MongoDB Management Suite Manual

Release 1.5-pre

MongoDB, Inc.

August 14, 2014

Contents

On-Prem MongoDB Management Service is a package for managing MongoDB deployments. On-Prem MongoDB Management Service provides MMS Monitoring and MMS Backup, which helps users optimize clusters and mitigate operational risk.

You can also download a PDF edition of the MMS Manual.

On Prem MMS Application Overview Introduces the operation and architecture of the MMS application, and describes the requirements for running MMS On Prem.

Install On Prem MMS Install the On Prem Application Components.

On Prem MMS Administration Configure and manage On-Prem MongoDB Management Service.

On Prem MMS Monitoring High level overview of the On Prem MMS Monitoring service.

On Prem MMS Backup High level overview of the On Prem MMS Backup service.

Frequently Asked Questions Common questions about the operation and use of MMS.

Reference Reference material for MMS components and operations.

Release Notes Changelogs and notes on MMS releases.

1 On Prem MMS Application Overview

MMS Functional Overview Describes the operation of the MMS application.

On Prem MMS Components Describes the components and operation of the MMS backup application.

On Prem MMS Example Deployments Diagrams of possible MMS application deployment patterns.

1.1 MMS Functional Overview

Overview

The MongoDB Management Service (MMS) is a service for monitoring and backing up a MongoDB infrastructure.

MMS Monitoring

MMS provides real-time reporting, visualization and alerting on key database and hardware indicators and presents the data in an intuitive web dashboard.

A lightweight Monitoring Agent runs within your infrastructure and connects to the configured MongoDB instances. The Monitoring Agent collect statistics from the nodes in your deployment and transmit it back to MMS. The MMS user interface allowsthe user to view the visualized data and set alerts.

MMS Backup

Engineered specifically for MongoDB, MMS Backup features scheduled snapshots and point in time recovery. Once the service is up and running, MMS provides a web interface to support backup and restoration. MMS Backup supports horizontal scaling.

A lightweight Backup Agent runs within your infrastructure and connects to the configured MongoDB instances. The agent performs an initial sync and then tails the oplog of a *replica set's primary*. For a *sharded cluster*, the Backup Agent tails the primary of each shard and each config server. The agent ships initial sync and oplog data over HTTPS back to the MMS Backup service.

The MMS Backup service recreates every replica set you backup and applies the *oplog* entries sent by the Backup Agents. MMS then maintains a standalone MongoDB database on disk, also called a *head*, for each backed up replica set. Each head is consistent with the original primary up to the last oplog supplied by the agent. The initial sync and tailing of the oplog are all done using standard MongoDB queries.

To backup a replica set, the Backup service uses a `mongod` with a version equal to or greater than the version of the replica set it backs up.

Operations The MMS Backup service recreates every replica set you back up and applies the oplog entries the Backup Agents send. The production replica set, or sharded cluster, is not aware of the copy of the backup data. The initial sync and tailing of the oplog are all done using standard MongoDB queries.

The service takes scheduled snapshots of all heads and retains those snapshots based on a user-defined policy. On Prem MMS Backup captures snapshots of replica sets interval based on an observed change in oplog time. Sharded

clusters snapshots temporarily stop the balancer via the `mongos` so that they can insert a marker token into all shards and config servers in the cluster. MMS takes a snapshot when the marker tokens appear in the backup data.

Compression and block-level deduplication technology reduces snapshot data size. The snapshot only stores the differences between successive snapshots. Snapshots use only a fraction of the disk space required for full snapshots.

Restores Restores of specific snapshots and point in time restores are both available for replica sets. Clusters restore from a snapshot time for consistency.

A snapshot restore reads directly from the Backup Blockstore Database and transfers files via an HTTPS download link (pull) or by the MMS service sending files via SSH (push).

A point in time restore first creates a local restore of a snapshot from the blockstore. After the MMS service has the snapshot locally it applies stored oplogs until the desired point in time. The service then delivers the point in time backup via the same HTTPS or SSH mechanisms.

The amount of oplog to keep per backup is configurable and affects the time window available for point in time restores.

1.2 On Prem MMS Components

See *On Prem MMS Example Deployments* for diagrams of potential deployment architectures and *On Prem MMS Reference* for system reference.

