenSearch logs with Amazon CloudWatch Logs

- Mappings and field types
- Flat object field type
- Optimize OpenSearch Refresh Interval
- Optimize bulk request size and compression
- Compressing HTTP requests in Amazon OpenSearch Service
- Reducing response size

Cost optimization

The cost optimization pillar focuses on avoiding unnecessary costs. Key topics include understanding spending over time and controlling fund allocation, selecting resources of the right type and quantity, and scaling to meet business needs without overspending.

Focus areas

- Design principles
- Practice Cloud Financial Management (CFM)
- Expenditure and usage awareness
- Cost-effective resources
- Optimize over time
- Key AWS services
- Resources

Design principles

In addition to the design principles described in the AWS Well-Architected Framework Cost Optimization Pillar whitepaper, the OpenSearch Lens identifies the following design principles to facilitate good design in the cloud for your OpenSearch workloads.

These design principles can help you build and operate cost-aware OpenSearch Service domains that achieve business outcomes while minimizing costs and allowing your organization to maximize its return on investment.

- **Optimize instance type:** Optimize instance type by using the right instance type for your OpenSearch domain, using the latest generation of instances, and employing the appropriate instance type for your OpenSearch workload.
- **Optimize storage tier:** Configure storage tiers effectively by implementing hot, warm, and cold architectures in your OpenSearch domain. You can use different storage options including Amazon EBS volumes, instance store volumes, UltraWarm, OR1/OR2 instances and Zero-ETL Direct query S3 data.
- **Implement reserved instances:** Use Reserved Instances for your OpenSearch domain to receive discounted pricing.

- **Monitor usage and cost:** Actively monitor the usage and cost of your OpenSearch domain to estimate and reduce your service costs.

Practice Cloud Financial Management (CFM)

Cloud Financial Management (CFM) refers to finance, product, technology, and business organizations managing, optimizing, and planning costs as they grow their usage and scale on AWS. Use CFM to manage, optimize, and plan costs as you grow usage and scale on AWS. AWS CFM offers a set of capabilities to manage, optimize, and plan for cloud costs while maintaining business agility. CFM is important not only to manage costs effectively but also to verify that investments are driving expected business outcomes. The four pillars of the Cloud Financial Management Framework in the AWS Cloud are see, save, plan, and run. Each of these pillar areas has a set of activities and capabilities.

Following the best practices in CFM is essential for managing costs in your OpenSearch workloads.

These CFM best practices help you establish cost transparency to control your resources and plan your spend to optimize your return on investment (ROI).

AOSCOST01: How do you select optimal compute resources for your OpenSearch Service workloads?

OpenSearch Service offers flexible instance type and storage options to optimize your workload. You can choose from memory-optimized, compute-optimized, and other instance types tailored to specific workloads. By selecting the right instance type, you can efficiently run your workloads while being cost optimized.

Best practices

- AOSCOST01-BP01 Use the latest generation of instances for your OpenSearch Service domains
- AOSCOST01-BP02 Employ the appropriate instance type and count
- AOSCOST01-BP03 Evaluate manager nodes

AOSCOST01-BP01 Use the latest generation of instances for your OpenSearch Service domains

Improve performance and reduce costs by using the latest instance generation, which offers enhanced memory, security features, and potentially better price-performance ratios.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You use the latest generation of instances for OpenSearch Service domains to gain optimal performance, increased memory, and enhanced security features.

Benefits of establishing this best practice:

- Enhance the performance of your OpenSearch Service domains.
- Modern instances can lead to cost savings, as they often provide better price-performance ratios compared to older instance types.

Implementation guidance

Amazon OpenSearch Service offers the option to create domains using the latest generation Amazon OpenSearch Service instance types, delivering enhanced performance and reduced instance costs. Regularly review the supported instance types, and upgrade your domain to use the latest generation of the instances.

- **Identify supported instance types:** Visit the Amazon OpenSearch Service to explore the available instance types.

- **Search for suitable instance types:** Use Amazon OpenSearch Service pricing to find instances that align with your workload and financial requirements. You can filter results by Region to narrow down the options.

Resources

- Supported instance types in Amazon OpenSearch Service
- Amazon OpenSearch Service pricing

AOSCOST01-BP02 Employ the appropriate instance type and count

Improve query performance, enhance domain stability, and reduce risk by selecting the appropriate instance type and count for your workload.

Level of risk exposed if this best practice is not established: High

Desired outcome: The correct instance type and count is used for OpenSearch Service workloads for optimal resource utilization and improved performance.

Benefits of establishing this best practice:

- Improved query performance through optimized data retrieval
- Enhanced domain stability by avoiding unnecessary resource utilization and potential over-provisioning
- Reduced risk of data loss or corruption due to inadequate resource allocation

Implementation guidance

Hardware requirements vary depending on the workload, with some workloads requiring more memory and others more processing power. OpenSearch is a CPU-intensive application that requires enough vCPU cores for efficient handling, with indexing-heavy workloads typically requiring more cores and searching or aggregating workloads requiring fewer cores with higher clock speeds.

