MongoDB University

MongoDB Ops Manager Workshop

MongoDB Ops Manager Workshop

Release 2.0

MongoDB, Inc.

June 20, 2016

Contents

1 MongoDB Ops Manager 2

1.1 Lab: Ops Manager Installation . . . 2

1.2 Lab: Ops Manager User Administration . . . 4

1.3 Lab: Enable the Ops Manager Public API . . . 6

1.4 Lab: Secure Replica Set . . . 6

1.5 Lab: Reconfig Replica Set . . . 8

1 MongoDB Ops Manager

Lab: Ops Manager Installation **(page 2)** Introduction to Ops Manager and installation

Lab: Ops Manager User Administration **(page 4)** Managing groups and users in Ops Manager

Lab: Enable the Ops Manager Public API **(page 6)** Setting up API access in Ops Manager

Lab: Secure Replica Set **(page 6)** Deploy a secure replica set using Ops Manager

Lab: Reconfig Replica Set **(page 8)** Reconfigure a replica set using the Ops Manager API

1.1 Lab: Ops Manager Installation

Permise

Ops Manager is a solution for on-prem MongoDB cluster operational management.

Enables features like:

- Automation
- Backup and Recovery
- Monitoring

Over the course of this lab we will be installing Ops Manager with high availability and scalability in mind.

Ops Manager HA

Ops Manager requires a number of servers for high availability.

- Monitoring and backup/recovery are essential for production operations.
- Therefore, it's important to assure high availability for Ops Manager.
- For this we need to follow a specific deployment topology.

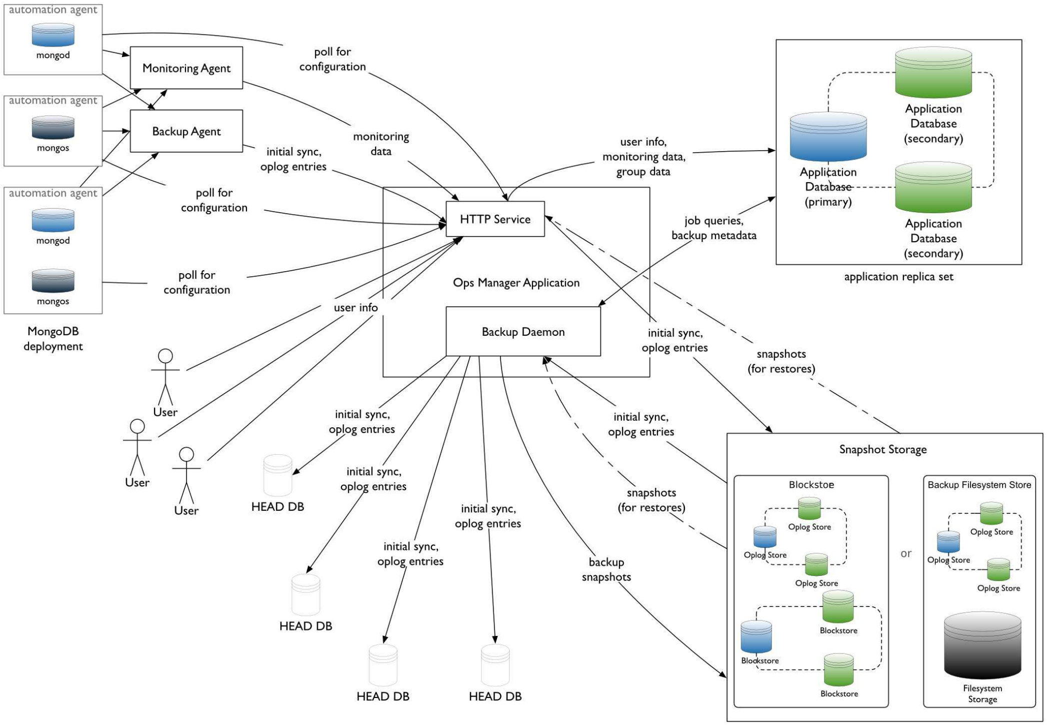
Ops Manager Scalability

Why do we need our operations tool to be scalable?