MMS Application Package

The front-end package contains the UI the end user interacts with, as well as HTTPS services used by the Monitoring Agent and Backup Agent to transmit data to and from MMS. All three components start automatically when the front-end MMS package starts. These components are stateless. Multiple instances of the front-end package can run as long as each instance has the same configuration. Users and agents can interact with any instance.

For MMS Monitoring, you **only** need to install the application package. The application package consists of the following components:

- MMS Application and Monitoring Server
- MMS Backup Ingestion Server
- MMS Backup Alerts Service

MMS HTTP Service

The HTTP server runs on port `8080` by default. This component contains the web interface for managing MMS users, monitoring of MongoDB servers, and managing those server's backups. Users can sign up, create new accounts and groups, as well as join an existing group. The MMS Web Server also contains endpoints used by the MMS Agent to report back information on monitored MongoDB instances.

Backup HTTP Service

The HTTP server runs on port `8081` by default. The Backup HTTP Service contains a set of web services used by the Backup Agent. The agent retrieves its configuration from this service. The agent also sends back initial sync and oplog data through this interface. There is no user interaction with this service. The Backup HTTP service runs on port `8081` by default.

The Backup HTTP Service exposes an endpoint that reports on the state of the service and the underlying database to support monitoring of the Backup service. This status also checks the connections from the service to the *MMS Application Database* and the *MMS Backup Blockstore Database*. See *Backup HTTP Service Endpoint*.

Backup Alert Service

The Backup Alert Service watches the state of all agents, local copies of backed up databases, and snapshots. It sends email alerts as problems occur. The Backup Alert Service exposes a health-check endpoint. See *Backup Alert Service Endpoint*.

Backup Daemon Package

Backup Daemon

The Backup Daemon is the only component in the Backup Daemon Package. The Backup Daemon manages all local copies of backed up database (i.e. HEADs) as well as backup snapshots. The daemon does scheduled work based on data coming in to the Backup HTTP Service from the Backup Agents. No client applications talk directly to the daemon. Its state and job queues come from the *MMS Application Database*.

The daemon creates a local copy of the backed up database on in its local storage in the `rootDirectory` path. If you run multiple Backup Daemons, when you add a new backup the system selects a daemon for that instance and the local copy of that instance resides with that Daemon.

The daemon will take scheduled snapshots and store the snapshots in the Snapshot Storage (also known as the *Blockstore*). It will also act on restore requests by retrieving data from the Blockstore and delivering it to the requested destination.

The server running the Backup Daemon acts as a hidden secondary for every *replica set* assigned to it.

Multiple Backup Daemons can increase your storage by scaling horizontally and can provide **manual** failover.

The Backup Daemon exposes a health-check endpoint. See *Backup Daemon Endpoint*.

Data Storage

MMS uses dedicated MongoDB databases to store the MMS Application's monitoring data and the Backup Service's snapshots. The "backing databases," run on separate dedicated replica sets, which are the *backing MongoDB instances*.

You can only use backing MongoDB instances for than storing MMS data. MMS requires that you use separate replica sets for each backing database and that these instances *only* host MMS data.

The backing databases are not part of the MMS package installation. Set them up separately and record their locations in the MMS configuration files.

MMS Application Database

This database will contain MMS users, groups, hosts, monitoring data, backup state, etc. This metadata should be small in size (less than 1 GB per monitored/backed up server) but will be updated frequently. It is highly recommended this database be configured as a replica set to provide durability and automatic failover from the MMS service. See *Preparing Backing MongoDB Instances* for more information.

MMS Backup Blockstore Database

This database contains all snapshots of databases backed up and oplogs retained for point in time restores. The blockstore database will require disk space proportional to the backed up databases.

Configure the Blockstore as a replica set to provide durability and automatic failover to the backup and restore components. See *Preparing Backing MongoDB Instances* for more information.

You **cannot** backup the blockstore database with MMS Backup.

Additional Information

To learn more about On Prem MMS Backup requirements and On Prem MMS Backup, see *Frequently Asked Questions: Backup* and the main *main Backup documentation page*.

