MongoDB on the AWS Cloud

Quick Start Reference Deployment

Karthik Krishnan

April 2015

Contents

About This Guide

This Quick Start reference deployment guide includes architectural considerations and configuration steps for deploying a MongoDB cluster on the Amazon Web Services (AWS) cloud. It discusses best practices for deploying MongoDB on AWS using services such as Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Virtual Private Cloud (Amazon VPC). It also provides links to automated AWS CloudFormation templates that you can leverage for your deployment or launch directly into your AWS account.

The guide is for IT infrastructure architects, administrators, and DevOps professionals who are planning to implement or extend their MongoDB workloads on the AWS cloud.

Quick Starts are automated reference deployments for key enterprise workloads on the AWS cloud. Each Quick Start launches, configures, and runs the AWS compute, network, storage, and other services required to deploy a specific workload on AWS, using AWS best practices for security and availability.

Overview

MongoDB on AWS

MongoDB is an open source, NoSQL database that provides support for JSON-styled, document-oriented storage systems. It supports a flexible data model that enables you to store data of any structure, and provides a rich set of features, including full index support, sharding, and replication.

AWS enables you to set up the infrastructure to support MongoDB deployment in a flexible, scalable, and cost-effective manner on the AWS cloud. This reference deployment will help you build a MongoDB cluster by automating configuration and deployment tasks.

This Quick Start supports a self-service deployment of the MongoDB cluster (version 2.6 or 3.0) on AWS.

Quick Links

The links in this section are for your convenience. Before you launch the Quick Start, please review the architecture, configuration, network security, and other considerations discussed in this guide.

View template

> **Note** These links set up a new Amazon Virtual Private Cloud (Amazon VPC). To deploy MongoDB into an existing Amazon VPC, see the Deployment section of this guide.

Launch Quick Start

The template includes default settings that you can customize by following the instructions in this guide.

Time to deploy: Approximately 15 minutes

Cost

This deployment launches MongoDB version 2.6 or 3.0 automatically into a configuration of your choice. You are responsible for the cost of the AWS services used while running this Quick Start reference deployment. There is no additional cost for using the Quick Start. The cost varies depending on the storage and compute configuration of the cluster you deploy.

The following table provides a cost estimate for some of the configurations as of the date of publication. The costs reflect per-month averages, and the storage data reflects the size of Amazon Elastic Block Store (Amazon EBS) volumes attached to each MongoDB node (excluding config servers, if applicable).

Note Because of space limitations, the table shows only a few of the available instance types and configuration options. AWS provides a number of other instance types and configurations that you can choose for your MongoDB deployment.

Instance Type	Cluster Shard Count	Replica Set	Storage (GiB)	Amazon EBS Volume	IOPS*	Estimated Cost/Month** ($)
t2.micro	0	1	10	gp2	30	0 (free tier***)
m3.large	0	1	256	gp2	768	121
m3.2xlarge	0	1	256	gp2	768	413
m3.2xlarge	0	3	500	gp2	1,500	1,496
m3.2xlarge	0	3	1,000	gp2	3,000	1,646
c3.4xlarge	0	3	2,000	gp2	6,000	2,561
c3.8xlarge	2	3	4,000	gp2	10,000	18,430
c3.8xlarge	2	3	4,000	io1	20,000	18,430
c3.8xlarge	2	3	8,000	io1	20,000	21,430

*To read more about GP2 and PIOPS volumes, see Amazon EBS Volume Types.

**Prices are subject to change. See the pricing pages for each AWS service you will be using or the AWS Simple Monthly Calculator for full details.

***Free tier: For details on eligibility, see the AWS Free Tier webpage.

AWS Services

The core AWS components used by this Quick Start include the following AWS services. (If you are new to AWS, see the Getting Started section of the AWS documentation.)

- Amazon EC2 – The Amazon Elastic Compute Cloud (Amazon EC2) service enables you to launch virtual machine instances with a variety of operating systems. You can choose from existing Amazon Machine Images (AMIs) or import your own virtual machine images.
- Amazon VPC – The Amazon Virtual Private Cloud (Amazon VPC) service lets you provision a private, isolated section of the AWS cloud where you can launch AWS services and other resources in a virtual network that you define. You have complete control over your virtual networking environment, including selection of your own IP address range, creation of subnets, and configuration of route tables and network gateways.
- Amazon Elastic Block Store (EBS) - Amazon Elastic Block Store (Amazon EBS) provides persistent block level storage volumes for use with Amazon EC2 instances in the AWS cloud. Each Amazon EBS volume is automatically replicated within its Availability Zone to protect you from component failure, offering high availability and durability. Amazon EBS volumes offer the consistent and low-latency performance needed to run your workloads.
- AWS CloudFormation – AWS CloudFormation gives you an easy way to create and manage a collection of related AWS resources, and provision and update them in an orderly and predictable way. You use a template to describe all the AWS resources (e.g., Amazon EC2 instances) that you want. You don't have to individually create and configure the resources or figure out dependencies; AWS CloudFormation handles all of that.
- IAM – AWS Identity and Access Management (IAM) enables you to securely control access to AWS services and resources for your users. With IAM, you can manage users, security credentials such as access keys, and permissions that control which AWS resources users can access, from a central location.

Architecture Overview

AWS CloudFormation provides an easy way to create and manage a collection of related AWS resources, provisioning and updating them in an orderly and predictable fashion.

The following AWS components are deployed and configured as part of this reference deployment:

- An Amazon VPC configured with public and private subnets.
- A network address translation (NAT) instance deployed into the public subnet and configured with an Elastic IP address (EIP) for outbound Internet connectivity and inbound SSH (Secure Shell) access. The NAT instance is used for Internet access if any Amazon EC2 instances are launched within the private network.
- An AWS Identity and Access Management (IAM) instance role with fine-grained permissions for access to AWS services necessary for the deployment process.
- Security groups for each instance or function to restrict access to only necessary protocols and ports.
- A fully customizable MongoDB cluster, including replica sets, shards, and config servers along with customizable Amazon EBS storage. The Quick Start launches all the MongoDB-related nodes in the private subnet, so the nodes are accessed by using SSH to connect to the NAT instance. In addition, the Quick Start sets up security groups to restrict access among nodes. The console supports additional customizations of security groups, if necessary.

MongoDB Constructs

Here are some of the building blocks that are used in this reference deployment.

Replica set. Refers to a group of `mongod` instances that hold the same data. The purpose of replication is to ensure high availability, in case one of the servers goes down. This reference deployment supports one or three replica sets. In the case of three replica sets, the reference deployment launches three servers in three different Availability Zones (if the region supports it). In production clusters, we recommend using three replica sets (*Primary*, *Secondary0*, *Secondary1*).

All clients typically interact with the primary node for read and write operations. It is possible to choose a secondary node as a preference during read operations, but write operations always go to the primary node and get replicated asynchronously in the secondary nodes. If you choose a secondary node for read operations, watch out for stale data, because the secondary node may not be in sync with the primary node. For more information about how read operations are routed in a replica set, see the MongoDB documentation.