- The main reason is backup and recovery requirements
- The amount of data individual applications generate will grow
- The number of applications your Ops Manager deployment supports will grow
- Plan to accommodate both forms of growth

Ops Manager Architecture Review

Let's review Ops Manager architecture[1]:



Launch Ops Manager MongoDB Instances

It's time to set up the virtual machines that will support Ops Manager Deployment.

We will need the following instances:

- Two Replica Set Clusters
  - Application Replica Set
  - BlockStore Replica Set
- Two instances to run redundant services of Ops Manager Application
- Load Balancer

[1] https://docs.opsmanager.mongodb.com/current/core/system-overview/

1. Configure Ops Manager Application Database

Ops Manager needs to store data:

- Configuration of nodes, groups, users
- Metrics for monitoring
- Backup metdata and job queries

Also consider relevant security settings[2] for this database.

From the available machines go ahead and set up a Replica Set to support the *Application Database*.

2. Configure and Launch OpsManager Service

Habemus Replica Set!

Now it's time to launch **Ops Manager** service. For this you will need to:

- Configure Ops Manager into two separate Virtual Machines
- Point these to the previously configured Replica Set
- Allow offline binary access[3]
- Configure Load Balancer
- Launch Ops Manager services

3. Install OpsManager Automation Agents

At this point **Ops Manager** should be up and running Now it's time to install our Automation Agents[4]:

- In the remaining VM, install the automation agent
- Make sure that all nodes are discoverable on the servers dashboard
- Validate that all agents are reporting pings correctly

1.2 Lab: Ops Manager User Administration

Learning Objectives

Upon completing this lab, students will be able to:

- Administer Ops Manager groups
- Identify the differences between Ops Manager user roles
- Create and define Ops Manager users

[2] https://docs.mongodb.com/manual/administration/security-checklist/

[3] https://docs.opsmanager.mongodb.com/current/tutorial/configure-local-mode/

[4] https://docs.opsmanager.mongodb.com/current/tutorial/nav/install-automation-agent/

Exercise: Create Group

Connect to your Ops Manager instance and create the following `group`:

- **CIRCUS_MAXIMUS**

Exercise: Create Users

Using the Ops Manager API[5] , create the following users:

- **aediles@localhost.com** :
  - password: "123ABCabc!"
  - role: Owner[6]
- **patrician@localhost.com** :
  - password: "123ABCabc!"
  - role: Monitoring Admin[7]
- **consus@localhost.com** :
  - password: "&o7chac0v3r3d"
  - role: Backup Admin[8]

Exercise: Create Global Users

In various different situations, we will need users with global roles. Please create, either through the API or web console, the following users:

- First user with Global Automation Admin[9] role : *automater@localhost.com*
- Second user granted Global User Admin[10] user : *masterchef@localhost.com*

After creating these users, connect with the most appropriate user to change the password of the **CIRCUS_MAXIMUS** *Owner* user. The new password should be *"$uperC00l"*

This last operation should be accomplished using the HTTP Rest API interface.

[5] https://docs.opsmanager.mongodb.com/current/api/

[6] https://docs.opsmanager.mongodb.com/current/reference/user-roles/#owner

[7] https://docs.opsmanager.mongodb.com/current/reference/user-roles/#monitoring-admin

[8] https://docs.opsmanager.mongodb.com/current/reference/user-roles/#backup-admin

[9] https://docs.opsmanager.mongodb.com/current/reference/user-roles/#global-automation-admin

[10] https://docs.opsmanager.mongodb.com/current/reference/user-roles/#global-user-admin

1.3 Lab: Enable the Ops Manager Public API

Learning Objectives

Upon completing this lab, students will be able to:

- Understand the requirements for enabling Ops Manager Public API
- Configure Ops Manager groups to allow Public API requests

Exercise: Selective Node Rest Client

Ops Manager enables administrators to determine from where Public API calls are allowed. From which client nodes it accepts such interface requests.

Enable your deployment of Ops Manager to allow only one specific client to perform API calls.

- Generate an API Key called "generic"
- Add CIDR block for ip whitelisting enabling

Exercise: Enable Group Public API

Groups are more than just sets of machines and MongoDB instances. Groups also enclose security considerations.