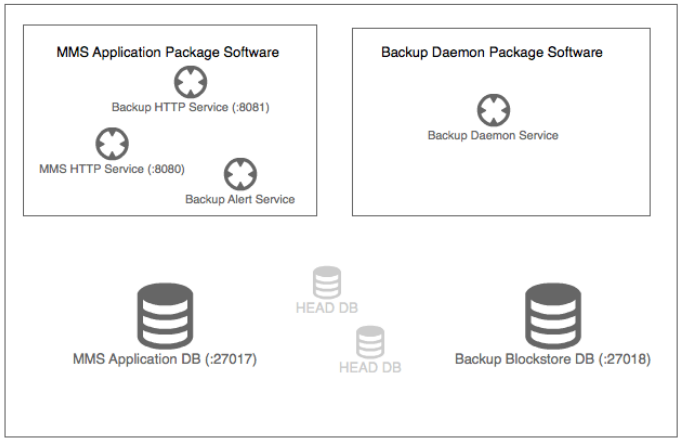
1.3 On Prem MMS Example Deployments

Consider the following diagrams of example MMS deployments:

Minimal Deployment

MMS Monitoring and Backup

Small deployment for development or testing



- The HEAD DBs are dynamically created/maintained by the Backup Daemon. The disk partition on which they live is specified in the conf-daemon.properties file.
- When deploying all components on a single server, please make note of the ulimit requirements in the documentation.

Cheat Sheet:

MMS Application Software Package

Start: sudo service mongodb-mms start
Stop: sudo service mongodb-mms stop
Log: /opt/mongodb/mms/logs
Configuration: /opt/mongodb/mms/conf/conf-mms.properties

Backup Daemon Software Package

Start: sudo service mongodb-mms-backup-daemon start
Stop: sudo service mongodb-mms-backup-daemon stop
Log: /opt/mongodb/mms-backup-daemon/logs
Configuration: /opt/mongodb/mms-backup-daemon/conf/conf-daemon.properties

MMS Application DB

Port: 27017
Data Directory: /mnt/data/mms

Backup Blockstore DB

Port: 27018
Data Directory: /mnt/data/mms-backup

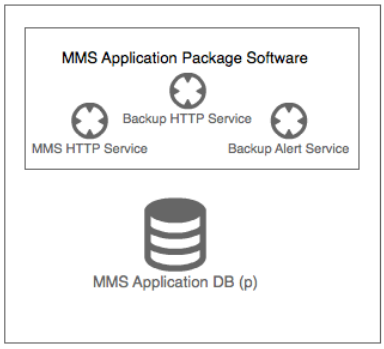
HEAD DBs

Port: Dynamic
Data Directory: /var/lib/mongodb/backup (see conf-daemon.properties)

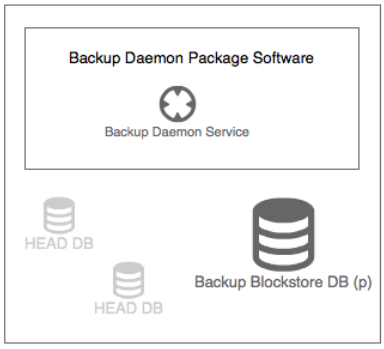
Moderate Deployment

MMS Monitoring and Backup

Medium-sized deployment with recommended data redundancy



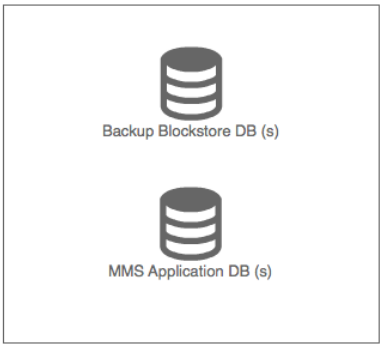
This server must satisfy the combined hardware and software requirements for
- MMS Application Server
- MMS Database Server



This server must satisfy the combined hardware and software requirements for
- Backup Blockstore Database Server
- Backup Daemon Server

Tips:

- The HEAD DBs are dynamically created/ maintained by the Backup Daemon. The disk partition on which they live is specified in the conf-daemon.properties file.
- For best performance HEAD DBs should never share a disk partition with the Backup Blockstore DB



If you do not wish to run with replica sets for the Backup Blockstore DB and the MMS Application DB, this server is optional. While we recommend using a replica set for data redundancy, it is not required.

This server must satisfy the combined hardware and software requirements for
- MMS Database Server
- Backup Blockstore Database Server

NOTE: All software services need to be able to communicate with both the MMS Application DB and the Backup Blockstore DB. Therefore, firewalls must allow traffic on appropriate ports between these servers.

Only the Backup Daemon Service needs to communicate with the HEAD DBs. The HEAD DBs will be automatically created with bind ips set to 127.0.0.1 so no external communication will occur.