Your sharding strategy plays a significant role in determine the instance type and count for your OpenSearch Service domain. For more detail see AOSPERF01. The following summary explores the main factors that impact your choice of the number of instances and the instance type you need:

- **Review your shards:** A general guideline is to try to keep shard size between 10-30 GiB for workloads where search latency is a key performance objective, and 30-50 GiB for write-heavy workloads such as log analytics. You can use `_cat/shards` to review shard size. For more detail, see Choosing the number of shards
- **Avoid shard skewness:** We recommended that you have a balanced shard distribution across your OpenSearch Service domain. Verify that the total number of shards is a multiple of the number of data nodes or that the total is divisible by the number of data nodes.
- **Shard to CPU ratio:** As a starting point, use a 1.5 multiplicator to calculate the ratio between shard number and CPUs in the domain. For example, if you have 10 shards, you will need 15

vCPUs in your OpenSearch Service domain. Another way to look at this is at data node level. For instance, if your data node instance has eight vCPUs, you should not have more than six shards on that node.

- **Search for suitable instance types:** For instances that align with your workload and financial requirements, see Choosing instance types and testing.

Resources

- Sizing OpenSearch Service domains
- Choosing instance types and testing

AOSCOST01-BP03 Evaluate manager nodes

Improve domain stability, reduce risk of data loss, and prevent over-provisioning by using an appropriate instance type for dedicated leader nodes.

Level of risk exposed if this best practice is not established: High

Desired outcome: The correct instance type is used for dedicated leader nodes in OpenSearch Service to gain optimal resource utilization and performance.

Benefits of establishing this best practice:

- Enhanced domain stability, as the leader role is handled by a dedicated node. This practice avoids unnecessary resource utilization and potential over-provisioning of data nodes.
- Reduced risk of data loss or corruption due to inadequate resource allocation or leader failure.

Implementation guidance

Consider dedicated manager nodes size correlated with data node instance sizes, number of instances, indexes, and shards they can manage. For production clusters, we recommend, at a minimum, the following instance types for dedicated leading nodes:

Instance count	Manager node RAM size	Maximum supported shard count	Recommended minimum dedicated leader instance type
1–10	8 GiB	10K	m5.large.search or m6g.large.search
11–30	16 GiB	30K	c5.2xlarge.search or c6g.2xlarge.search
31–75	32 GiB	40K	r5.xlarge.search or r6g.xlarge.search
76–125	64 GiB	75K	r5.2xlarge.search or r6g.2xlarge.search
126–200	128 GiB	75K	r5.4xlarge.search or r6g.4xlarge.search

Resources

- Dedicated manager nodes in Amazon OpenSearch Service

Expenditure and usage awareness

To effectively manage costs and drive efficiency, it's essential to understand your organization's expenses, identify cost drivers, and attribute resource costs to specific workloads, teams, or product owners. This understanding informs your decisions about resource allocation and encourages efficient usage behavior.

AOSCOST02: How do you choose appropriate storage tiering?

Implement a tiered data storage strategy for long-term data retention or infrequently accessed read-only data. This approach optimizes cost in OpenSearch domains by offloading less frequently used data to cost-effective storage options.

Best practices

- AOSCOST02-BP01 Use the latest Amazon EBS gp3 volumes with your OpenSearch Service nodes
- AOSCOST02-BP02 Use instances optimized for heavy indexing use cases
- AOSCOST02-BP03 Use the warm storage tier to optimize storage for a significant amount of read-only data
- AOSCOST02-BP04 Use the cold tier storage option to store and retrieve infrequently accessed or historical data

AOSCOST02-BP01 Use the latest Amazon EBS gp3 volumes with your OpenSearch Service nodes

Improve baseline performance and scalability by using the latest Amazon EBS gp3 volumes, which offer higher baseline performance and more scalable high performance.

Level of risk exposed if this best practice is not established: Low

Desired outcome: The latest Amazon EBS gp3 volumes are used with OpenSearch Service nodes to provide optimal performance, durability, and cost-effectiveness.

Benefits of establishing this best practice:

- **Improved baseline performance:** Using gp3 EBS volumes provides higher baseline performance compared to gp2 volumes.
- **Scalable high performance:** With gp3 volumes, you can provision higher performance independently of the volume size, allowing for more scalable and efficient performance in your OpenSearch Service domains.

Implementation guidance

OpenSearch Service launched support for the next generation, general purpose SSD (gp3) EBS volumes. OpenSearch Service data nodes require low latency and high throughput storage to provide fast indexing and query. We recommend that you consider gp3 as an effective Amazon EBS option for price, performance, and flexibility.

For more details about implementing this best practice for cost optimization, see AOSPERF03-BP03.

Resources

- Making configuration changes in Amazon OpenSearch Service

AOSCOST02-BP02 Use instances optimized for heavy indexing use cases

Improve indexing throughput and cost-effectiveness by using OR1 instances, which provide optimized storage for heavy indexing use cases.

Level of risk exposed if this best practice is not established: Low

Desired outcome: OR1 instances are used in Amazon OpenSearch Service to deliver price-performance for heavy indexing use cases.

Benefits of establishing this best practice:

- **Improved indexing throughput:** Use OR1 instances in Amazon OpenSearch Service for heavy indexing use cases, which provides increased indexing throughput due to the optimized storage structure.
- **Cost-effective data storage:** Using OR1 instances offers a cost-effective way to store large amounts of data, making it an ideal choice for applications with high indexing demands.

