AWS Prescriptive Guidance

MongoDB Atlas on AWS: Migrating from a self-managed environment to the AWS Cloud

AWS Prescriptive Guidance: MongoDB Atlas on AWS: Migrating from a self-managed environment to the AWS Cloud

Table of Contents

MongoDB Atlas on AWS: Migrating from a self-managed environment to the AWS Cloud

Suresh Veeragoni, Senior Partner Solutions Architect, Amazon Web Services

Paresh Saraf, Senior Partner Solutions Architect, MongoDB

October 2020

This guide describes the architecture, tools, considerations, and best practices for migrating from a self-managed MongoDB environment (including MongoDB Community Server, Enterprise Server, Enterprise Advanced, mLab, or any managed MongoDB cluster) to MongoDB Atlas on Amazon Web Services (AWS). For the migration steps, see the pattern Migrate a self-hosted MongoDB environment to MongoDB Atlas on the AWS Cloud on the AWS Prescriptive Guidance website.

If you want to run MongoDB Atlas in the AWS Cloud, get started with MongoDB Atlas in AWS Marketplace.

MongoDB is a document database built on a scale-out architecture for developers who build scalable web and business applications by using agile methodologies. Development teams use MongoDB for the following reasons:

- The document data model is a powerful way to store and retrieve data.
- MongoDB’s horizontal, scale-out architecture can support large volumes of both data and traffic.
- Developers can install MongoDB and start writing code immediately.

MongoDB Atlas is a fully managed database as a service (DBaaS) that’s available on all major public cloud providers, including AWS. MongoDB Atlas supports:

- A single view that aggregates data from multiple sources
- Real-time data analysis
- Internet of Things (IoT) applications
- Scalable mobile applications
- Database personalization at scale
- Catalogs of database assets, entities, and metadata that simplify data storage, data retrieval, and schema changes
- Ability to store and manage any type of content
- Ability to offload mainframe applications to modern platforms
- Database support for gaming applications

For more information about these features, see MongoDB Use Cases on the MongoDB website.

The guide is intended for AWS Service Integrator Partners (SI Partners) and AWS users.

Overview

This section provides a summary of the migration environment, considerations, and assumptions; required skills; and associated costs and licenses.

MongoDB migration at a glance

Migration use cases		Single view, IoT, real-time analytics, mobile, personalization, catalog, content management, mainframe offloading, gaming
Workload	Source workload	MongoDB (self-managed) or MongoDB on mLab, ObjectRocket, or Compose
	Source environment	Any MongoDB environment
	Destination workload	MongoDB Atlas
	Destination environment	MongoDB Atlas on AWS
Migration	Migration strategy (7 Rs (p. 18))	Replatform
	Is this an upgrade in workload version?	Not applicable. The version doesn't affect migration.
	Is the source workload different from the ISV workload?	No
	Migration duration	3 days to 2 weeks (including all the tasks in the associated pattern). The duration might vary based on data size, application complexity, network bandwidth between source and target, and similar factors.
	Quick Start to automatically provision workload infrastructure on AWS	Coming soon
Cost	Cost of running ISV workload on AWS	See MongoDB Pricing on the MongoDB website.
Assumptions and prerequisites	System limitations (minimum/ maximum requirements)	No
	Service-level agreements (SLAs)	For availability, see MongoDB Atlas Service Level Agreement on the MongoDB website.

		Cluster durability depends on a cluster’s geographical deployment configuration. For cluster limits, connection limits, organization and project limits, see Atlas Limitations on the MongoDB website.
	Recovery time objectives (RTOs)	For information about how MongoDB Atlas automated backup features can meet your strict RTOs, see Reliability on the MongoDB website.
	Recovery point objectives (RPOs)	For information about how MongoDB Atlas point-in-time recovery features satisfy RPOs, see Reliability and Cloud Backups on the MongoDB website.
	Licensing and operating model for the target AWS account	SaaS
	Migration tooling	Atlas Live Migration Service
	AWS services used	• Amazon Elastic Compute Cloud (Amazon EC2) • Amazon Virtual Private Cloud (Amazon VPC)
	Benchmarks	See MongoDB Benchmark on the MongoDB website.
Compliance	Compliance certifications	HIPAA, GDPR, SOC, EU-US Privacy Shield, ISO/IEC 27001:2013, PCI DSS. For details, see Trust Center on the MongoDB website.

Roles and responsibilities

The process for migrating data to MongoDB Atlas involves the following roles and expertise.

Role	Responsibilities
MongoDB DBA and application architect	• Assess and size the workload. • Deploy the database and configure security settings. • Migrate and monitor the database. • Implement the cutover plan.
AWS architect	• Advise and provision AWS services and infrastructure resources.

Role	Responsibilities
	• Establish AWS Identity and Access Management (IAM) roles for users and groups. • Provide security administration for all provisioned AWS services.

Cost and licensing

The pricing for MongoDB Atlas includes three types of costs:

- Cluster configuration costs
- Data transfer costs
- Costs for subscriptions and additional services

For more information, see Billing in the MongoDB documentation.