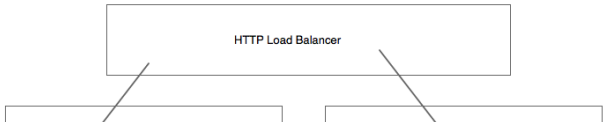
Full Deployment

MMS Monitoring and Backup

Large, HA deployment with multiple Backup Daemons and recommended data redundancy.

NOTE: All software services need to be able to communicate with both the MMS Application DBs and the Backup Blockstore DBs. Therefore, firewalls must allow traffic on appropriate ports between these servers.

Only the Backup Daemon Service needs to communicate with the HEAD DBs. The HEAD DBs



The load balancer balances HTTP traffic for the MMS HTTP Service and the Backup HTTP Service.

MMS does not provide a load balancer. This must be provided and configured by the user.

Preparing Backing MongoDB Instances Configure the underling MongoDB instances for On Prem MMS.

Install the On Prem MMS Application Documentation that describes the procedure for installing the core On Prem MMS Application and Monitoring Component with all supported packaging options.

Optional: Install On Prem MMS Backup Describes the procedure for installing the On Prem MMS Backup software.

Optional On Prem MMS Configuration Introduces optional On Prem MMS application configuration.

Start and Stop MMS Application Manage the On Prem MMS application process.

2.1 On Prem MMS Hardware and Software Requirements

For an overview of the components of an On Prem MMS deployment, see *On Prem MMS Components* and *On Prem MMS Example Deployments*. For a higher level view of MMS itself see *MMS Functional Overview*.

Hardware Requirements

To install MMS you will need a server for the MMS application, which includes Monitoring, and a MongoDB instance (i.e. replica set). Backup is an optional component of MMS, and if you want support for Backup, you must deploy additional servers for the Backup Daemon and for blockstore storage.

MMS Application Server

Deploy the MMS Application Package on the MMS Application Server with requirements according to the following table.

Number of Monitored Hosts	CPU Cores	RAM
Up to 400 monitored hosts	4+	15 GB
Up to 2000 monitored hosts	8+	15 GB
More than 2000 hosts	Contact MongoDB Account manager	Contact MongoDB Account manager

These requirements support only the software components in the MMS Application Package. If you wish to install the MMS Application Database on the same physical server, you will additionally need to satisfy the storage requirements for the *MMS Application Database*.

MMS Application Database Servers

Please see *Preparing Backing MongoDB Instances* for requirements for the MMS Application Database Servers.

Optional: MMS Backup Daemon

This server is required only if you are installing and running MMS Backup.

Number of Hosts	CPU Cores	RAM	Storage Capacity	Storage IOPS/s
Up to 200 hosts	4+ 2Ghz+	15 GB	Contact MongoDB Account manager	Contact MongoDB Account manager

A server running the Backup Daemon Package will act as a hidden secondary for every replica set assigned to it. Therefore this system must have the disk space and write capacity to maintain for all replica sets. To support point in

time restore the disk must have capacity to store an additional copy of the data. Typically daemons must be able to store 2 to 2.5 times the data size.

Before installing MMS Backup we recommend contacting your MongoDB Account Manager for assistance in estimating the storage requirements for your Backup Daemon server.

Optional: MMS Backup Blockstore Database Servers

Blockstore servers store snapshots of MongoDB instances. Only provision Blockstore servers if you are deploying MMS Backup.

Blockstores must have the capacity to store 2 to 3 times the total backed up data size. Please contact your MongoDB Account Manager for assistance in estimating the storage requirements for your blockstore server.

Medium grade HDDs will have enough I/O throughput to handle the load of the Blockstore. Each replica set member should have 4 x 2ghz+ CPU cores. We recommend 8 GB of RAM for every 1 TB disk of Blockstore to provide good snapshot and restore speed. MMS defines 1 TB of Blockstore as 1024^4 bytes.

Software Requirements

Operating System

MMS supports the following 64-bit Linux distributions:

- CentOS 5 or later,
- Red Hat Enterprise Linux 5, or later,
- SUSE 11 or Later,
- Amazon Linux AMI (latest version only,)
- Ubuntu 12.04 or later.

Ulimits

The MMS packages automatically raise the open file, max user processes, and virtual memory ulimits. On Red Hat, be sure to check for a `/etc/security/limits.d/90-nproc.conf` file that may override the max user processes limit. If the `/etc/security/limits.d/90-nproc.conf` file exists, remove it before continuing.