In a development environment, you can start with a single replica set and move to three replica sets during production. Figure 1 shows the MongoDB reference deployment with a replication factor of 3.

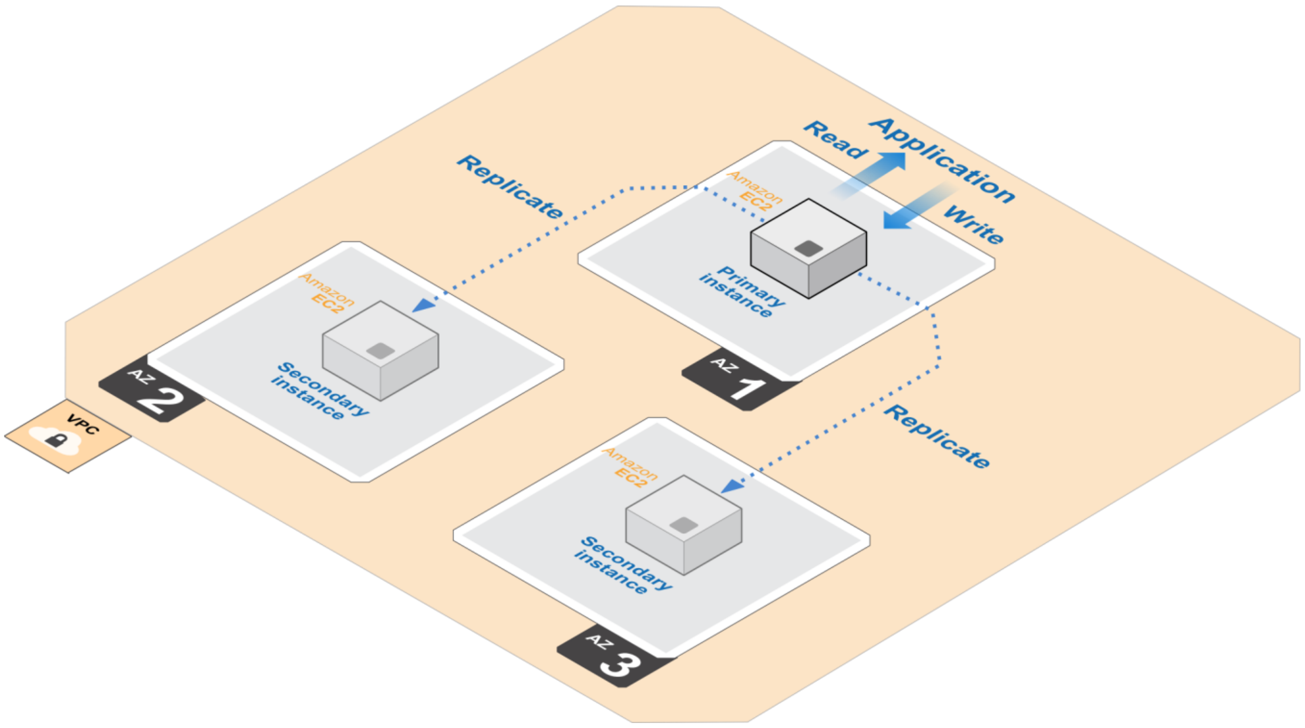


Figure 1: MongoDB Cluster on AWS with Three Replica Sets

When a primary instance fails, one of the secondary instances from another Availability Zone will become the new primary node, thereby guaranteeing automatic failover.

Sharding. Refers to distribution of data across multiple nodes. Storing distinct data across multiple nodes provides horizontal scalability for read and write performance. When you have a large data set, a single node could be bottlenecked by CPU or I/O performance. Sharding resolves this bottleneck by reducing the number of operations each shard node handles, and improves overall cluster performance. This Quick Start provides support for up to three shards. A shard count of 0 results in a simple replicated cluster.

Sharded cluster. MongoDB supports sharding via sharded clusters. Each shard can be a replica set. This Quick Start supports one and three replica sets, so each shard can be housed in one or three Availability Zones. Sharded clusters provide high availability of individual shards and are recommended for production.

Query routers. These are instances that run the *mongos* process and interface with clients. The *mongos* process is responsible for handling queries from the application layer and determining the location of the query data in the sharded cluster. This reference deployment runs one query router on primary nodes. Additional routers can be started manually, if needed.

Config servers. This reference deployment launches three config servers in three different Availability Zones in a sharded cluster. These are lightweight instances that store the cluster's metadata, so a small instance type is sufficient for a config server.

Figure 2 shows a two-way sharded and three-way replicated cluster.

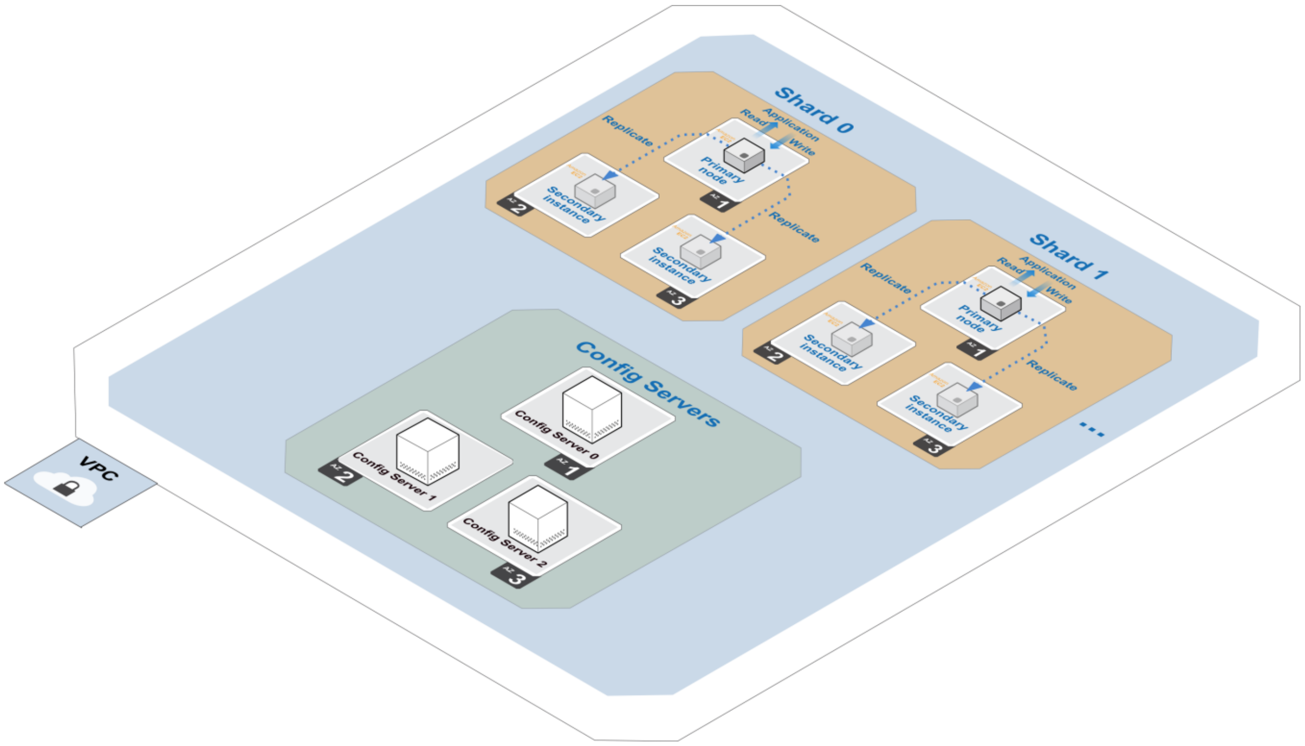


Figure 2: MongoDB cluster on AWS with Three Replica Sets and Two-Way Sharding

Performance Considerations

The reference implementation offers various compute and storage choices. The following table shows some of the compute choices to consider.