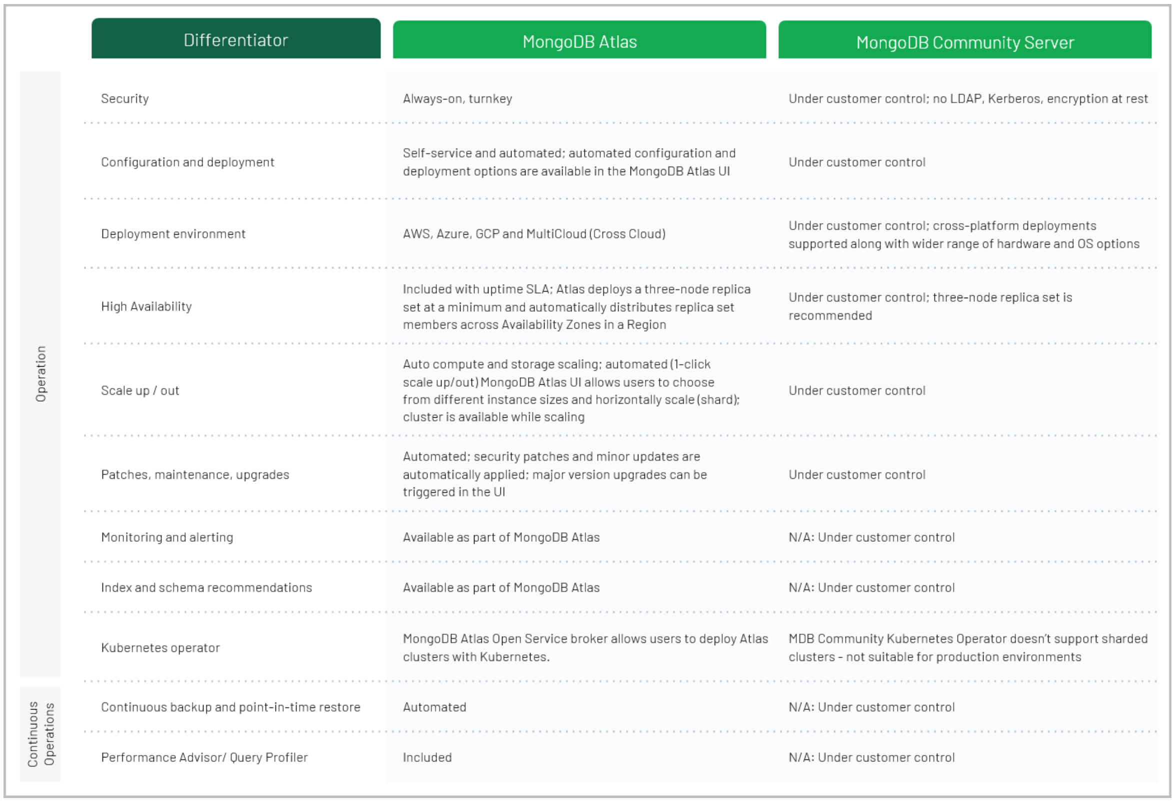
MongoDB Atlas can be purchased directly from AWS Marketplace, which simplifies the billing process. You can choose from these two MongoDB offerings in AWS Marketplace:

- Atlas Starter Package
- Atlas Pro Package

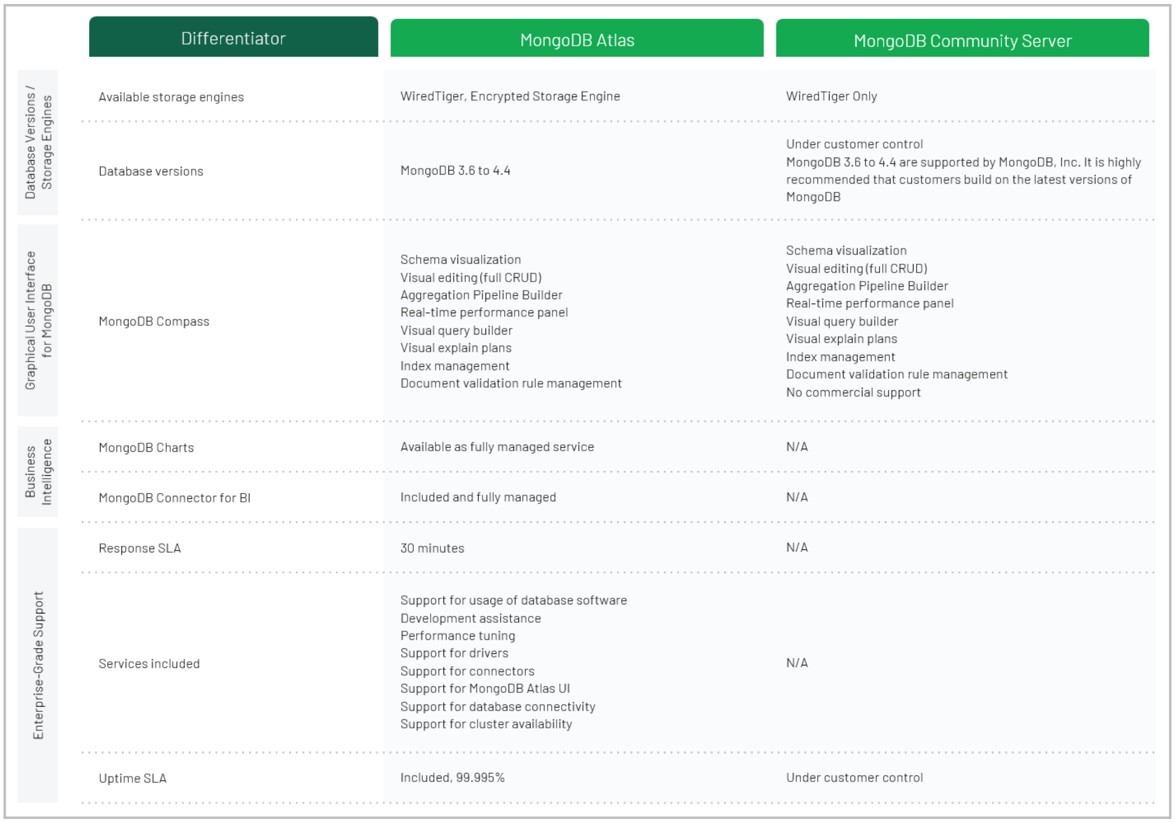
AWS systems integration (SI) Partners can extend private offers to their customers to meet specific requirements or use cases. To extend a private offer, contact MongoDB.