MongoDB

The MongoDB databases backing MMS must be MongoDB 2.4.9 or later. To back up a Replica set with MMS, all members of the set must run at least MongoDB 2.2 or later. To back up a sharded cluster, all components of the sharded cluster must be MongoDB 2.4.3 or later.

Web Browsers

On-Prem MongoDB Management Service supports clients using the following browsers:

- Chrome 8 and greater.
- Firefox 12 and greater.
- IE 9 and greater.

- Safari 6 and greater.

The MMS application will display a warning on non-supported browsers.

SMTP

MMS requires email for fundamental server functionality such as password reset and alerts.

Many Linux server-oriented distributions include a local SMTP server by default, for example, Postfix, Exim, or Sendmail. You also may configure MMS to send mail via third party providers, including Gmail and Sendgrid.

SNMP

If your environment includes SNMP, you can configure an SMNP trap receiver with periodic heartbeat traps to monitor the internal health of MMS.

For more details, see *Configure SNMP Heartbeat Support*.

2.2 Preparing Backing MongoDB Instances

Overview

A backing MongoDB instance is a dedicated replica set that hosts a *backing database* where MMS stores data. The MMS Application requires separate backing instances for monitoring and the Backup services. Backing instances are dedicated to MMS operation. Do not use these replica sets to store any other data.

All backing MongoDB instances must be running MongoDB 2.4.9 or later.

MMS Application Database

MMS Monitoring requires a MongoDB replica set to hold the MMS Application database. The database stores:

- Monitoring data collected from Monitoring Agents.
- Meta data for MMS users, groups, hosts, monitoring data, and backup state.

Topology The application database should be a single three-member MongoDB replica set. If you cannot allocate 3 data instances, you may deploy 2 `mongod` instances that hold data and one arbiter.

You may run the MMS Application Server on the same physical server as one member of the MMS Application Database replica set. A MongoDB standalone may also be used in please of a replica set, but this is not recommended for production deployments.

Hardware Specifications For up to 400 monitored hosts, the MMS Application Database requires 200 GB of storage space. For up to 2000 monitored hosts, the MMS Application Database requires 500 GB of storage space. If you have more than 2000 monitored hosts, please contact your MongoDB Account Manager.

The MMS Application software requires 15 GB of RAM. The MMS Application Database requires additional RAM for the monitored hosts, as follows:

- Up to 400 monitored hosts, 8 GB additional RAM.
- Up to 2000 monitored hosts, 15 GB additional RAM.

For the best results use SSD-backed storage.

Optional: MMS Backup Blockstore Database

MMS Backup requires a separate, **dedicated** MongoDB replica set to hold the Backup Service's snapshot data. This **cannot** be a replica set used for purposes other than holding the snapshots. In particular, this cannot be a replica set that stores the data the snapshots back up.

The replica set should have enough capacity to store 2 to 3 times the total backed up data size.

Prior to installing MMS Backup, consider contacting your MongoDB Account Manager to arrange a sizing consultation for the MMS Backup database.

For testing *only* you may use a standalone `mongod` instance in place of a replica set.

Prerequisites

Read the MongoDB Production Notes before installing MongoDB and other software on your servers. If you are deploying MMS on Amazon AWS, please see :`http://mms.mongodb.com/helptutorial/configure-aws-hosts`.

The MMS Application requires `$where`. As a result, you must ensure that `security.javaScriptEnabled` is `true`, which is the default, for all backing `mongod` instances.

Setup and Install Replica Sets

To create replica sets, start the `mongod` instances for each member of a new replica set, configure the replica set, then add the `mongod` instances to the replica set.

The Deploy a Replica Set guide has step by step details how to setup and deploy replica sets for use as backing MongoDBs for MMS.

Each backing instance should be a single three-member MongoDB replica set. If you cannot allocate 3 data instances, you may deploy 2 `mongod` instances that hold data and one arbiter.

For additional information on installing MongoDB, see Install MongoDB.

2.3 Install the On Prem MMS Application

Install from DEB Packages Describes the procedure for installing On Prem MMS on Debian and Ubuntu systems.

Install from RPM Packages Describes the procedure for installing On Prem MMS on Red Hat, Fedora, CentOS, and Amazon AMI Linux.

Install from Archives Describes the procedure for installing On Prem MMS on other Linux systems without using package management.