Implementation guidance

OR1 is an instance family for Amazon OpenSearch Service that provides a cost-effective way to store large amounts of data. A domain with OR1 instances uses Amazon Elastic Block Store (Amazon EBS) volumes for primary storage, with data copied synchronously to Amazon S3. This storage structure provides increased indexing throughput with high durability.

You can use OR1 instances with OpenSearch 2.11+ or upgrade to 2.15+ on existing domains.

Implementation steps

- Log in to the AWS Management Console.
- Navigate to the Amazon OpenSearch Service console.
- Create a new domain or modify an existing domain:
  - For a new domain, choose **Create domain**. For an existing domain, select the domain name and choose **Actions**, then **Edit cluster configuration**.

  - For new domains, select **Standard create** under Domain creation method tab.
- Under Data nodes tab, choose **OR1** from the Instance family drop down box.
- Proceed with other options.

Resources

- Introducing highly durable Amazon OpenSearch Service clusters with 30% price/performance improvement
- OR1 storage for Amazon OpenSearch Service
- Amazon OpenSearch Service Under the Hood: OpenSearch Optimized Instances (OR1)
- OpenSearch optimized instance (OR1) is game changing for indexing performance and cost
- Improve your Amazon OpenSearch Service performance with OpenSearch Optimized Instances

AOSCOST02-BP03 Use the warm storage tier to optimize storage for a significant amount of read-only data

Reduce costs, improve query performance, and enhance domain stability by storing a significant amount of read-only data in the warm tier.

Level of risk exposed if this best practice is not established: Low

Desired outcome: The warm tier is used in OpenSearch Service to store a significant amount of read-only data, reducing costs while maintaining data availability.

Benefits of establishing this best practice:

- Reduced costs due to optimized storage utilization
- Improved query performance through optimized data retrieval
- Enhanced domain stability by avoiding unnecessary resource utilization

Implementation guidance

Use data lifecycle management policies to move large amounts of read-only data to UltraWarm tier for low-latency queries and reduced costs.

To optimize costs and performance in Amazon OpenSearch Service, you can use the UltraWarm tier for storing a significant amount of read-only data.

Implementation steps

- Log in to the AWS Management Console.
- Navigate to the Amazon OpenSearch Service console.
- Select the domain name and choose **Actions**, then **Edit cluster configuration.**
- Enable UltraWarm:
  - In the Cluster configuration section, look for the option Enable warm data nodes located in the Warm and cold data storage box.

    > **Note**
    >
    > You need to have dedicated leader nodes in your domain to be able to use warm and cold storage.

  - Toggle on the option to enable UltraWarm nodes.
  - Select the Instance type.
  - Set the number of warm data nodes (it is required to have a minimum of three).
- After configuring the settings, review your setup and choose **Create** (for a new domain) or **Save changes** (for an existing domain).

Resources

- UltraWarm storage for Amazon OpenSearch Service

AOSCOST02-BP04 Use the cold tier storage option to store and retrieve infrequently accessed or historical data

Reduce costs and improve data management by storing infrequently accessed or historical data in the cold tier, maintaining efficient data retrieval.

Level of risk exposed if this best practice is not established: Low

Desired outcome: The cold tier storage option is used in Amazon OpenSearch Service to store and retrieve infrequently accessed or historical data efficiently, reducing costs while maintaining data availability.

Benefits of establishing this best practice:

- **Cost reduction**: Using the cold tier storage option in Amazon OpenSearch Service helps reduce costs associated with storing and retrieving infrequently accessed or historical data.
- **Efficient data management**: By using cold tier for infrequently accessed data, you can efficiently manage your data storage, making it easier to maintain data availability while minimizing storage expenses.

Implementation guidance

If you have data that must be preserved for compliance or regulatory reasons and or have historical value you can use cold tier to store such data at a lower cost than other storage tiers. cold storage is appropriate if you need to do periodic research or forensic analysis on your older data.

To optimize costs and performance in Amazon OpenSearch Service, you can use the cold tier storage option for storing and retrieving infrequently accessed or historical data.

Implementation steps

- Log in to the AWS Management Console.
- Navigate to the Amazon OpenSearch Service console.
- Select the domain name and choose **Actions**, then **Edit cluster configuration.**
- Enable UltraWarm:
  - In the Cluster configuration section, look for the option **Enable warm data nodes** located in the Warm and cold data storage box.

    > **Note**
    >
    > You need to have dedicated leader nodes in your domain to be able to use warm and cold storage.

  - Toggle on the option to enable UltraWarm nodes.
  - Select the **Instance type.**
  - Set the number of warm data nodes (required to have a minimum of three).

- Toggle on the option **Enable cold storage.**
- After configuring the settings, review your setup and choose **Create** (for a new domain) or **Save changes** (for an existing domain).

Resources

- Cold storage for Amazon OpenSearch Service

Cost-effective resources

OpenSearch Service offers a range of cost-effective resources that can help you optimize your search infrastructure costs.

AOSCOST03: How do you take advance of capacity reservation commitments to reduce costs?

For long-running applications with predictable usage, reserved instance offers significant discounts as compared to On-Demand Instances. If you have consistent usage for the next one to three years, Reserved Instances (RIs) can provide significant discounts. RIs are essentially a billing discount applied to On-Demand Instances within your account.