Instance Type	vCPU	Memory (GiB)	Workload Type
c3.4xlarge	16	55	Compute-optimized
c3.8xlarge	32	60	Compute-optimized
c4.8xlarge	36	60	Compute-optimized
r3.4xlarge	16	122	Memory-optimized

Instance Type	vCPU	Memory (GiB)	Workload Type
r3.2xlarge	8	61	Memory-optimized
r3.8xlarge	32	244	Memory-optimized

As a general guideline, consider growing instances horizontally instead of vertically. Horizontal scaling overcomes the limitations of single nodes and avoids single points of failure, and can potentially increase the overall throughput of your cluster.

For storage, depending on your database requirement, you may choose to change the storage volume to be attached to each node. Amazon EBS provides three volume types: General Purpose (SSD) volumes, Provisioned IOPS (SSD) volumes, and Magnetic volumes. These differ in performance characteristics and cost, so you can choose the right storage performance and price depending on the needs of your application. All Amazon EBS volume types offer the same durable snapshot capabilities and are designed for 99.999% availability. This reference deployment supports General Purpose and Provisioned IOPS storage volumes.

The following table shows some of the performance characteristics of each storage type. Depending on your performance requirements, you may want to benchmark your application before deciding on the storage type and Amazon EBS Provisioned IOPS capacity (if chosen).

Volume Type	General Purpose (SSD)	Provisioned IOPS (SSD)
Storage media	SSD-backed	SSD-backed
Maximum volume size	16 TiB	16 TiB
Maximum IOPS/volume	10,000	20,000

MongoDB Cluster in a Private Subnet

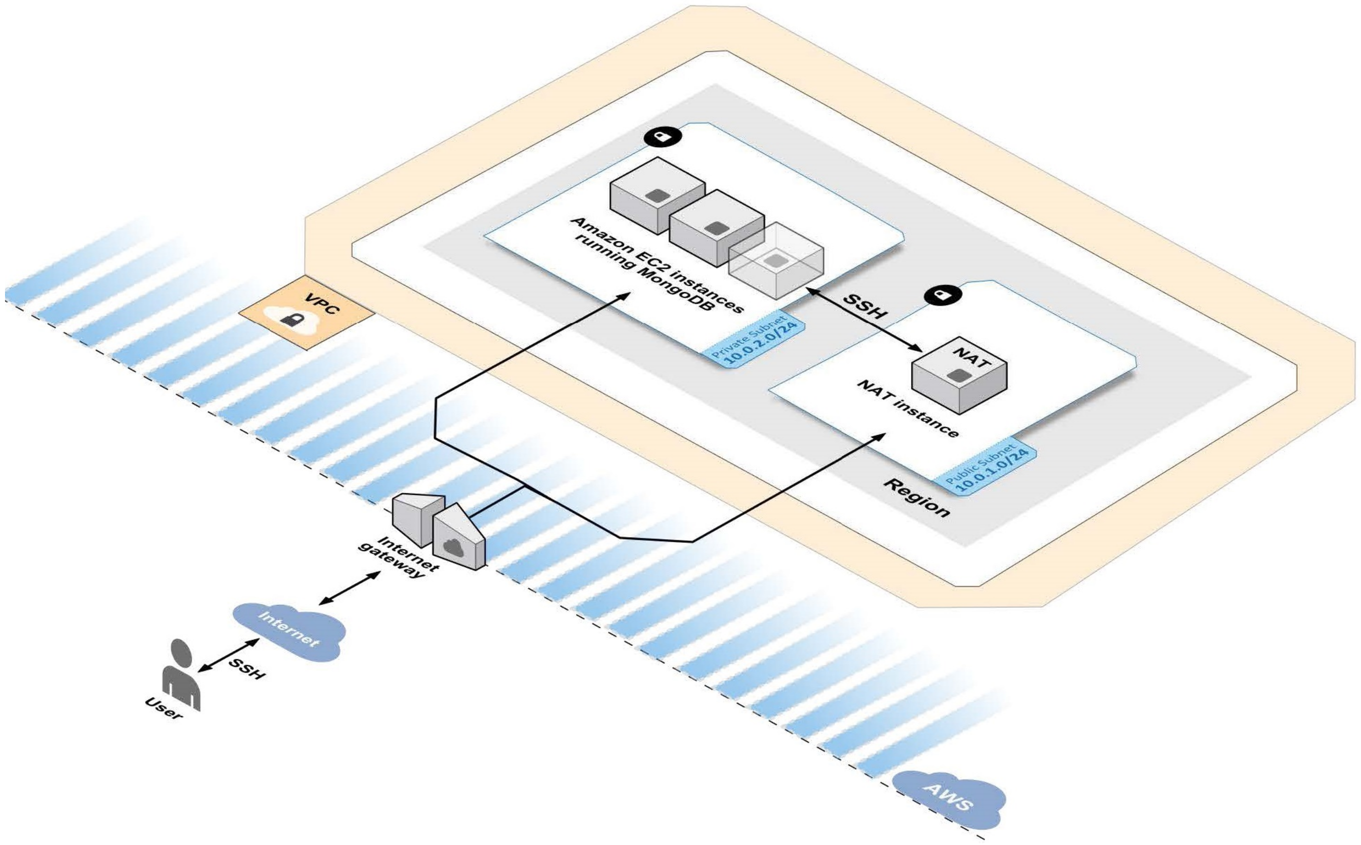

Amazon EC2 instances running MongoDB
Private Subnet 10.0.2.0/24
SSH
NAT
NAT instance
Public Subnet 10.0.1.0/24
Region
VPC
Internet gateway
Internet
SSH
User
AWS

Figure 3: MongoDB Subnet Topology

The MongoDB subnet cluster topology includes a NAT instance with an associated Elastic IP Address (EIP) and a security group that allows SSH access to the NAT instance. All other MongoDB-related Amazon EC2 instances are created within the private subnet. In this topology, the Amazon EC2 instances within the MongoDB cluster do not have direct access to the Internet or to other AWS services. Instead, their access is routed through the NAT instance that resides in the public subnet. For more information, see the article High Availability for Amazon VPC NAT Instances.