MongoDB and MongoDB Atlas feature comparison

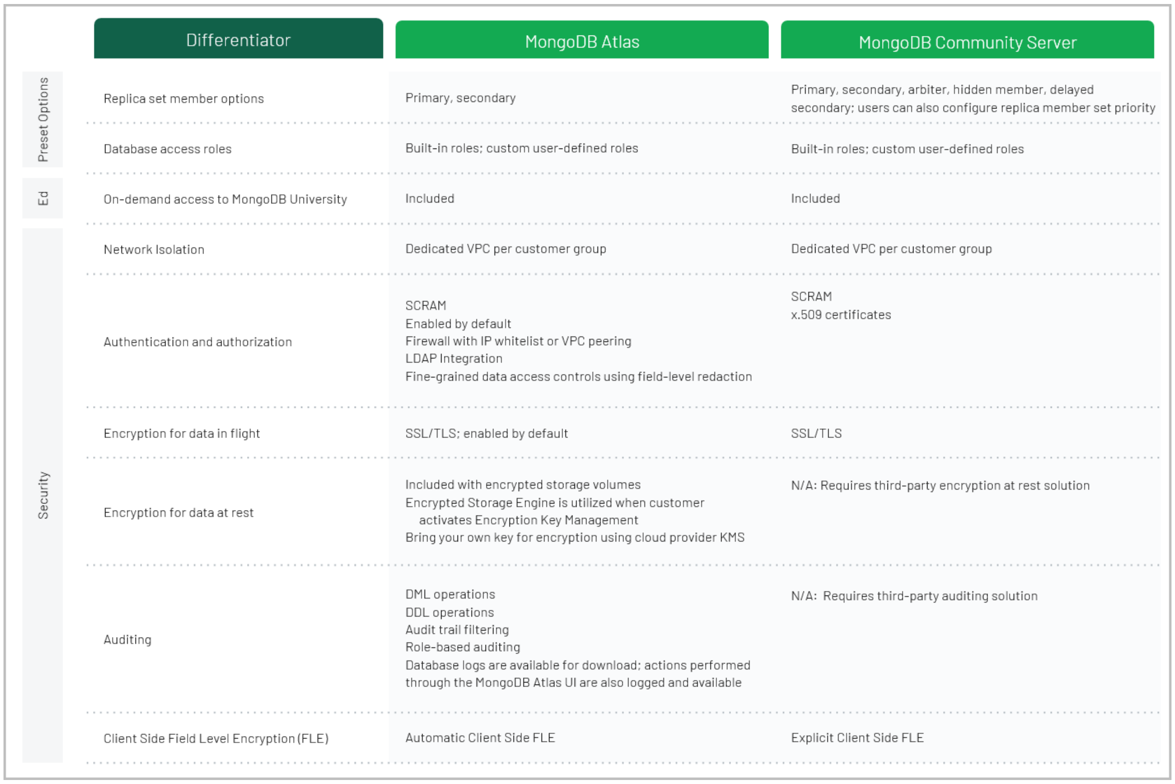
The following tables provide a comparison of MongoDB Community Server and MongoDB Atlas Enterprise features.



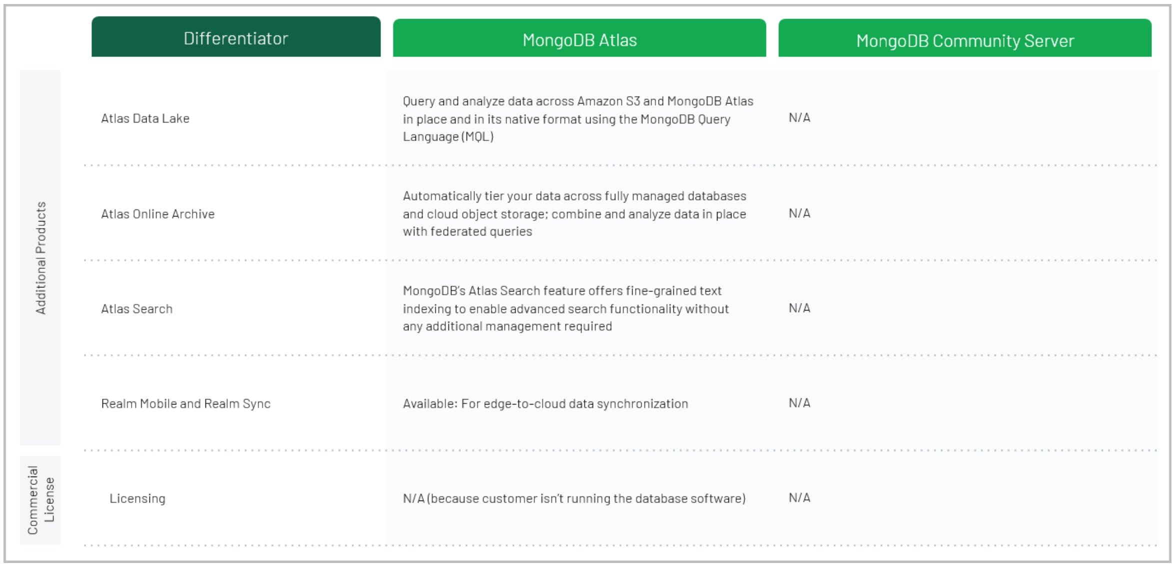
	Differentiator	MongoDB Atlas	MongoDB Community Server
Operation	Security	Always-on, turnkey	Under customer control; no LDAP, Kerberos, encryption at rest
	Configuration and deployment	Self-service and automated; automated configuration and deployment options are available in the MongoDB Atlas UI	Under customer control
	Deployment environment	AWS, Azure, GCP and MultiCloud (Cross Cloud)	Under customer control; cross-platform deployments supported along with wider range of hardware and OS options
	High Availability	Included with uptime SLA; Atlas deploys a three-node replica set at a minimum and automatically distributes replica set members across Availability Zones in a Region	Under customer control; three-node replica set is recommended
	Scale up / out	Auto compute and storage scaling; automated (1-click scale up/out) MongoDB Atlas UI allows users to choose from different instance sizes and horizontally scale (shard); cluster is available while scaling	Under customer control
	Patches, maintenance, upgrades	Automated; security patches and minor updates are automatically applied; major version upgrades can be triggered in the UI	Under customer control
	Monitoring and alerting	Available as part of MongoDB Atlas	N/A: Under customer control
	Index and schema recommendations	Available as part of MongoDB Atlas	N/A: Under customer control
	Kubernetes operator	MongoDB Atlas Open Service broker allows users to deploy Atlas clusters with Kubernetes.	MDB Community Kubernetes Operator doesn't support sharded clusters - not suitable for production environments
Continuous Operations	Continuous backup and point-in-time restore	Automated	N/A: Under customer control
	Performance Advisor/ Query Profiler	Included	N/A: Under customer control