Quick Start Installation Describes the procedure for quickly setting up an On Prem MMS instance with all components running on a single machine. Use this process for becoming familiar with the installation process and to test the On Prem MMS Application before deploying a production version of the application.

Install or Upgrade the On-Prem MMS Monitoring with `deb` Packages

Overview

On-Prem MongoDB Management Service is a service to monitor and back up a MongoDB infrastructure. This tutorial describes the basic process to install or upgrade the MMS Application Package.

At a high level, a basic installation will look like the following. The estimated setup time is less than an hour.

1. Configure an MMS Application Server that meets the *hardware requirements*.
2. Install a single MongoDB replica set to be used for the MMS Application database, as described in *Preparing Backing MongoDB Instances*. You **cannot** use a replica set that is being used to store other data.
3. Install an SMTP email server as appropriate for your environment.
4. Install the MMS Application Package.
5. Configure the MMS server's URL, email addresses, and Mongo URI connection strings. See *Configuration* for more information.
6. Start up the MMS Application. See *Start and Stop MMS Application* for more information.
7. Optionally install the MMS Backup Service, as described in *Install On Prem MMS Backup with deb Packages*.

Prerequisites

Configure an MMS Application Server that meets the *hardware requirements and software requirements*.

Ensure that you have *deployed the backing MongoDB instances* that MMS uses to store application data. For additional information on installing MongoDB, see the MongoDB Installation tutorials.

Procedures

Install and Start the On Prem MMS Service

Step 1: Download On Prem MMS Monitoring. Download the latest On Prem Monitoring releases from the downloads page.

Step 2: Install On Prem MMS Monitoring. Install the `.deb` package by issuing the following command, where `<version>` is the version of the `.deb` package:

```
sudo dpkg --install mongodb-mms_<version>_x86_64.deb
```

When installed, the base directory for the MMS software is `http://mms.mongodb.com/helpopt/mongodb/mms/`. The `.deb` package creates a new system user `mongodb-mms` under which the server will run.

Step 3: Configure On Prem MMS Monitoring. In the `conf-mms.properties` file, ensure that the following required settings are correct:

- `mms.centralUrl`
- `mms.backupCentralUrl`
- `mms.fromEmailAddr`
- `mms.replyToEmailAddr`
- `mms.adminFromEmailAddr`
- `mms.adminEmailAddr`
- `mms.bounceEmailAddr`
- `mongo.mongoUri`

- `mongo.replicaSet`

Consider the following example configuration:

```
mms.centralUrl=http://<public_ip>:8080
mms.backupCentralUrl=http://<public_ip>:8081

mms.fromEmailAddr=mms-admin@example.net
mms.replyToEmailAddr=mms-admin@example.net
mms.adminFromEmailAddr=mms-admin@example.net
mms.adminEmailAddr=mms-admin@example.net
mms.bounceEmailAddr=mms-admin@example.net

mongo.mongoUri=mongodb://<mms_mongod_ip>:27017/
mongo.replicaSet=rs0
```

At this point, you can also configure authentication, email, and optional Kerberos integration, as described in the *Configuration*.

If you would like to run the MMS application in a highly available configuration, please consider *Configure a Highly Available MMS Application Server*.

Step 4: Start On Prem MMS Monitoring. To start MMS, issue the following command:

```
sudo service mongodb-mms start
```

Upgrade On-Prem MongoDB Management Service from 1.3 and Later If you have an existing On Prem MMS deployment, use the following procedure to upgrade to the latest release. There are no supported downgrade paths for On Prem MMS.

Step 1: *Recommended.* Take a full backup of the MMS database before beginning the upgrade procedure.

Step 2: Shut down MMS. For example:

```
sudo service mongodb-mms stop
```

Step 3: If you are running MMS Backup, shutdown the MMS Backup Daemon. The daemon may be installed on a different server. If is *critical* that this is also shut down. To shut down, issue a command similar to the following:

```
sudo service mongodb-mms-backup-daemon stop
```

Step 4: Save a copy of your previous configuration file. For example:

```
sudo cp /opt/mongodb/mms/conf/conf-mms.properties ~/.
```

Step 5: Upgrade the package. For example:

```
sudo dpkg -i mongodb-mms_<version>_x86_64.deb
```

Step 6: Edit the new configuration file. Fill in the new configuration file at `http://mms.mongodb.com/helpopt/mongodb/mms/conf/conf-mms.properties` using your old file as a reference point.