Deployment

You can deploy MongoDB easily on the flexible AWS platform. This guide serves as a reference for customers who want to set up a fully customizable MongoDB cluster on demand. Building a scalable, on-demand infrastructure on AWS provides a cost-effective solution for handling large-scale compute and storage requirements.

The flexible AWS architecture allows you to choose the most appropriate network, compute, and storage infrastructure for your environment. At the top level, you can deploy the Quick Start into an existing Amazon VPC or create a new Amazon VPC for the MongoDB cluster.

What We'll Cover

The procedure for deploying MongoDB on AWS consists of the following steps. For detailed instructions, follow the links for each step.

Step 1. Prepare an AWS account

- Sign up for an AWS account, if you don't already have one.
- Choose the region where you want to deploy the stack on AWS.
- Create a key pair in the region.
- Review account limits for Amazon EC2 instances and EBS volumes, and request a limit increase, if needed.

Choose step 2(a) **or** step 2(b) depending on whether you have an existing Amazon VPC.

Step 2 (option a, for a new Amazon VPC). Launch the Quick Start into your AWS account

When you launch the Quick Start to deploy MongoDB into a new Amazon VPC, the AWS CloudFormation template automates the following steps:

- Sets up the Amazon VPC.
- Creates various network resources needed during MongoDB deployment, including private and public subnets within an Amazon VPC, a NAT instance, security groups, and an IAM role.
- Sets up Amazon EBS to store MongoDB. You can customize Amazon EBS depending on your needs. The CloudFormation template provides options for configuring both storage type and storage size. You can also choose Provisioned IOPS for the EBS volumes.
- Provides options for customizing the MongoDB configuration before launch. You can choose the MongoDB version number (2.6 or 3.0), number of replica sets (one or three), shard count (0, 1, 2, or 3), and microshards per instance.
- Fully automates security settings to provide minimal privileges to each MongoDB instance, and firewall options to open up access between MongoDB nodes.

Step 2 (option b, for an existing Amazon VPC). Launch the Quick Start into your AWS account

This Quick Start provides a separate template for deploying MongoDB into an existing Amazon VPC. If you have already constructed an Amazon VPC within AWS and want to deploy MongoDB inside your Amazon VPC, use this option. The template automates all the steps in step 2(a) except for the creation of the Amazon VPC.

Step 3. Connect to MongoDB nodes

You use SSH to connect to MongoDB nodes via the NAT instance, because the nodes are in a private subnet.

Step 1. Prepare an AWS Account

1. If you don't already have an AWS account, create one at http://aws.amazon.com by following the on-screen instructions. Part of the sign-up process involves receiving a phone call and entering a PIN using the phone keypad.
2. Use the region selector in the navigation bar to choose the Amazon EC2 region where you want to deploy the MongoDB cluster on AWS.

   Amazon EC2 locations are composed of *regions* and *Availability Zones*. Regions are dispersed and located in separate geographic areas. All Amazon EC2 instances (except R3 instances) can be launched in any of the regions. R3 instances are currently available in all AWS regions except GovCloud (US), China (Beijing), and South America (São Paulo).

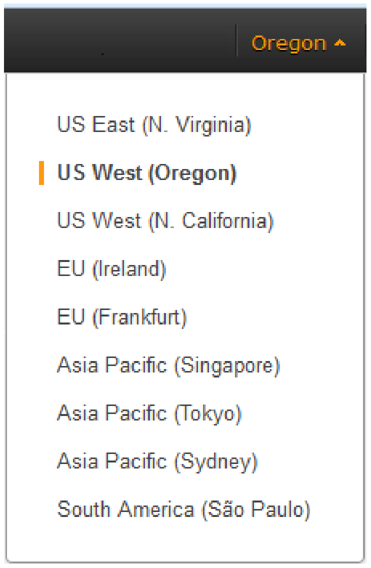


Figure 4: Choosing an Amazon EC2 Region

Tip Consider choosing a region closest to your data center or corporate network to reduce network latency between systems running on AWS and the systems and users on your corporate network.

3. Create a key pair in your preferred region. To do this, in the navigation pane of the Amazon EC2 console, choose **Key Pairs**, **Create Key Pair**, type a name, and then choose **Create**.