	Differentiator	MongoDB Atlas	MongoDB Community Server
Database Versions / Storage Engines	Available storage engines	WiredTiger, Encrypted Storage Engine	WiredTiger Only
	Database versions	MongoDB 3.6 to 4.4	Under customer control MongoDB 3.6 to 4.4 are supported by MongoDB, Inc. It is highly recommended that customers build on the latest versions of MongoDB
Graphical User Interface for MongoDB	MongoDB Compass	Schema visualization Visual editing (full CRUD) Aggregation Pipeline Builder Real-time performance panel Visual query builder Visual explain plans Index management Document validation rule management	Schema visualization Visual editing (full CRUD) Aggregation Pipeline Builder Real-time performance panel Visual query builder Visual explain plans Index management Document validation rule management No commercial support
Business Intelligence	MongoDB Charts	Available as fully managed service	N/A
	MongoDB Connector for BI	Included and fully managed	N/A
Enterprise-Grade Support	Response SLA	30 minutes	N/A
	Services included	Support for usage of database software Development assistance Performance tuning Support for drivers Support for connectors Support for MongoDB Atlas UI Support for database connectivity Support for cluster availability	N/A
	Uptime SLA	Included, 99.995%	Under customer control



	Differentiator	MongoDB Atlas	MongoDB Community Server
Preset Options	Replica set member options	Primary, secondary	Primary, secondary, arbiter, hidden member, delayed secondary; users can also configure replica member set priority
	Database access roles	Built-in roles; custom user-defined roles	Built-in roles; custom user-defined roles
Ed	On-demand access to MongoDB University	Included	Included
Security	Network Isolation	Dedicated VPC per customer group	Dedicated VPC per customer group
	Authentication and authorization	SCRAM Enabled by default Firewall with IP whitelist or VPC peering LDAP Integration Fine-grained data access controls using field-level redaction	SCRAM x.509 certificates
	Encryption for data in flight	SSL/TLS; enabled by default	SSL/TLS
	Encryption for data at rest	Included with encrypted storage volumes Encrypted Storage Engine is utilized when customer activates Encryption Key Management Bring your own key for encryption using cloud provider KMS	N/A: Requires third-party encryption at rest solution
	Auditing	DML operations DDL operations Audit trail filtering Role-based auditing Database logs are available for download; actions performed through the MongoDB Atlas UI are also logged and available	N/A: Requires third-party auditing solution
	Client Side Field Level Encryption (FLE)	Automatic Client Side FLE	Explicit Client Side FLE



	Differentiator	MongoDB Atlas	MongoDB Community Server
Additional Products	Atlas Data Lake	Query and analyze data across Amazon S3 and MongoDB Atlas in place and in its native format using the MongoDB Query Language (MQL)	N/A
	Atlas Online Archive	Automatically tier your data across fully managed databases and cloud object storage; combine and analyze data in place with federated queries	N/A
	Atlas Search	MongoDB's Atlas Search feature offers fine-grained text indexing to enable advanced search functionality without any additional management required	N/A
	Realm Mobile and Realm Sync	Available: For edge-to-cloud data synchronization	N/A
Commercial License	Licensing	N/A (because customer isn't running the database software)	N/A

MongoDB Atlas reference architectures on AWS

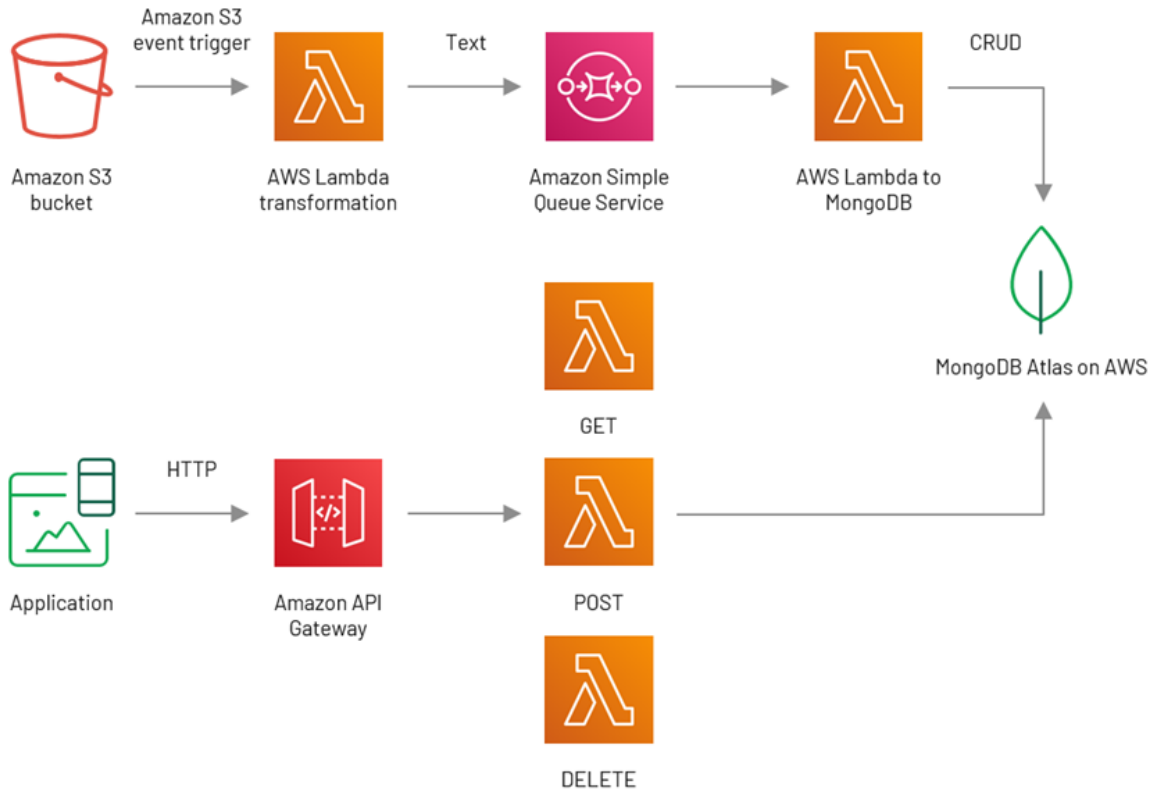
MongoDB Atlas systems are single-tenant, dedicated Amazon Elastic Compute Cloud (Amazon EC2) virtual servers that are created solely for an Atlas user. These virtual servers are isolated within their own virtual private cloud (VPC) and do not share logical data storage or processing with other users. AWS data centers are compliant with a number of security standards. For more information, see AWS Compliance.

The following sections describe three reference architectures for MongoDB Atlas on AWS: serverless, serverless with trigger events, and integrated with artificial intelligence (AI) / machine learning (ML).