Best practices

- AOSCOST03-BP01 Evaluate forecasting for your workloads

AOSCOST03-BP01 Evaluate forecasting for your workloads

Reduce costs and improve predictability by setting up Reserved Instances for consistent workloads over one to three years, which can result in significant cost savings.

Level of risk exposed if this best practice is not established: Low

Desired outcome: The workload is consistent for the next one to three years, allowing for Reserved Instances to be set up and used effectively.

Benefits of establishing this best practice:

- **Significant cost savings:** Using Reserved Instances for a consistent workload over one to three years can result in significant cost savings, as you're committing to a fixed price for that period.
- **Predictable expenses:** By reserving instances upfront, you can improve the predictability of your expenses for the next one to three years, which can help with budgeting and financial planning.

Implementation guidance

Maximize savings for your OpenSearch Service domain by using Reserved Instances (RIs) if your workload is consistent over the next one to three years. You can reserve instance using AWS CLI, AWS Management Console, or AWS SDKs. You can also use AWS Cost Explorer to examine the usage pattern and then decide to buy Reserved Instances (RIs).

Implementation steps

- Log in to AWS Management Console.
- Navigate to the Amazon OpenSearch Service console.
- Choose **Reserved Instance leases**, located under **Managed clusters** in the left navigation pane.
- Choose **Order Reserved Instances**.
- Proceed with the options of your choice.

Resources

- Reserved Instances in Amazon OpenSearch Service

Optimize over time

You can optimize cost over time by reviewing new instance types and implementing them in your workload. As AWS releases new instances and features, it is a best practice to review your existing architectural decisions to verify that they remain cost effective. As your requirements change, be aggressive in decommissioning resources, components, and workloads that you no longer require.

Consider the following best practices to help you optimize over time. While optimizing your workloads over time and improving your CFM culture in your organization, evaluate the cost of effort for operations in the cloud, review your time-consuming cloud operations, and automate them to reduce human efforts and cost by adopting related AWS services, third-party products, or custom tools (like AWS CLI or AWS SDKs)

AOSCOST04: How can regular monitoring and tracking of costs help achieving cost optimization?

Actively monitoring the usage and cost of Amazon OpenSearch Service is crucial for cost optimization because it allows you to identify inefficiencies, optimize resource allocation, budget management and plan for growth. You can use service metrics to monitor actively the performance metrics and costs of OpenSearch Service.

AOSCOST05: How do estimate your OpenSearch Service cost?

It is important to estimate costs for the services you use. We recommend using AWS Pricing Calculator to estimate costs.

Best practices

- AOSCOST04-BP01 Apply cost allocation tags to your OpenSearch resources for detailed cost tracking and analysis
- AOSCOST05-BP01 Assess the pricing for instances and storage in Amazon OpenSearch Service
- AOSCOST05-BP02 Examine the costs associated with Amazon S3 storage for manually creating snapshots of your OpenSearch Service domain

AOSCOST04-BP01 Apply cost allocation tags to your OpenSearch resources for detailed cost tracking and analysis

Improve cost visibility and enhance financial reporting by applying cost allocation tags to track and analyse costs associated with each resource in your OpenSearch Service domain.

Level of risk exposed if this best practice is not established: Low

Desired outcome: Cost allocation tags are applied to enable detailed cost tracking and analysis of your OpenSearch Service domain.

Benefits of establishing this best practice:

- **Improved cost visibility**: Applying cost allocation tags to Amazon OpenSearch Service resources enables detailed cost tracking and analysis using AWS Cost Explorer, providing a clear understanding of costs associated with each resource.
- **Enhanced financial reporting**: By categorizing workloads using key-value pairs (for example, environment, project, department), you can provide accurate financial reporting and separate costs for better expense management.

Implementation guidance

- Assign unique tags to your OpenSearch Service domains, and use key-value pairs to categorize workloads by attributes such as environment (like development, production, and QA testing), project, or department.
- Use tags to track cost usage and manage expenses with AWS Cost Explorer. This best practice helps you segment your costs and improve the accuracy of your financial reporting.
- For more detail on tagging, see Tagging OpenSearch Service domains.

Resources

- Using AWS cost allocation tags
- What is AWS Billing and Cost Management and Cost Management?

AOSCOST05-BP01 Assess the pricing for instances and storage in Amazon OpenSearch Service

Improve cost transparency and accuracy of cost estimation by assessing pricing for instances and storage to make informed decisions on resource allocation.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You can estimate costs by assessing the pricing for instances and storage in Amazon OpenSearch Service.

Benefits of establishing this best practice:

- **Cost transparency:** Assessing pricing for instances and storage in Amazon OpenSearch Service provides a clear understanding of the associated costs, including On-Demand and Reserved Instances, as well as storage options.

- **Accurate cost estimation:** By using tools such as AWS Pricing Calculator, you can estimate costs, make informed decisions about resource allocation, and keep your project within budget.

Implementation guidance

Understand cost associated with On-Demand and Reserved Instances, as well as cost associated with storage options. For a better understanding of how to estimate the costs for Amazon OpenSearch Service, see Amazon OpenSearch Service Pricing, Supported instance types in Amazon OpenSearch Service, and AWS Pricing Calculator.