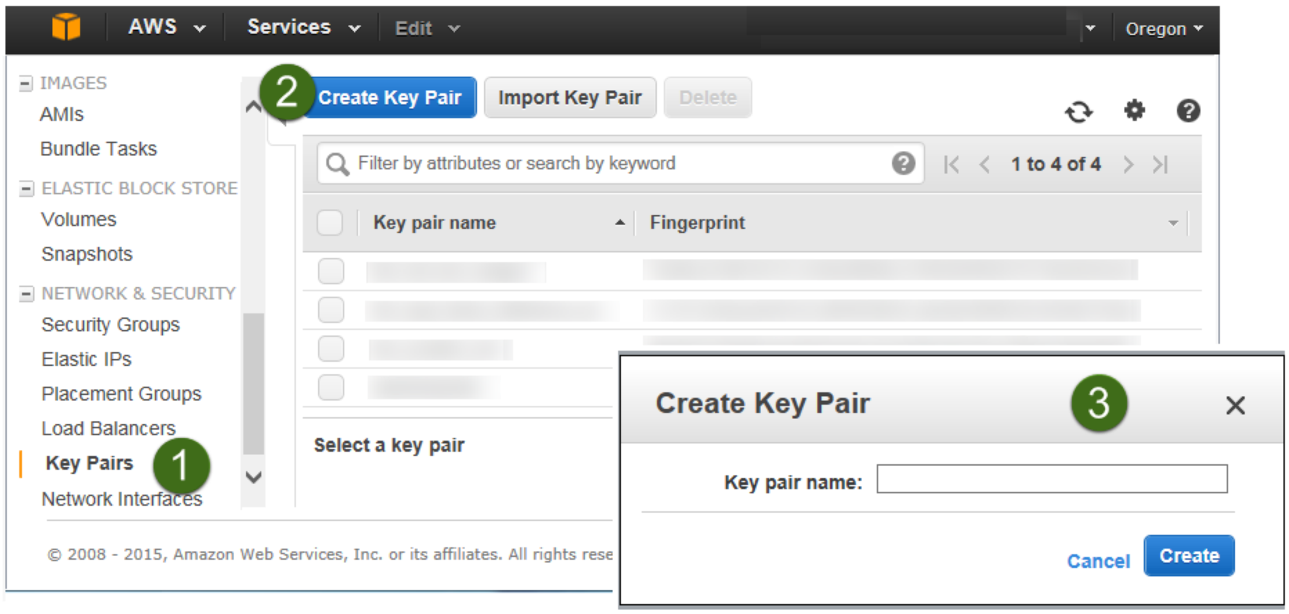
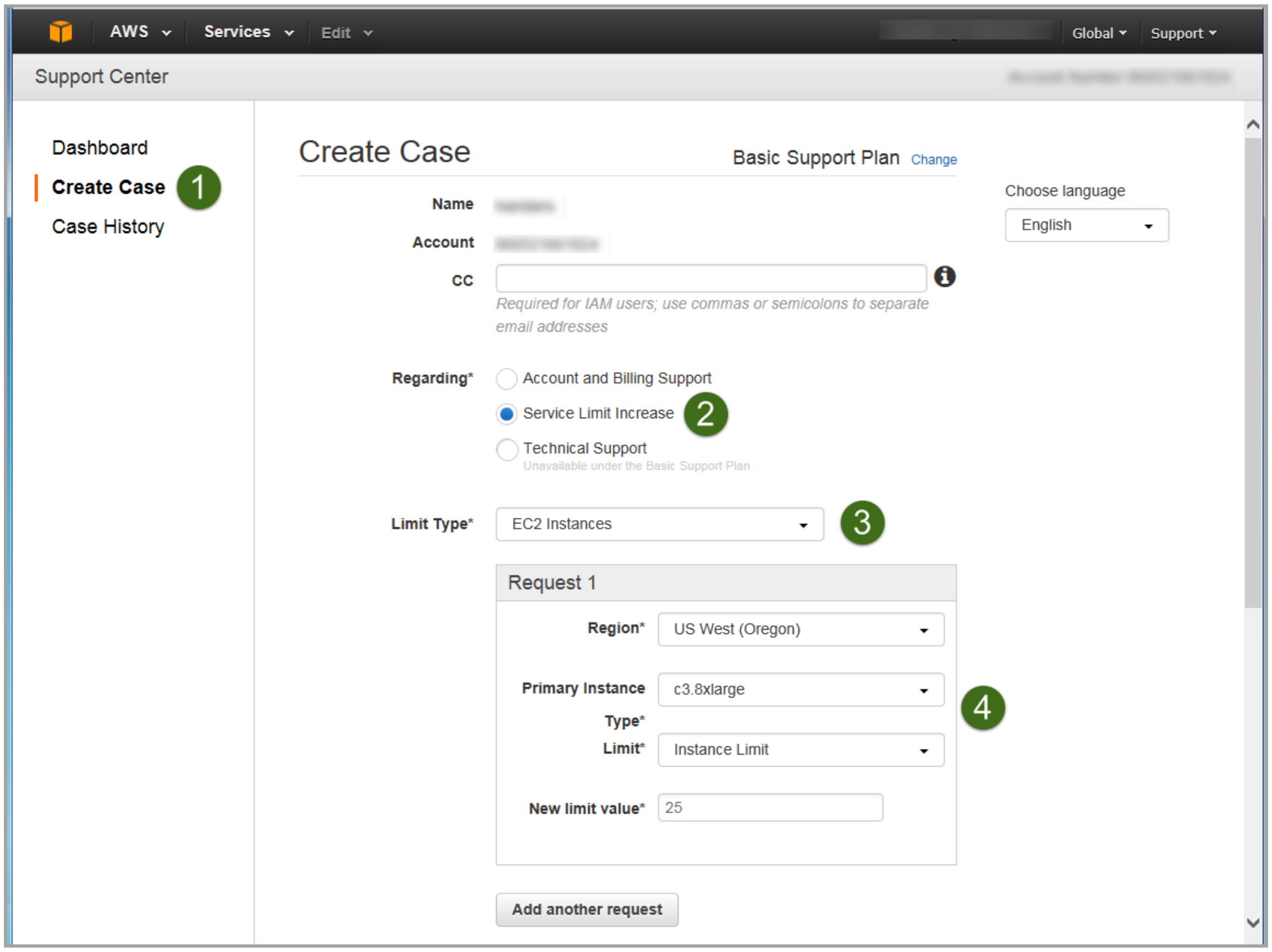


Figure 5: Creating a Key Pair

Amazon EC2 uses public-key cryptography to encrypt and decrypt login information. To be able to log in to your instances, you must create a key pair. On Linux, we use the key pair to authenticate SSH login.

4. If necessary, request a service limit increase for the Amazon EC2 instance types that you intend to deploy.

   Depending on the instance type, the default limit for the number of instances that can be run ranges from 2 to 20 (see the Amazon EC2 FAQs page). If you have existing deployments that also use this instance type, or if you plan to exceed this default with this reference deployment, you will need to request an Amazon EC2 instance service limit increase. It might take a few days for the new service limit to become effective. For more information, see Amazon EC2 Service Limits in the AWS documentation.



Support Center [redacted]

Create Case

Basic Support Plan Change

Choose language: English

Name: [redacted]

Account: [redacted]

CC:

Required for IAM users; use commas or semicolons to separate email addresses

Regarding*: Account and Billing Support [✓ Service Limit Increase] Technical Support (Unavailable under the Basic Support Plan)

Limit Type*: EC2 Instances

Request 1

Region*: US West (Oregon)

Primary Instance Type*: c3.8xlarge

Limit*: Instance Limit

New limit value*: 25

Figure 6: Requesting a Service Limit Increase

5. If necessary, request a limit increase for the EBS volumes that you can use. Choose **EBS** for the limit type, and complete the fields on the limit request form, similar to step 4.

Step 2(a). Launch the Quick Start into Your AWS Account (New Amazon VPC)

In this step, you will launch an AWS CloudFormation template that automates the following:

- Configures the Amazon VPC that provides the base AWS network infrastructure for your MongoDB deployment.
- Creates the network resources needed for MongoDB deployment, including public and private subnets within the Amazon VPC, a NAT instance launched within the public subnet, security groups, and an IAM role.

- Launches Amazon EC2 instances running Amazon Linux depending on the replica set and sharding options you choose. In addition, Amazon EBS storage volumes are automatically attached to those instances.
- Downloads the selected version of MongoDB along with the necessary scripts and configuration files.

All of these steps are fully automated by the AWS CloudFormation template. You can change some of the configuration settings, but the only mandatory input expected by the template is *KeyName*, which is the name of the key pair you created in step 1, when you set up your AWS account.

1. Launch the AWS CloudFormation template into your AWS account.

   Launch (for new VPC)

   The template is launched in the US East (N. Virginia) region by default. You can change the region by using the region selector in the navigation bar.

   This stack takes approximately 15 minutes to create.

   > **Note** You are responsible for the cost of the AWS services used while running this Quick Start reference deployment. There is no additional cost for using this Quick Start. See the Cost section for cost estimates for different instance types and storage configurations. Prices are subject to change. For full details, see the pricing pages for each AWS service you will be using in this Quick Start.

   You can also download the template to use it as a starting point for your own implementation.

6. On the **Select Template** page, keep the default settings for the stack name and template source, and then choose **Next**.

7. On the **Specify Parameters** page, review the parameters for the template. Provide a value for the *KeyName* parameter. You can also customize the following additional parameters. The AWS CloudFormation template uses these settings to generate a cluster configuration file. When you're done, choose **Next**.