Topics

MongoDB Atlas serverless architecture

The following serverless reference architecture uses AWS Lambda functions. It shows real-time extract, transform, load (ETL) operations from Amazon Simple Storage Service (Amazon S3) to MongoDB Atlas.

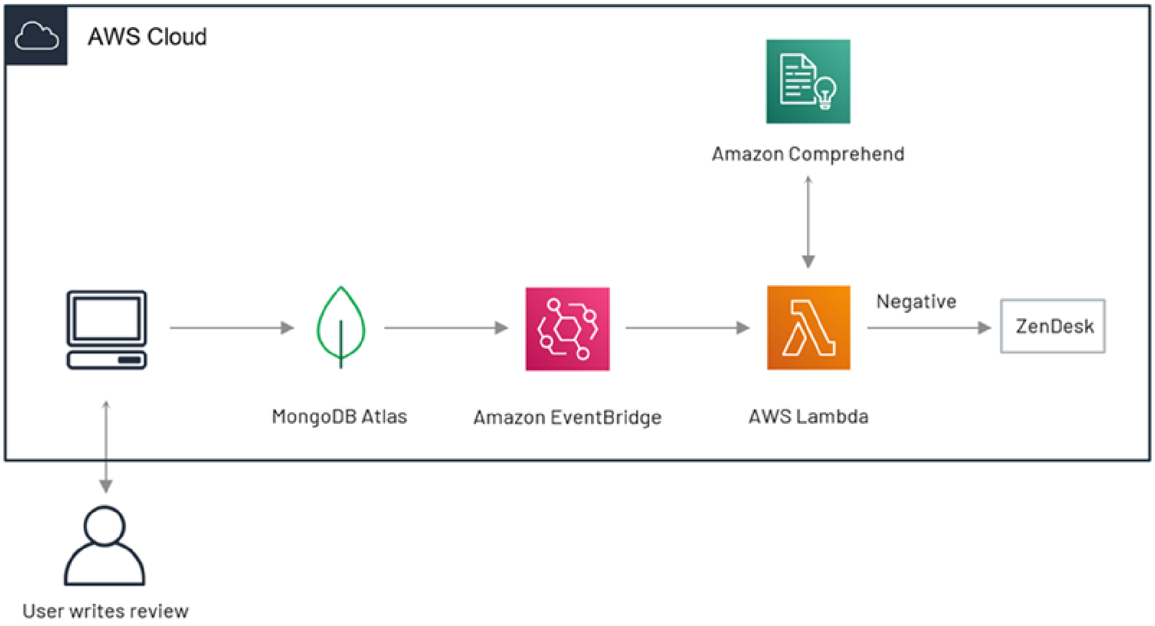


Key advantages of this architecture:

- The infrastructure requires minimal management.
- The stateless design provides radical scaling in response to workloads.
- The event-driven design helps with complex computations.
- The solution is cost-effective because you pay only for what you use.
- The architecture supports agility and availability, because you can go global in minutes.

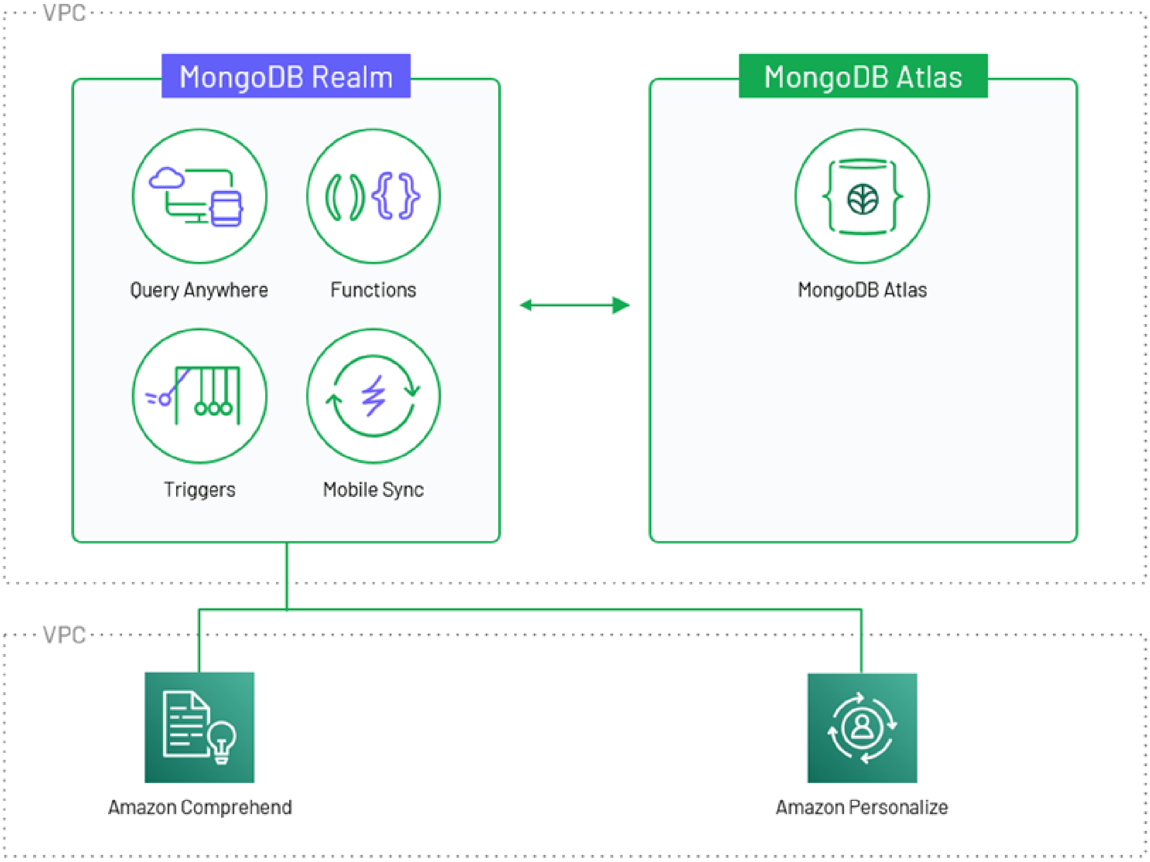
MongoDB Atlas serverless architecture with trigger events