Resources

- Amazon OpenSearch Service Pricing
- AWS Pricing Calculator

AOSCOST05-BP02 Examine the costs associated with Amazon S3 storage for manually creating snapshots of your OpenSearch Service domain

Improve cost awareness and inform backup strategy by examining Amazon S3 storage costs associated with manually creating snapshots of your OpenSearch Service domain.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You can estimate costs by examining the costs associated with Amazon S3 storage for manually creating snapshots of your OpenSearch Service domain.

Benefits of establishing this best practice:

- **Cost awareness**: Examining the costs associated with Amazon S3 storage for manually creating snapshots of your OpenSearch Service domain helps you understand that backups can incur standard Amazon S3 usage charges.
- **Informed backup strategy**: By considering the cost of manual backups, you can choose the most suitable backup strategy for your needs, whether it's manual backups or using AWS services like Automated Backups or Snapshots in OpenSearch.

Implementation guidance

If you take manual backups of your OpenSearch Service domain, understand that the manual snapshots storing in Amazon S3 can incur standard Amazon S3 usage charges. For detailed information about the cost of the different storage tiers available in Amazon S3, see Amazon S3 pricing.

Resources

- Amazon S3 pricing
- Amazon OpenSearch Service Pricing

Key AWS services

- Amazon OpenSearch Service: A fully managed service that helps you search, ingest, and analyze data in real-time.
- Amazon CloudWatch: Monitoring and logging service that helps you monitor and troubleshoot your AWS resources.
- Amazon SNS: A fully managed messaging service that allows you to fan out messages to multiple subscribers.
- Amazon CloudWatch Alarms: A feature of CloudWatch that allows you to set up alarms based on metrics and receive notifications when thresholds are breached.

Resources

- Supported instance types in Amazon OpenSearch Service
- Sizing Amazon OpenSearch Service domains
- Dedicated leader nodes in Amazon OpenSearch Service
- Making configuration changes in Amazon OpenSearch Service
- OR1 storage for Amazon OpenSearch Service
- UltraWarm storage for Amazon OpenSearch Service
- Cold storage for Amazon OpenSearch Service
- Reserved Instances in Amazon OpenSearch Service
- What is AWS Billing and Cost Management and Cost Management?

- Using AWS cost allocation tags
- Amazon OpenSearch Service Pricing
- AWS Pricing Calculator

Sustainability

The sustainability pillar helps you reduce your workload's environmental impact while optimizing costs. By improving energy efficiency and resource utilization across your architecture components, you can decrease both carbon emissions and operational expenses.

Environmental sustainability extends beyond carbon reduction to include economic, social, and ecological considerations. To address climate change, organizations can implement resource-efficient architectures that minimize waste and use renewable energy sources. This pillar shows you how to deploy and maintain OpenSearch nodes with sustainability best practices that reduce both carbon emissions and costs.

Focus areas

- Design principles
- Region selection
- Software and architecture
- Data management
- Process and culture
- Key AWS services
- Resources

Design principles

- **Choose Regions strategically:** Choose Regions based on business requirements and sustainability goals, considering factors like energy consumption, carbon footprint, and proximity to users.
- **Select sustainable instance families:** Select instance families that align with your sustainability objectives, such as Graviton-based instances, which are more environmentally friendly due to their reduced energy consumption.
- **Optimize resource utilization:** Optimize resource utilization by using the minimum number of instances required, reducing unnecessary data, and using Index State Management (ISM) to manage dataset lifecycle.
- **Consolidate workloads:** Consolidate development and test workloads into fewer OpenSearch domains, using security features for protection and reducing the overall environmental impact.

Region selection

The choice of Region for your workload significantly affects its KPIs, including performance, cost, and carbon footprint. To improve these KPIs effectively, you should choose Regions for your workloads based on both business requirements and sustainability goals.

AOSSUS01: How do you choose Regions for your OpenSearch Service workloads?

Choosing AWS Regions for your OpenSearch Service workload involves considering various factors to optimize performance, availability, costs, and carbon footprint. To improve these KPIs effectively, you should choose your OpenSearch Service Region for your workloads based on both business requirements and sustainability goals.

The following are some key considerations when selecting AWS Regions for OpenSearch Service:

- **Renewable energy:** Choose Regions near Amazon renewable energy projects and Regions where the grid has a published carbon intensity that is lower than other locations (or Regions).
- **Proximity to users:** Choose a Region that is geographically close to your users. Test the network latency between your end user locations and each AWS Region that meets your sustainability goals. Selecting a Region with low latency is crucial for applications where user experience is a priority.
- **Compliance and data residency:** Consider regulatory requirements and data residency constraints. Verify that the AWS Region you choose complies with legal or regulatory restrictions on data storage and processing. Some industries or countries may have specific regulations that dictate where data must be stored.
- **High availability and fault tolerance:** Enhance fault tolerance and high availability by distributing your Amazon OpenSearch Service domain across multiple Availability Zones within a Region. This makes your workload resilient to failures in a single Availability Zone. Choose a Region that provides multiple Availability Zones for redundancy. Improve customer experience by taking advantage of the cross-replication feature of Amazon OpenSearch Service to create clusters across different geographical Regions.
- **Service and feature availability:** Check if the AWS Region supports all the features and services you intend to use with OpenSearch Service. Some AWS services and features might be available in specific Regions first before being rolled out globally. Verify that the necessary services and features for your workload are supported in your chosen Region.