Network Settings

Parameter	Default	Description
VPCCIDR	10.0.0.0/16	CIDR block for the Amazon VPC you are creating.
Public Subnet	10.0.1.0/24	CIDR block for the public DMZ subnet located in the new Amazon VPC.

Parameter	Default	Description
PrimaryReplicaSubnet	10.0.2.0/24	CIDR block for the private subnet where the primary replica will be deployed.
SecondaryReplicaSubnet0	10.0.3.0/24	CIDR block for the private subnet of secondary replica set 0 (applicable only when *ClusterReplicaSetCount* >= 2).
SecondaryReplicaSubnet1	10.0.4.0/24	CIDR block for the private subnet of secondary replica set 1 (applicable only when *ClusterReplicaSetCount* >= 2).
RemoteAccessCIDR	0.0.0.0/0	IP CIDR from which you are likely to connect to the NAT instance by using SSH.
KeyName	*Requires input*	An existing public/private key pair, which allows you to connect securely to your instance after it launches. This is the key pair you created in step 1, when you prepared your AWS account.

Compute Settings

Parameter	Default	Description
NATInstanceType	m1.small	Amazon EC2 instance type for the NAT instances.
NodeInstanceType	m3.medium	MongoDB node instance type.
ConfigServerInstanceType	t2.micro	Amazon EC2 instance type for the config servers.

Storage Settings

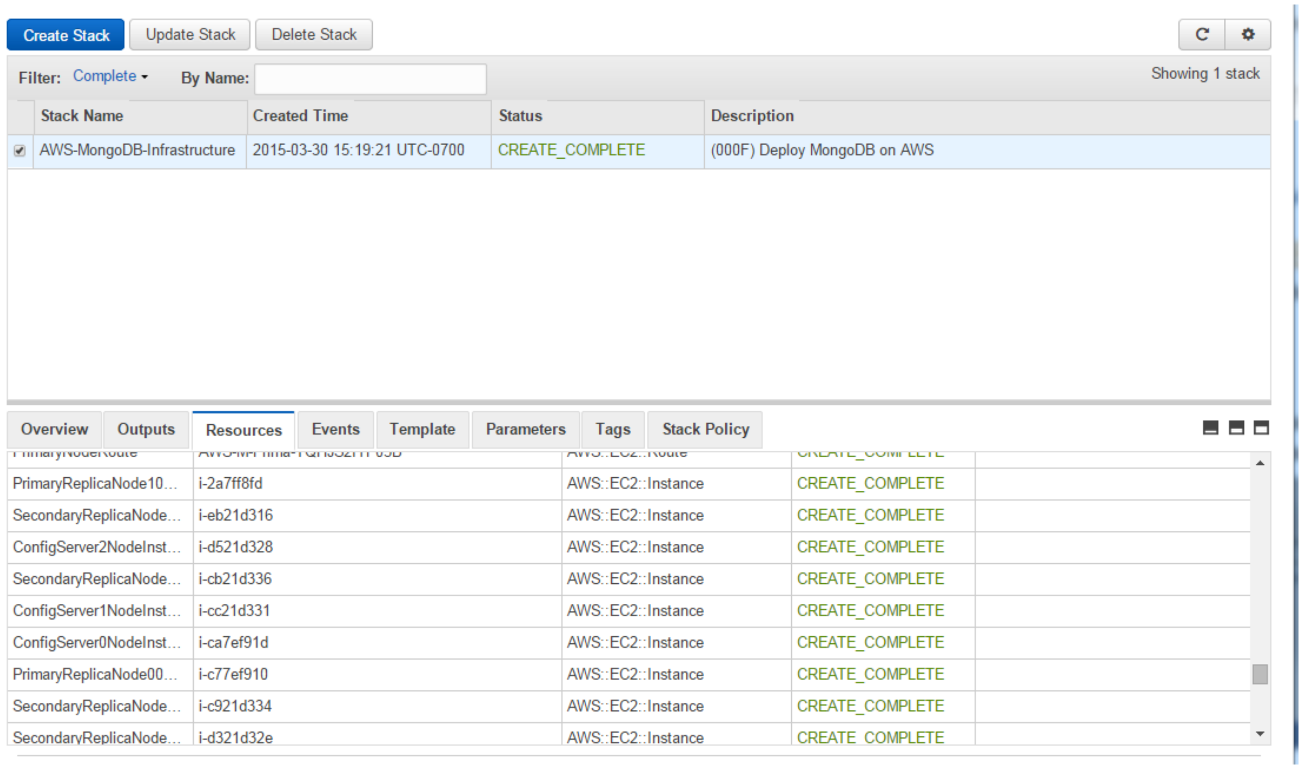
Parameter	Default	Description
VolumeSize	400	Amazon EBS volume size (data) to be attached to the node, in GiBs.
VolumeType	gp2	Amazon EBS volume type (`gp2` or `io1`).
Iops	100	IOPS of the Amazon EBS volume when the `io1` volume type is chosen. Otherwise, this setting is ignored.

MongoDB Options

Parameter	Default	Description
ClusterReplicaSetCount	1	Number of replica sets. Choose 1 or 3.
ClusterShardCount	0	Number of shards. Choose 0, 1, 2, or 3. 0==no sharding. Set to > 1 for sharding.
ShardsPerNode	1	Number of microshards per node.
MongoDBVersion	3.0	MongoDB version (2.6 or 3.0).

8. On the **Options** page, choose **Next**.
9. On the **Review** page, review and confirm the settings, and then choose **Create** to deploy the stack.

When **CREATE_COMPLETE** is displayed in the status field, as shown in Figure 7, the MongoDB cluster is ready.



	Stack Name	Created Time	Status	Description
✓	AWS-MongoDB-Infrastructure	2015-03-30 15:19:21 UTC-0700	CREATE_COMPLETE	(000F) Deploy MongoDB on AWS

PrimaryNodeRoute	[illegible]	AWS::EC2::Route	CREATE_COMPLETE	
PrimaryReplicaNode10...	i-2a7ff8fd	AWS::EC2::Instance	CREATE_COMPLETE	
SecondaryReplicaNode...	i-eb21d316	AWS::EC2::Instance	CREATE_COMPLETE	
ConfigServer2NodeInst...	i-d521d328	AWS::EC2::Instance	CREATE_COMPLETE	
SecondaryReplicaNode...	i-cb21d336	AWS::EC2::Instance	CREATE_COMPLETE	
ConfigServer1NodeInst...	i-cc21d331	AWS::EC2::Instance	CREATE_COMPLETE	
ConfigServer0NodeInst...	i-ca7ef91d	AWS::EC2::Instance	CREATE_COMPLETE	
PrimaryReplicaNode00...	i-c77ef910	AWS::EC2::Instance	CREATE_COMPLETE	
SecondaryReplicaNode...	i-c921d334	AWS::EC2::Instance	CREATE_COMPLETE	
SecondaryReplicaNode...	i-d321d32e	AWS::EC2::Instance	CREATE_COMPLETE	

Figure 7: Successful Creation of the MongoDB Cluster

Step 2(b). Launch the Quick Start into Your AWS Account (Existing Amazon VPC)

If you have an existing Amazon VPC, you can still use the Quick Start to build a MongoDB cluster within it. The deployment steps are same as in step 2(a) except for the network parameters. Instead of CIDR ranges, you input the IDs of the corresponding subnets, as shown in the following table. All other options remain the same.