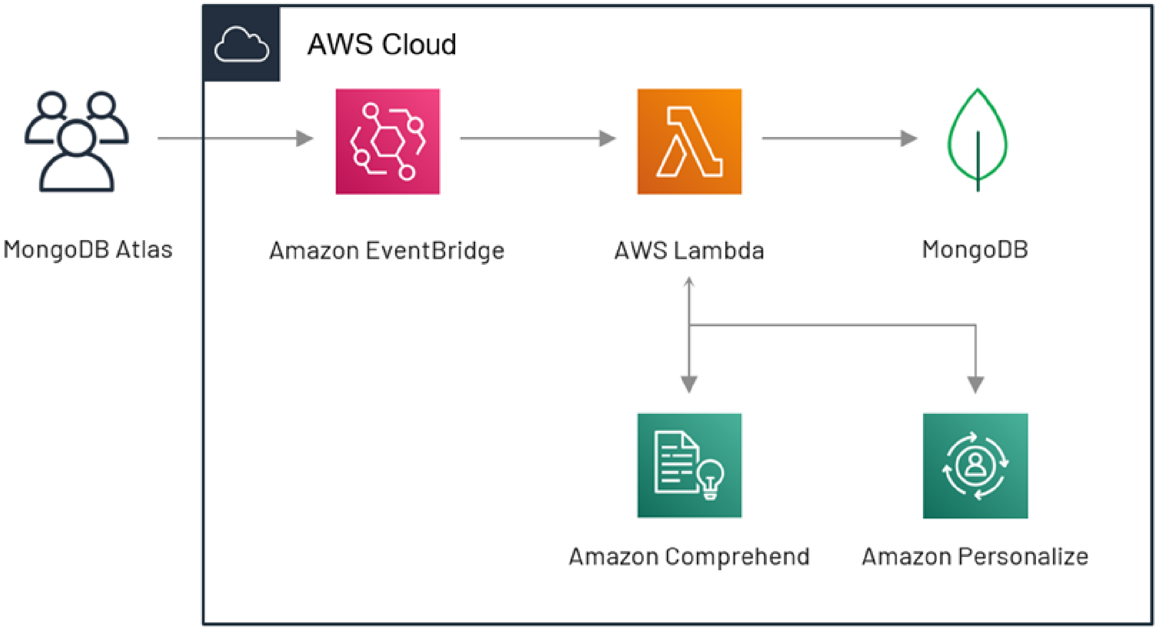
MongoDB offers an Amazon EventBridge partner event source that lets you send trigger events to an event bus instead of calling a function. You can configure any trigger to send events to EventBridge. To send trigger events to EventBridge, you only need an AWS account ID. The following diagram shows how to deploy an application from the AWS Serverless Application Repository that uses EventBridge to seamlessly integrate with and automate Zendesk, which is a customer service application. The application performs sentiment analysis on Zendesk support tickets with Amazon Comprehend. It then uses AWS Lambda and AWS Step Functions to categorize and orchestrate the escalation priority, based on configurable SLA wait times.



Integrating AI/ML with MongoDB Atlas and Realm

The following diagram shows how you can build a Single View in MongoDB Atlas and MongoDB Realm, and use Amazon Comprehend and Amazon Personalize to implement complex AI/ML requirements.



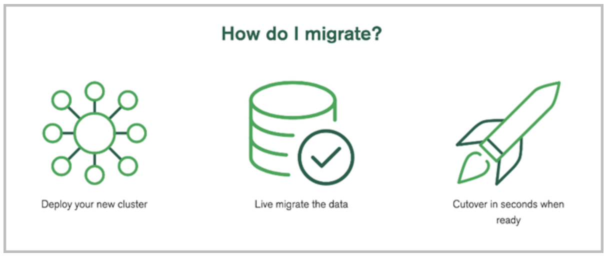


Amazon Personalize enables developers to build applications that provide real-time personalized recommendations. It reflects the vast experience that Amazon has in building personalization systems. Amazon Comprehend is a natural language processing (NLP) service that uses machine learning to find insights and relationships in text. Amazon Personalize and Amazon Comprehend do not require any machine learning experience.

High-level migration steps

After you deploy MongoDB Atlas on AWS, you can use the Atlas Live Migration Service to migrate data from your existing MongoDB environment—whether it's on a public cloud or in your data center, or you're using a third-party DBaaS provider—to MongoDB Atlas, with minimal impact to your applications.

The following diagram illustrates the data migration process.



The Atlas Live Migration Service makes an initial copy of the data from your source database, and watches for any changes so that it stays in sync until you're ready to cut over. During this process, your application can continue to read and write from your source database. When you're ready to cut over, you change the connection string in your application from your source database to your Atlas cluster, and the migration is complete.

The migration workflow consists of these steps:

Step 1. Prepare

- Assess your current MongoDB data size and workload.
- Determine the size of your Atlas cluster on AWS.
- Prepare your migration and cutover plans.

Step 2. Deploy

- Set up a new environment on AWS for the MongoDB Atlas cluster, based on the sizing you determined in step 1.
- Configure network access.
- Configure security for the destination cluster.

Step 3. Migrate

- Configure the source and destination in the Atlas Live Migration Service.
- Run the migration.
- Cut over to MongoDB Atlas on AWS.

For step-by-step migration instructions, see the pattern Migrate a self-hosted MongoDB environment to MongoDB Atlas on the AWS Cloud on the AWS Prescriptive Guidance website.

For more information about the Atlas Live Migration Service, see the MongoDB website.