- **Global applications:** If your application serves a global user base, evaluate whether a multi-Region setup is necessary. This involves deploying OpenSearch domains in multiple AWS Regions and using cross-cluster replication to provide optimal performance and availability for users in different geographic locations.

Best practices

- AOSSUS01-BP01 Select the Region for your OpenSearch Service deployment based on a combination of business requirements and sustainability goals

AOSSUS01-BP01 Select the Region for your OpenSearch Service deployment based on a combination of business requirements and sustainability goals

Improve alignment with business objectives, meet compliance requirements, and reduce non-compliance risk by selecting the OpenSearch Service Region that best aligns with your company's needs and sustainability goals.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You choose a Region for the OpenSearch Service deployment based on a combination of business requirements and sustainability goals.

Benefits of establishing this best practice:

- Improved alignment with business objectives and goals
- Enhanced ability to meet compliance and regulatory requirements
- Reduced risk of non-compliance due to incorrect Region selection

Implementation guidance

The selection of a Region for your workload plays a crucial role in influencing its key performance indicators (KPIs), such as performance, cost, and carbon footprint. To enhance these KPIs effectively, choose Regions for your workloads based on both business requirements and sustainability goals.

The following guidance is provided to aid your selection of most sustainable Regions in your area:

- Shortlist potential Regions based on the following topics:
  - Data security and privacy issues
  - Regulatory compliance requirements
  - The operational efficiency of your workloads
  - Local data sovereignty concerns
  - Several services and features that optimize sustainability
- Select Regions by market-based or location-based methods in line with your financial services industry's internal relevant sustainability guidelines that are used to track and to compare your organization's year-to-year emissions.
- Wherever possible, choose a Region that provides better than 95% renewable energy, using the market-based method and low grid carbon intensity, as well as using a typical location-based method.

Resources

- Understanding your carbon emission estimations
- Amazon Around the Globe
- Renewable Energy Methodology
- What to Consider when Selecting a Region for your Workloads
- How to select a Region for your workload based on sustainability goals

Software and architecture

AOSSUS02: How do you use instance families in OpenSearch Service to support your organization's sustainability goals?

Use OpenSearch instance families that align with sustainability objectives. Certain OpenSearch instance families are designed to be more energy-efficient than others. Choosing energy-efficient instances contributes to the overall reduction in power consumption and carbon footprint, helping lower the environmental impact of your infrastructure.

Remember that sustainability is not only about energy efficiency but also about optimizing resource usage. Choosing appropriate instance families helps in resource optimization, which helps you use computing resources efficiently without unnecessary waste. This can lead to cost savings as well.

Best practices

- AOSSUS02-BP01 Evaluate instances in alignment to sustainability goals
- AOSSUS02-BP02 Use the minimum number of instances necessary to meet your workload requirements

AOSSUS02-BP01 Evaluate instances in alignment to sustainability goals

Reduce carbon footprint, improve sustainability performance, and enhance regulatory compliance by evaluating instance choices that align with your organization's sustainability goals.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You use energy-efficient instances that help reduce your carbon footprint.

Benefits of establishing this best practice:

- Reduced carbon footprint due to increased energy efficiency
- Improved sustainability performance and compliance with organizational goals
- Enhanced ability to meet environmental regulations and standards

Implementation guidance

AWS Graviton-based instances use up to 60% less energy than comparable instances.

- **Identify supported Graviton instance types:** Explore the available Graviton-based instance types, such as r7g and r6g, that are optimized for OpenSearch and energy efficiency.
- **Search for suitable instance types:** Explore Amazon OpenSearch Service pricing to find Graviton-based instances that align with your workload and financial requirements. You can filter results by Region to narrow down the options.
- **Conduct performance testing:** Run rigorous performance tests to determine the most suitable Graviton instance type for your specific workload.

Resources

- Supported instance types in Amazon OpenSearch Service
- Amazon OpenSearch Service Pricing
- OpenSearch Benchmark tool

AOSSUS02-BP02 Use the minimum number of instances necessary to meet your workload requirements

Reduce costs, improve resource utilization, and eliminate unnecessary resource usage by using the minimum number of instances necessary to meet your workload requirements.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: Your workload uses the minimum number of instances necessary to meet requirements, reducing unnecessary resource usage.

Benefits of establishing this best practice:

- Reduced costs and increased cost efficiency
- Improved resource utilization and reduced waste

Implementation guidance

Use the minimum amount of hardware to meet your workload and business needs efficiently.

- **Determine the required instance type:** Based on your workload characteristics, such as query volume, document size, indexing needs, and storage requirements, select a suitable Graviton instance type that meets your performance requirements.
- **Estimate instance capacity:** Estimate the total capacity required to meet your workload demands. For detail on sizing OpenSearch Service domains, see Performance efficiency and Cost optimization.

Resources

- Sizing OpenSearch Service domains

Data management

AOSSUS03: How do you align data management practices with environmental sustainability objectives?

Implementing data management practices in OpenSearch Service is vital for achieving sustainability objectives, improving the efficiency of resource utilization, and achieving environmentally-responsible operations. The significance of incorporating these practices is underscored by several key factors.

Efficient data lifecycle management, involving well-defined policies for data retention and removal, prevents unnecessary storage costs and mitigates the environmental impact associated with data storage. Similarly, the application of policies addressing data redundancy, compression, and deduplication results in a reduced storage footprint, optimizing costs and minimizing the need for additional hardware resources.