Launch
(for existing VPC)

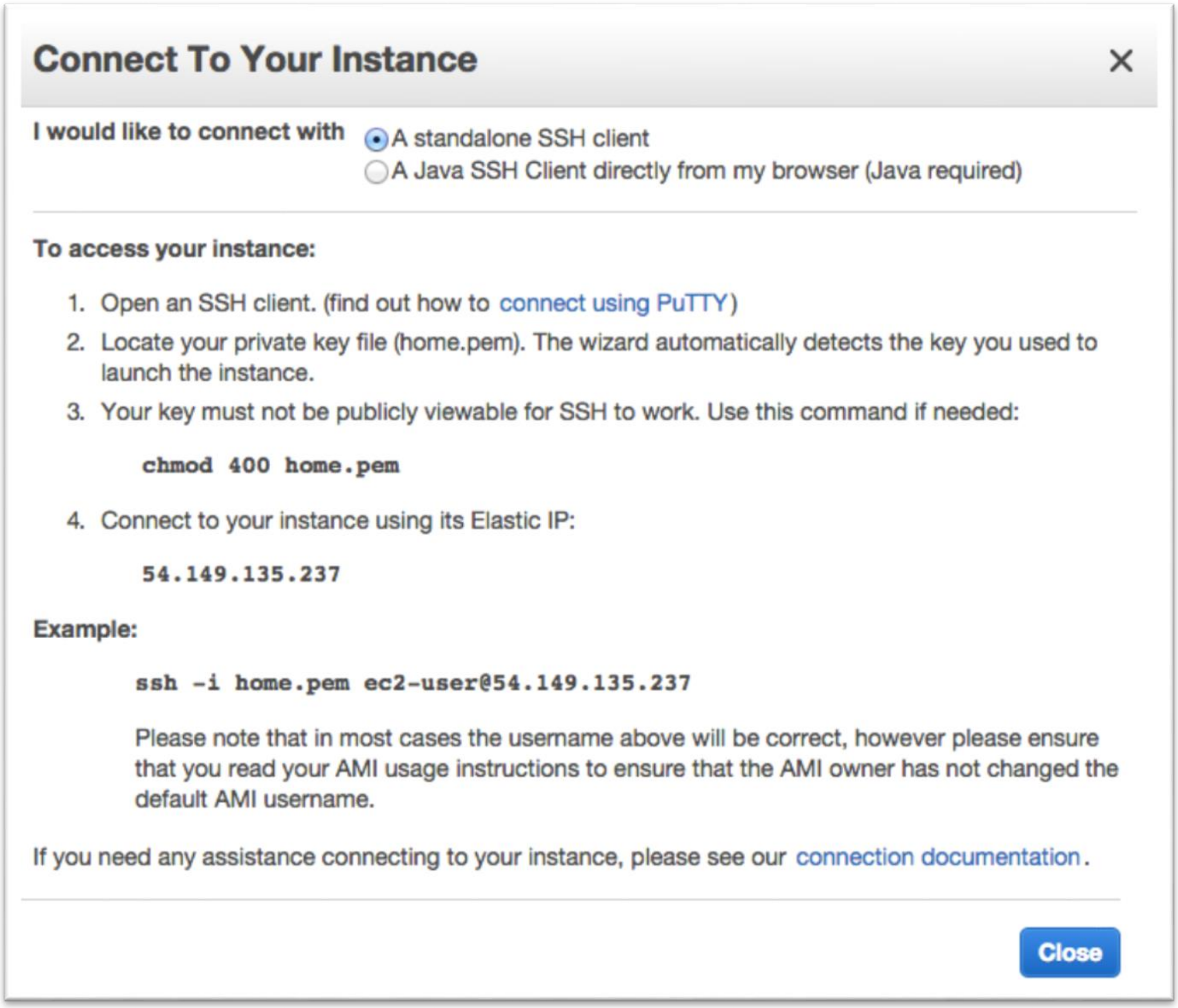
Network Settings

Parameter	Default	Description
PrimaryNodeSubnet	subnet-xxxx (*requires input*)	ID of an existing subnet in your Amazon VPC where you want to deploy the primary node.
Secondary0NodeSubnet	subnet-xxxx (*requires input*)	ID of an existing subnet in your Amazon VPC where you want to deploy secondary replica set 0 (if applicable).
Secondary1NodeSubnet	subnet-xxxx (*requires input*)	ID of an existing subnet in your Amazon VPC where you want to deploy secondary replica set 1 (if applicable).

You can also download the template to use it as a starting point for your own implementation.

Step 3. Connect to MongoDB Nodes

Once the AWS CloudFormation template has successfully created the stack, all the MongoDB nodes will be running with the software installed in your AWS account. To connect to any of the MongoDB nodes, use SSH to connect to the NAT instance. In the Amazon EC2 console, choose the instance, and then choose **Connect**.



Connect To Your Instance

I would like to connect with [✓ A standalone SSH client] A Java SSH Client directly from my browser (Java required)

To access your instance:

1. Open an SSH client. (find out how to connect using PuTTY)
2. Locate your private key file (home.pem). The wizard automatically detects the key you used to launch the instance.
3. Your key must not be publicly viewable for SSH to work. Use this command if needed:

```
chmod 400 home.pem
```

4. Connect to your instance using its Elastic IP:

```
54.149.135.237
```

Example:

```
ssh -i home.pem ec2-user@54.149.135.237
```

Please note that in most cases the username above will be correct, however please ensure that you read your AMI usage instructions to ensure that the AMI owner has not changed the default AMI username.

If you need any assistance connecting to your instance, please see our connection documentation.

Figure 8: Connecting to a MongoDB Node

Once you connect to the NAT instance by using SSH, you can connect to any of the MongoDB nodes in a similar fashion (choose the node, and then choose **Connect** to find the SSH command).

Important You need the private key (.pem) file to connect to MongoDB nodes. Copy the private key (.pem) file into the NAT instance; for example:

```
scp -i mykey.pem mykey.pem ec2-user@NAT-public-ip:/home/ec2-user/mykey.pem
```

Note that all the MongoDB nodes are launched with an IAM role that grants them privileges to create and delete Amazon DynamoDB tables, to access Amazon Simple Storage Service (Amazon S3), to create and delete Amazon EC2 instances, and so on. You can modify the policy by using the IAM console. For details about the benefits of IAM roles, see Using IAM Roles to Delegate Permissions to Applications that Run on Amazon EC2 in the AWS documentation.