For more information about the following migration scenarios, see the MongoDB documentation:

- Migrating from a self-managed MongoDB replica set to MongoDB Atlas
- Migrating from a self-managed replica set on AWS to MongoDB Atlas
- Migrating from mLab to MongoDB Atlas
- Additional migration guides

Integrating MongoDB with AWS services

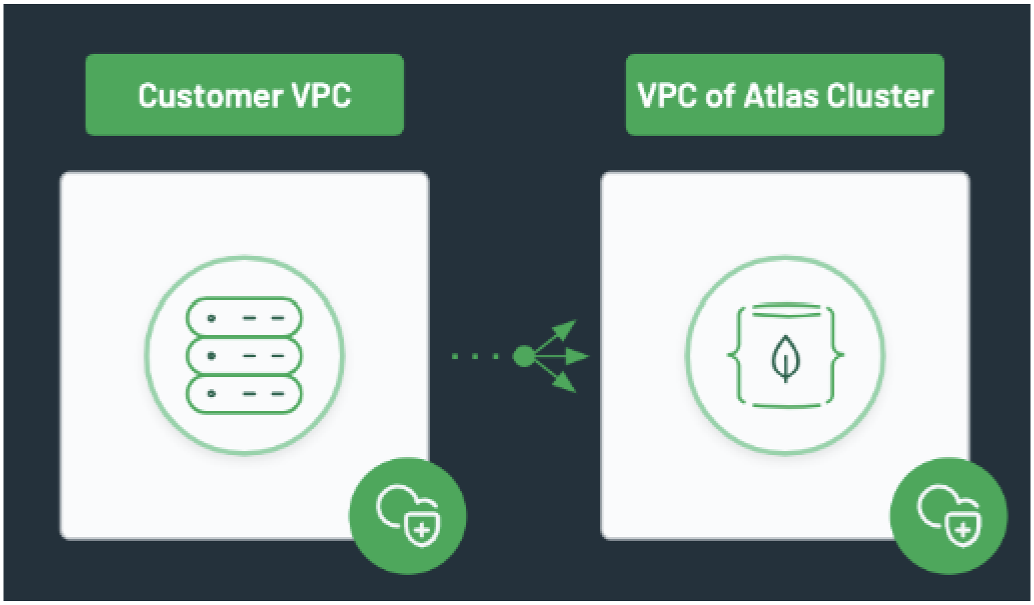
You can use AWS services to optimize your MongoDB Atlas environment. For example, you can:

- Configure connections between your applications and AWS services by using AWS PrivateLink.
- Implement Security Assertion Markup Language (SAML) authentication by using AWS IAM Identity Center (successor to AWS Single Sign-On).
- Deliver data to MongoDB Atlas from various data sources by integrating Atlas with Amazon Kinesis Data Firehose.
- Query and analyze data across Atlas and Amazon S3.
- Run code without provisioning or managing servers, by using AWS Lambda.

The following sections describe these integrations in more detail.

Configuring connections

You can use AWS PrivateLink to connect MongoDB Atlas to your AWS applications and ensure private connectivity between all your AWS services and accounts.



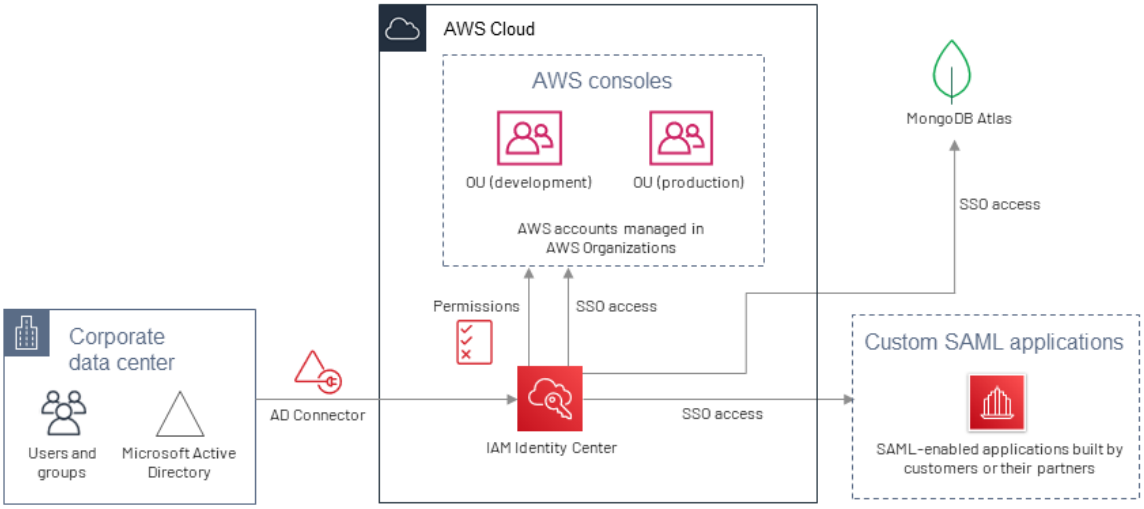
AWS PrivateLink provides these benefits:

- One-way connection – no extension of the perceived network trust boundary.
- Consolidated security controls across AWS applications and environments.

- Transitive connectivity from peered and AWS Direct Connect contexts – you can access Atlas from local environments through a virtual private network (VPN).

Implementing SAML authentication

Atlas supports SAML authentication through integration with IAM Identity Center and other identity management providers. SAML authentication is the open standard for exchanging identity and security information between applications and service providers. It lets customers centralize access management to Atlas by supporting single sign-on using corporate directory credentials. The following diagram shows how IAM Identity Center is used with Atlas.



Integrating data from multiple sources

Amazon Kinesis Data Firehose supports the MongoDB Cloud platform as one of its delivery destinations. This native integration between Kinesis Data Firehose and MongoDB Cloud provides a managed, secure, scalable, and fault-tolerant mechanism for delivering data to MongoDB Atlas.

You can stream your data through Amazon Kinesis Data Streams or push data directly to Kinesis Data Firehose and configure it to deliver data to MongoDB Atlas. You can also configure Kinesis Data Firehose to transform the data before delivering it to its destination. You don't have to write applications and manage resources to read data and push to MongoDB. It's all managed by AWS, which makes it easier to estimate costs based on your data volume. For more information, see Integrating the MongoDB Cloud with Amazon Kinesis Data Firehose on the AWS Big Data blog.

Querying and analyzing data

MongoDB Atlas Data Lake is a fully managed data lake as a service that enables you to natively query and analyze data across Amazon S3 and MongoDB Atlas. You can seamlessly combine and analyze your richly structured data stored in JSON, BSON, CSV, TSV, Avro, ORC, and Parquet formats without the cost and complexity of data movement and transformation. With this feature, you can query heterogeneous data stored in Amazon S3 and MongoDB Atlas in place and in its native format by using the MongoDB Query Language (MQL). For more information about using Atlas Data Lake with Amazon

S3, see MongoDB Atlas Data Lake Lets Developers Create Value from Rich Modern Data on the AWS Partner Network (APN) blog, and the MongoDB documentation.