Adopting ISM policies automates the lifecycle management of indices and enables transitions based on criteria such as age or size. This approach facilitates the movement of infrequently accessed data to lower-cost storage solutions, which reduces the environmental impact associated with maintaining high-performance storage for rarely accessed data.

Efficient data management patterns contribute to the optimization of computing resources through proper indexing strategies, shard management, and resource allocation. This optimization minimizes energy consumption and overall resource usage within the OpenSearch cluster, which helps your organization align with sustainability objectives.

Finally, effective data management policies play a crucial role in optimizing indexing and query patterns. This involves selecting appropriate data types, using efficient queries, and designing indices that align with the nature of the data. The outcome is enhanced performance, faster response times, reduced resource usage, and a reinforced commitment to sustainability goals in the context of Amazon OpenSearch Service.

Best practices

- AOSSUS03-BP01 Use Index State Management to manage the lifecycle of your dataset
- AOSSUS03-BP02 Reduce unnecessary or redundant data from your domain

- AOSSUS03-BP03 Take manual snapshots of your indices only when it is difficult to recreate the dataset

AOSSUS03-BP01 Use Index State Management to manage the lifecycle of your dataset

Improve data management, meet compliance requirements, and reduce data loss risk by using Index State Management (ISM) to manage the lifecycle of your dataset.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: The lifecycle of the dataset is managed using Index State Management (ISM) to support sustainability goals.

Benefits of establishing this best practice:

- Improved data management and organization
- Enhanced ability to meet data compliance and regulatory requirements
- Reduced risk of data loss or corruption due to proper data lifecycle management

Implementation guidance

ISM removes the necessity for establishing and overseeing external processes to run your index operations. It also allows you to perform operations like moving data from Hot storage to Warm.

- Implement a data classification policy to understand its criticality to business outcomes and choose the right energy-efficient storage tier. Determine criticality, confidentiality, integrity, and availability of data based on risk to the organization.
- Evaluate your data characteristics and access pattern to collect the key characteristics of your storage needs. Key characteristics to consider include:
  - **Data type:** Structured, semistructured, unstructured
  - **Data growth:** Bounded, unbounded
  - **Data durability:** Persistent, ephemeral, transient
  - **Access patterns:** Reads or writes, frequency, spiky, or consistent

- Implement an ISM policy to move data between different storage tiers to meet business and regulatory requirements. Additionally, implement an ISM policy to delete unnecessary data. For detailed implementation guidance, see AOSOPS01-BP01.

Resources

- Index State Management in Amazon OpenSearch Service

AOSSUS03-BP02 Reduce unnecessary or redundant data from your domain

Reduce storage costs, improve resource utilization, and enhance resource management by removing unnecessary or redundant data from your domain.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You have removed unnecessary or redundant data from your domain to support sustainability goals.

Benefits of establishing this best practice:

- Reduced storage costs and increased cost efficiency
- Improved resource utilization and reduced waste
- Enhanced ability to manage and optimize resources

Implementation guidance

You can reduce unneeded and redundant data through various methods, like using indexing strategies, implementing ISM, and archiving data.

- Run `_cat/indices?v` to list your indices
- Use `DELETE /<index-name>` to remove unnecessary or redundant indices

Resources

- Index State Management in Amazon OpenSearch Service
- Cat Indices

- Delete Index

AOSSUS03-BP03 Take manual snapshots of your indices only when it is difficult to recreate the dataset

Reduce unnecessary snapshot creation, Amazon EBS, and Amazon S3 storage costs by taking manual snapshots only when it's difficult to recreate the dataset.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You take manual snapshots of indices only when it is difficult to recreate the dataset, reducing unnecessary snapshot creation and Amazon S3 storage.

Benefits of establishing this best practice:

- Reduced Amazon EBS and Amazon S3 storage costs
- Improved resource utilization and reduced waste

Implementation guidance

Snapshots in Amazon OpenSearch Service serve as backups for a domain's indexes and state. Excessive snapshot leads to unnecessary storage and energy wastage.

Delete unneeded snapshots using `DELETE _snapshot/repository-name/snapshot-name`.

Resources

- Deleting manual snapshots

Process and culture

AOSSUS04: How do you optimize your development and test environment footprint?

Consolidating development and test workloads into fewer OpenSearch domains results in a reduced hardware footprint, contributing to energy and resource savings.

A smaller hardware footprint aligns with sustainability objectives by decreasing the environmental impact, leads to more efficient use of computing resources, is more energy-efficient, lowers carbon emissions, avoids redundancy of data, and provides cost savings.

Best practices

- AOSSUS04-BP01 Consolidate OpenSearch Service domain environments

AOSSUS04-BP01 Consolidate OpenSearch Service domain environments

Reduce costs, improve resource utilization, and enhance data protection by consolidating development and test workloads into fewer OpenSearch Service domains with implemented security features.

Level of risk exposed if this best practice is not established: Medium

Desired outcome: You consolidate development and test workloads into fewer OpenSearch Service domains, with security features implemented to protect data.