Administrators have the ability to enable the Public API interface on a per group basis.

For this exercise go ahead and:

- Create a group called "LOVE_API_CALLS"
- Enable Public API for this group

1.4 Lab: Secure Replica Set

Premise

- Setting up a MongoDB Replica set is quite easy and fast.
- Setting up a Secured MongoDB replica set requires a few extra steps.
- In this lab we will be exploring how to setup a secured Replica Set through Ops Manager.

X.509 Authentication Mechanism

We will be using X.509 certificates[11] for authentication and TLS/SSL network encryption.

Ops Manager Group SSL and Auth

To build secured MongoDB deployments you first need to enable Auth and SSL[12] on your group.

All VMs have a set of certificates that you will be using to configure your secured deployment.

In folder */shared/etc/ssl* you will find:

- `ca.pem`: SSL CA certificate
- `automation.pem`: Automation agent certificate
- `backup.pem`: Backup agent certificate
- `monitor.pem`: Monitoring agent certificate
- `nodeX.pem`: Replica set member certificates (X)
- `dbadmin.pem`: MongoDB DB Admin certificate

`VERYSAFE` Group

Let's start by creating a group called `VERYSAFE` that has SSL enabled.

- Using the existing certificates, configure the agents accordingly.
- You need to specify certificates for
  - Certificate Authority
  - Monitoring Agent
  - Backup Agent
  - Automation Agent
- **The existing certificates do not have any decryption password!**

[11] https://docs.mongodb.com/manual/core/security-x.509/

[12] https://docs.opsmanager.mongodb.com/current/tutorial/enable-ssl-for-a-deployment/

Secure Replica Set Deploy

Once the automation agent has been reconfigured and servers are detected on your deployment, it's then time to deploy our secure replica set.

Create a replica set named **SECURE** with the following configuration:

- 3 Nodes: port range {27000 -> 27002}
- **clusterAuthMechanism**: x509
- **sslMode**: requiredSSL
- **sslPEMKeyFile**: */shared/etc/ssl/nodeX.pem*

X509 Users

Time to create users that will authenticate using an X.509 certificate.

- Go ahead and create a dbAdminAnyDatabase[13] user that authenticates using `dbadmin.pem` certificate.
- To create users that authenticate using X509 certificates you should check Certificate Subject as user[14] docs
- After the user has been created, connect to the *Primary* node of the replica set and create database "`allgood`".

1.5 Lab: Reconfig Replica Set

Learning Objectives

Upon completing this lab, students should be able to:

- Reconfigure a replica set
- Outline the different stages of deployments

Dependencies

- In order to complete all purposed exercises we need to first enable the Public API.
- Go to your group settings and enable the Public API.
- Do not forget to set an appropriate CIDR block for the IP whitelist and Generate the API Key.

[13] https://docs.mongodb.com/manual/reference/built-in-roles/#dbAdminAnyDatabase

[14] https://docs.mongodb.com/manual/tutorial/configure-x509-client-authentication/#add-x-509-certificate-subject-as-a-user

Exercise: Initial Replica Set

- Using the Ops Manager UI go ahead and create a 3 node replica set:
  - Replica set name `METAMORPHOSIS`
  - Two data bearing nodes
  - One arbiter
- *All instances should be installed using MongoDB 3.2.1*

Exercise: Add Replica Set Members

- Let's assume that we require highler level of High Availability (HA).
- Add 2 new data bearing nodes
  - First node should have priority 0
  - Second node should be an hidden replica.

Exercise: Decomission Replica Member

- One of your nodes is not making the cut.
- Change your replica set by "decommissioning" one of the instances
- Make sure that your replica set keeps up majority and previous level of node failure resilience

Exercise: Upgrade MongoDB Version

- Our CTO, for compliance reasons, demands that all of our nodes should be on the latest version of MongoDB.
- Upgrade all nodes in your replica set without downtime.

Exercise: Upgrade Replica Set Name

- We decided that we are not going to change our cluster going forward.
- Using Ops Manager API, update the replica set name to "PRODUCTION"