Step 7: Start MMS. For example:

```
sudo service mongodb-mms start
```

Step 8: Update all Monitoring Agents and Backup Agents. See *Getting Started with MMS Monitoring* for more information.

Step 9: Update the Backup Daemon Package and any Backup Agent, as appropriate. If you are running MMS Backup, update the Backup Daemon Package and any Backup Agent.

See *Install On Prem MMS Backup with deb Packages* and *Getting Started with MMS Backup* for more information.

Upgrade On-Prem MongoDB Management Service from 1.2 and Earlier Due to the company name change, the name of the MMS package changed between versions 1.2 and 1.3. Therefore, to upgrade the On-Prem MongoDB Management Service server from *any* version before 1.3, use the following procedure:

1. *Recommended.* Take a full backup of the MMS database before beginning the upgrade procedure.
2. Shut down MMS, using the following command:

   ```
   /etc/init.d/10gen-mms stop
   ```

3. Download the latest package from the downloads page and proceed with the instructions for a fresh install. Do not attempt to use your package manager to do an upgrade.

   When complete, On-Prem MongoDB Management Service is installed in the `http://mms.mongodb.com/helpopt/mongodb/mms` directory.
4. Follow all procedures for a new install include configuring the options in `http://mms.mongodb.com/helpopt/mongodb/mms/conf/conf-mms.properties`. If you used encrypted authentication credentials you will need to regenerate these manually. *Do not copy the credentials from your old properties file. Old credentials will not work.*
5. Start MMS using the new package name:

   ```
   sudo /etc/init.d/mongodb-mms start
   ```

6. Update any Monitoring Agent. See *Getting Started with MMS Monitoring* for more information.

Additional Information

- See *Configuration* for documentation of all configuration options for the MMS application.
- For complete instructions on managing the MMS application process, see *Start and Stop MMS Application*.
- To configure the MMS application for high availability, see *Configure a Highly Available MMS Application Server*.

Install or Upgrade the On Prem MMS Monitoring with `rpm` Packages

Overview

On-Prem MongoDB Management Service is a service to monitor and back up a MongoDB infrastructure. This tutorial describes the basic process to install or upgrade the MMS Application Package.

At a high level, a basic installation will look like the following. The estimated setup time is less than an hour.

1. Configure an MMS Application Server that meets the *hardware requirements*.
2. Install a single MongoDB replica set to be used for the MMS Application database, as described in *Preparing Backing MongoDB Instances*. You **cannot** use a replica set that is being used to store other data.
3. Install an SMTP email server as appropriate for your environment.
4. Install the MMS Application Package
5. Configure the MMS server's URL, email addresses, and Mongo URI connection strings. See *Configuration* for more information.
6. Start up the MMS Application. See *Start and Stop MMS Application* for more information.
7. Optionally install the MMS Backup Service, as described in *Install On Prem MMS Backup with rpm Packages*.

Prerequisites

Configure an MMS Application Server that meets the *hardware requirements and software requirements*.

Ensure that you have *deployed the backing MongoDB instances* that MMS uses to store application data. For additional information on installing MongoDB, see the MongoDB Installation tutorials.

Procedures

Install and Start the On Prem MMS Service

Step 1: Download On Prem MMS Monitoring. Download the latest On Prem Monitoring releases from the downloads page.

Step 2: Install On Prem MMS Monitoring. Install the `.rpm` package by issuing the following command, where `<version>` is the version of the `.rpm` package:

```
sudo rpm -ivh mongodb-mms-<version>.x86_64.rpm
```

When installed, the base directory for the MMS software is `http://mms.mongodb.com/helpopt/mongodb/mms/`. The RPM package creates a new system user `mongodb-mms` under which the server runs.

Step 3: Configure On Prem MMS Monitoring. In the `conf-mms.properties` file, ensure that the following required settings are correct:

- `mms.centralUrl`
- `mms.backupCentralUrl`
- `mms.fromEmailAddr`
- `mms.replyToEmailAddr`
- `mms.adminFromEmailAddr`
- `mms.adminEmailAddr`
- `mms.bounceEmailAddr`
- `mongo.mongoUri`
- `mongo.replicaSet`

Consider the following example configuration:

```
mms.centralUrl=http://<public_ip>:8080
mms.backupCentralUrl=http://<public_ip>:8081

mms.fromEmailAddr=mms-admin@example.net
mms.replyToEmailAddr=mms-admin@example.net
mms.adminFromEmailAddr=mms-admin@example.net
mms.adminEmailAddr=mms-admin@example.net
mms.bounceEmailAddr=mms-admin@example.net

mongo.mongoUri=mongodb://<mms_mongod_ip>:27017/
mongo.replicaSet=rs0
```

At this point, you can also configure authentication, email, and optional Kerberos integration, as described in the *Configuration*.