Benefits of establishing this best practice:

- Reduced costs and increased cost efficiency
- Improved resource utilization and reduced waste
- Enhanced ability to manage and optimize resources
- Improved data protection and compliance due to security features

Implementation guidance

Combining multiple workloads into a smaller number of OpenSearch Service domains not only minimizes compute and storage waste but also contributes to a reduction in carbon emissions.

Additionally, you can use fine-grained access control to implement tight security controls over your data.

Resources

- Storing Multi-Tenant SaaS Data with Amazon OpenSearch Service
- Build a multi-tenant serverless architecture in Amazon OpenSearch Service

- Building Multi-Tenant Solutions with Amazon OpenSearch Service
- Security in Amazon OpenSearch Service

Key AWS services

- Amazon OpenSearch Service
- Amazon EC2
- Graviton-based instances: A type of instance family that uses up to 60% less energy than comparable instances.
- Amazon Around the Globe: A resource that provides information on AWS Regions and their locations.
- Renewable Energy Methodology: A resource that provides information on AWS's renewable energy initiatives.

Resources

- Understanding your carbon emission estimations
- Amazon Around the Globe
- Renewable Energy Methodology
- What to Consider when Selecting a Region for your Workloads
- How to select a Region for your workload based on sustainability goals
- Supported instance types in Amazon OpenSearch Service
- Sizing Amazon OpenSearch Service domains
- Index State Management in Amazon OpenSearch Service
- Deleting manual snapshots
- Storing Multi-Tenant SaaS Data with Amazon OpenSearch Service
- Build a multi-tenant serverless architecture in Amazon OpenSearch Service
- Building Multi-Tenant Solutions with Amazon OpenSearch Service
- Security in Amazon OpenSearch Service

Conclusion

The Amazon OpenSearch Service Lens provides comprehensive guidance for organizations building search and analytics workloads on AWS. It offers best practices and strategies across the six pillars of Well-Architected, helping you design OpenSearch domains that are secure, reliable, performant, and cost-effective. By following the principles outlined in this lens, you can build and operate OpenSearch workloads that meet your business requirements while maintaining operational excellence and sustainability.

Throughout this document, you've gained valuable insights into key aspects of optimizing OpenSearch Service workloads. From fine-tuning shard and CPU ratios for optimal performance to implementing automated data lifecycle management through ISM, this lens covers essential topics for successful OpenSearch implementations. It emphasizes the importance of proper backup strategies using manual snapshots to customer-owned S3 buckets and highlights how newer instance generations can provide better cost-performance ratios. The lens also stresses the significance of monitoring, automation, and continuous optimization to ensure your OpenSearch domains remain efficient and adaptable to changing requirements.

By applying the best practices and recommendations provided in this lens, organizations can confidently navigate the complexities of search and analytics workloads. Whether you're building real-time search applications, log analytics solutions, or security information and event management (SIEM) systems, this lens offers practical guidance to help you achieve your objectives while adhering to Well-Architected principles.

As organizations scale their OpenSearch Service workloads, it's important to view this lens as a living document. Search and analytics requirements evolve continuously, and new features and capabilities are regularly added to OpenSearch Service. Therefore, it's crucial to stay current with the latest best practices and periodically reassess your OpenSearch domains to ensure they remain well-architected and aligned with your business needs.

In conclusion, the Amazon OpenSearch Service Lens is an essential resource for anyone involved in designing, deploying, and operating OpenSearch workloads on AWS. By following the guidance outlined in this document, organizations can harness the full potential of OpenSearch Service while building solutions that are secure, reliable, and cost-effective. As search and analytics requirements continue to grow in complexity, this lens will serve as a valuable guide, enabling organizations to innovate and derive meaningful insights from their data through the power of OpenSearch Service.

Contributors

The following individuals and organizations contributed to this document:

- Muslim Abu-Taha, Senior OpenSearch Solutions Architect, EMEA, Amazon Web Services
- Shih-Yong Wang, Manager, Solutions Architecture, Amazon Web Services
- Ankush Agarwal, Solutions Architect II, Amazon Web Services
- Jun-Tin Yeh, Cloud Optimization Success Solutions Architect, Amazon Web Services
- Cedric Pelvet, Principal OpenSearch Solutions Architect, EMEA, Amazon Web Services
- Hajer Bouafif, Senior OpenSearch Solutions Architect, EMEA, Amazon Web Services
- Francisco Losada, OpenSearch Solutions Architect, EMEA, Amazon Web Services
- Bharav Patel, OpenSearch Solutions Architect, EMEA, Amazon Web Services
- Praveen Prasad, Senior Specialist Technical Account Manager, EMEA, Amazon Web Services
- Mahmoud Matouk, Principal Security Lead SA, Amazon Web Services
- Ryan Dsouza, Principal Guidance Lead SA, Amazon Web Services
- Arvind Raghunathan, Principal Operations Lead SA, Amazon Web Services
- Steven Arita, Cloud Optimization Success SA, Amazon Web Services
- Bruce Ross, Well-Architected Lens Leader, Amazon Web Services
- Stewart Matzek, Sr. Technical Writer,, Amazon Web Services
- Madhuri Srinivasan, Sr. Technical Writer, Amazon Web Services
- Matthew Wygant, Sr. TPM Guidance, Amazon Web Services

Document revisions

Change	Description	Date
Initial release	Initial release of the Amazon OpenSearch Service Lens.	June 23, 2025

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the *AWS Glossary Reference*.