Testing MongoDB

After the AWS CloudFormation template has completed successfully, the system will have a *mongos* instance running on each of the primary replica set nodes. To validate the system and verify the configuration, follow these steps:

1. Use SSH to log in to one of the primary instances created by the Quick Start template.
2. Execute the following commands from the terminal:

   ```
   mongo
   sh.printShardingStatus()
   ```

3. Verify that the mongo shell connects to the local host on the default TCP port (27017), and that the output reflects the configuration that you specified for the Quick Start template.

For additional information on testing the MongoDB server, see the MongoDB documentation.

Backing Up Your Data

For backup, we recommend using Amazon S3 to keep a copy of your MongoDB data. Amazon S3 stores data objects redundantly on multiple devices across multiple facilities and allows concurrent read or write access to these data objects by many separate clients or application threads. You can use the redundant data stored in Amazon S3 to recover quickly and reliably from instance or application failures.

For other backup strategies, see the MongoDB documentation.

Security

The AWS cloud provides a scalable, highly reliable platform that helps customers deploy applications and data quickly and securely.

When you build systems on the AWS infrastructure, security responsibilities are shared between you and AWS. This shared model can reduce your operational burden as AWS operates, manages, and controls the components from the host operating system and virtualization layer down to the physical security of the facilities in which the services operate. In turn, you assume responsibility and management of the guest operating system (including updates and security patches), other associated applications, as well as the configuration of the AWS-provided security group firewall. For more information about security on AWS, visit the AWS Security Center.

AWS Identity and Access Management (IAM)

This solution leverages an IAM role with least privileged access. It is not necessary or recommended to store SSH keys, secret keys, or access keys on the provisioned instances.

OS Security

The root user on cluster nodes can be accessed only by using the SSH key specified during the deployment process. AWS doesn't store these SSH keys, so if you lose your SSH key you can lose access to these instances.

Operating system patches are your responsibility and should be performed on a periodic basis.

Security Groups

A *security group* acts as a firewall that controls the traffic for one or more instances. When you launch an instance, you associate one or more security groups with the instance. You add rules to each security group that allow traffic to or from its associated instances. You can modify the rules for a security group at any time. The new rules are automatically applied to all instances that are associated with the security group.

The security groups created and assigned to the individual instances as part of this solution are restricted as much as possible while allowing access to the various functions needed by MongoDB. We recommend reviewing security groups to further restrict access as needed once the cluster is up and running.

Additional Resources

AWS services

- Getting Started
  http://docs.aws.amazon.com/gettingstarted/latest/awsgsg-intro/intro.html
- AWS CloudFormation
  http://aws.amazon.com/documentation/cloudformation/
- Amazon EC2
  - User's guide:
    http://aws.amazon.com/documentation/ec2/
  - Regions and Availability Zones:
    http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html
  - Key pairs:
    http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html
  - Instance stores:
    http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/InstanceStorage.html#instance-storage-concepts
  - FAQ:
    http://aws.amazon.com/ec2/faqs
- Amazon EBS
  - Overview:
    http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AmazonEBS.html
  - Volume types:
    http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html
- Amazon Identity and Access Management
  - User's guide:
    http://aws.amazon.com/documentation/iam/
  - Benefits of the IAM role:
    http://docs.aws.amazon.com/IAM/latest/UserGuide/role-usecase-ec2app.html

- Amazon VPC
  - Documentation:
    http://aws.amazon.com/documentation/vpc/
  - High availability for NAT instances
    https://aws.amazon.com/articles/2781451301784570
- AWS Security Center
  http://aws.amazon.com/security/
- AWS Simple Monthly Calculator
  http://calculator.s3.amazonaws.com/index.html

MongoDB

- MongoDB on AWS: Guidelines and Best Practices
  http://d0.awsstatic.com/whitepapers/AWS_NoSQL_MongoDB.pdf
- MongoDB production notes
  http://docs.mongodb.org/master/administration/production-notes/
- MongoDB MMS documentation
  https://docs.mms.mongodb.com/
- MongoDB Architecture Guide
  http://www.mongodb.com/lp/white-paper/architecture-guide
- Performance Best Practices for MongoDB
  http://www.mongodb.com/lp/white-paper/performance-best-practices
- MongoDB Multi-Data Center Deployments
  http://www.mongodb.com/lp/white-paper/multi-dc
- Testing the MongoDB Server
  http://www.mongodb.org/about/contributors/tutorial/test-the-mongodb-server/
- MongoDB Backup Methods
  http://docs.mongodb.org/manual/core/backups/
- mongodump reference
  http://docs.mongodb.org/manual/reference/program/mongodump/

Additional Quick Start Reference Deployments

- https://aws.amazon.com/quickstart/

Appendix: Security Group Specifics

The following tables show the configured inbound and outbound protocols and ports allowed for the various instances deployed by this Quick Start.

NAT Security Group			
Inbound			
Source	Protocol	Port Range (Service)	Comments
Restricted to CIDR block specified during the deployment process	TCP	22 (SSH)	Allows inbound SSH access to Linux instances from your network (over the Internet gateway).
10.0.0.0/16	TCP	80 (HTTP)	Allows inbound HTTP access only from instances deployed in the Amazon VPC.
10.0.0.0/16	TCP	443 (HTTPS)	Allows inbound HTTPS access only from instances deployed in the Amazon VPC.
Outbound			
Destination	Protocol	Port Range	Comments
10.0.1.0/24	TCP	22 (SSH)	Allows SSH access from the NAT instance to the 10.0.1.0 subnet.
0.0.0.0/0	TCP	80 (HTTP)	Allows outbound HTTP access from instances deployed in the Amazon VPC to anywhere.
0.0.0.0/0	TCP	443 (HTTPS)	Allows outbound HTTPS access from instances deployed in the Amazon VPC to anywhere.
10.0.2.0/24	TCP	22 (SSH)	Allows SSH access from the NAT instance to the 10.0.2.0 subnet (Primary replica).
10.0.3.0/24	TCP	22 (SSH)	Allows SSH access from NAT instance to the 10.0.3.0 subnet (Secondary0 replica).
10.0.4.0/24	TCP	22 (SSH)	Allows SSH access from the NAT instance to the 10.0.4.0 subnet (Secondary1 replica).

MongoDB Cluster Nodes			
Inbound			
Source	Protocol	Port Range (Service)	Comments
Restricted to CIDR block specified during the deployment process	TCP	22 (SSH)	Allows inbound SSH access to Linux instances from your network (over the Internet gateway).
Custom TCP rule	TCP	1-65535	10.0.2.0/24
Custom TCP rule	TCP	1-65535	10.0.3.0/24
Custom TCP rule	TCP	1-65535	10.0.4.0/24
Outbound			
Destination	Protocol	Port Range	Comments
0.0.0.0/0	TCP	1-65535	Allows outbound access from all the cluster nodes to anywhere.

Send Us Feedback

We welcome your questions and comments. Please post your feedback on the AWS Quick Start Discussion Forum.

You can visit our GitHub repository to download the templates and scripts for this Quick Start, and to share your customizations with others.