Serverless development

AWS Lambda lets you run code without provisioning or managing servers. You pay only for the compute time you consume. For more information about using Atlas with Lambda, see Best practices for Connecting from AWS Lambda on the MongoDB website.

Additional resources

MongoDB documentation

- MongoDB Use Cases
- Billing
- Atlas Live Migration Service
- Getting started and migration guides
- Atlas Production Best Practices
- Best practices for Connecting from AWS Lambda

AWS resources

- Migrate a self-hosted MongoDB environment to MongoDB Atlas on the AWS Cloud
- MongoDB Atlas Data lake Lets Developers Create Value from Rich Modern Data (blog post)
- Integrating the MongoDB Cloud with Amazon Kinesis Data Firehose (blog post)

AWS Prescriptive Guidance glossary

AI and ML terms (p. 18) | Migration terms (p. 19) | Modernization terms (p. 23)

AI and ML terms

The following are commonly used terms in artificial intelligence (AI) and machine learning (ML)-related strategies, guides, and patterns provided by AWS Prescriptive Guidance. To suggest entries, please use the **Provide feedback** link at the end of the glossary.

binary classification

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

classification

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

data preprocessing

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

deep ensemble

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

deep learning

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

exploratory data analysis (EDA)

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

features

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

feature importance

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

feature transformation

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

interpretability

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

multiclass classification

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

regression

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

training

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

target variable

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

tuning

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

uncertainty

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

Migration terms

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

7 Rs

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

- Refactor/re-architect – Move an application and modify its architecture by taking full advantage of cloud-native features to improve agility, performance, and scalability. This typically involves porting the operating system and database. Example: Migrate your on-premises Oracle database to the Amazon Aurora PostgreSQL-Compatible Edition.

- Replatform (lift and reshape) – Move an application to the cloud, and introduce some level of optimization to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Amazon Relational Database Service (Amazon RDS) for Oracle in the AWS Cloud.
- Repurchase (drop and shop) – Switch to a different product, typically by moving from a traditional license to a SaaS model. Example: Migrate your customer relationship management (CRM) system to Salesforce.com.
- Rehost (lift and shift) – Move an application to the cloud without making any changes to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Oracle on an EC2 instance in the AWS Cloud.
- Relocate (hypervisor-level lift and shift) – Move infrastructure to the cloud without purchasing new hardware, rewriting applications, or modifying your existing operations. This migration scenario is specific to VMware Cloud on AWS, which supports virtual machine (VM) compatibility and workload portability between your on-premises environment and AWS. You can use the VMware Cloud Foundation technologies from your on-premises data centers when you migrate your infrastructure to VMware Cloud on AWS. Example: Relocate the hypervisor hosting your Oracle database to VMware Cloud on AWS.
- Retain (revisit) – Keep applications in your source environment. These might include applications that require major refactoring, and you want to postpone that work until a later time, and legacy applications that you want to retain, because there’s no business justification for migrating them.
- Retire – Decommission or remove applications that are no longer needed in your source environment.

application portfolio

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

artificial intelligence operations (AIOps)

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

AWS Cloud Adoption Framework (AWS CAF)

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

AWS landing zone

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

AWS Workload Qualification Framework (AWS WQF)

A tool that evaluates database migration workloads, recommends migration strategies, and provides work estimates. AWS WQF is included with AWS Schema

Conversion Tool (AWS SCT). It analyzes database schemas and code objects, application code, dependencies, and performance characteristics, and provides assessment reports.

business continuity planning (BCP)

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

Cloud Center of Excellence (CCoE)

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

cloud stages of adoption

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

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

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

configuration management database (CMDB)

A database that contains information about a company's hardware and software products, configurations, and inter-dependencies. You typically use data from a CMDB in the portfolio discovery and analysis stage of migration.

epic

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

heterogeneous database migration

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

homogeneous database migration

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

idle application

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

IT information library (ITIL)

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

IT service management (ITSM)

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

large migration

A migration of 300 or more servers.

Migration Acceleration Program (MAP)

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

Migration Portfolio Assessment (MPA)

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

Migration Readiness Assessment (MRA)

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

migration at scale

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

migration factory

Cross-functional teams that streamline the migration of workloads through automated, agile approaches. Migration factory teams typically include operations, business analysts and owners, migration engineers, developers, and DevOps professionals working in sprints. Between 20 and 50 percent of an enterprise application portfolio consists of repeated patterns that can be optimized by a factory approach. For more information, see the discussion of migration factories and the CloudEndure Migration Factory guide in this content set.

migration metadata

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

migration pattern

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

migration strategy

The approach used to migrate a workload to the AWS Cloud. For more information, see the 7 Rs (p. 19) entry in this glossary and see Mobilize your organization to accelerate large-scale migrations.

operational-level agreement (OLA)

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

operations integration (OI)

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

organizational change management (OCM)

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

playbook

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

portfolio assessment

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

responsible, accountable, consulted, informed (RACI) matrix

A matrix that defines and assigns roles and responsibilities in a project. For example, you can create a RACI to define security control ownership or to identify roles and responsibilities for specific tasks in a migration project.

runbook

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

service-level agreement (SLA)

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

task list

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

workstream

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

zombie application

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

Modernization terms

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

business capability

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

domain-driven design

An approach to developing a complex software system by connecting its components to evolving domains, or core business goals, that each component serves. This concept was introduced by Eric Evans in his book, *Domain-Driven Design: Tackling Complexity in the Heart of Software* (Boston: Addison-Wesley

Professional, 2003). For information about how you can use domain-driven design with the strangler fig pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

microservice

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

microservices architecture

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

modernization

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

modernization readiness assessment

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

monolithic applications (monoliths)

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

polyglot persistence

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

split-and-seed model

A pattern for scaling and accelerating modernization projects. As new features and product releases are defined, the core team splits up to create new product teams. This helps scale your organization's capabilities and services, improves developer productivity, and supports rapid innovation. For more information, see Phased approach to modernizing applications in the AWS Cloud.

strangler fig pattern

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

two-pizza team

A small DevOps team that you can feed with two pizzas. A two-pizza team size ensures the best possible opportunity for collaboration in software development. For more information, see the Two-pizza team section of the Introduction to DevOps on AWS whitepaper.

Document history

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

update-history-change	update-history-description	update-history-date
Initial publication (p. 26)	—	October 30, 2020