If you would like to run the MMS application in a highly available configuration, please consider *Configure a Highly Available MMS Application Server*.

Step 4: Start On Prem MMS Monitoring. To start MMS, issue the following command:

```
sudo service mongodb-mms start
```

Upgrade On-Prem MongoDB Management Service from 1.3 and Later If you have an existing On Prem MMS deployment, use the following procedure to upgrade to the latest release. There are no supported downgrade paths for On Prem MMS.

Step 1: *Recommended.* Take a full backup of the MMS database before beginning the upgrade procedure.

Step 2: Shut down MMS. For example:

```
sudo service mongodb-mms stop
```

Step 3: If you are running MMS Backup, shutdown the MMS Backup Daemon. The daemon may be installed on a different server. If is *critical* that this is also shut down. To shut down, issue a command similar to the following:

```
sudo service mongodb-mms-backup-daemon stop
```

Step 4: Save a copy of your previous configuration file. For example:

```
sudo cp /opt/mongodb/mms/conf/conf-mms.properties ~/.
```

Step 5: Upgrade the package. For example:

```
sudo rpm -U mongodb-mms-<version>.x86_64.rpm
```

Step 6: Move the new version of the configuration file into place. For example:

```
sudo mv /opt/mongodb-mms/conf/conf-mms.properties.rpmnew /opt/mongodb-mms/conf/conf-mms.properties
```

Step 7: Edit the new configuration file. Fill in the new configuration file at `http://mms.mongodb.com/helpopt/mongodb/mms/conf/conf-mms.properties` using your old file as a reference point.

Step 8: Start MMS. For example:

```
sudo service mongodb-mms start
```

Step 9: Update all Monitoring Agents and Backup Agents. See *Getting Started with MMS Monitoring* for more information.

Step 10: Update the Backup Daemon Package and any Backup Agent, as appropriate. If you are running MMS Backup, update the Backup Daemon Package and any Backup Agent.

See *Install On Prem MMS Backup with rpm Packages* and *Getting Started with MMS Backup* for more information.

Upgrade On-Prem MongoDB Management Service from 1.2 and Earlier Due to the company name change, the name of the MMS package changed between versions 1.2 and 1.3. Therefore, to upgrade the On-Prem MongoDB Management Service server from *any* version before 1.3, use the following procedure:

1. *Recommended.* Take a full backup of the MMS database before beginning the upgrade procedure.
2. Shut down MMS, using the following command:

   ```
   /etc/init.d/10gen-mms stop
   ```

3. Download the latest package from the downloads page and proceed with the instructions for a fresh install. Do not attempt to use your package manager to do an upgrade.

   When complete, On-Prem MongoDB Management Service is installed in the `http://mms.mongodb.com/helpopt/mongodb/mms` directory.
4. Follow all procedures for a new install include configuring the options in `http://mms.mongodb.com/helpopt/mongodb/mms/conf/conf-mms.properties`. If you used encrypted authentication credentials you will need to regenerate these manually. *Do not copy the credentials from your old properties file. Old credentials will not work.*
5. Start MMS using the new package name:

   ```
   sudo /etc/init.d/mongodb-mms start
   ```

6. Update any Monitoring Agent. See *Getting Started with MMS Monitoring* for more information.

Additional Information

- See *Configuration* for documentation of all configuration options for the MMS application.
- For complete instructions on managing the MMS application process, see *Start and Stop MMS Application*.
- To configure the MMS application for high availability, see *Configure a Highly Available MMS Application Server*.

Install or Upgrade the On Prem MMS Service: tar.gz and zip

Overview

On-Prem MongoDB Management Service is a service to monitor and back up a MongoDB infrastructure. This tutorial describes the basic process to install or upgrade the MMS Application Package.

At a high level, a basic installation will look like the following. The estimated setup time is less than an hour.

1. Configure an MMS Application Server that meets the *hardware requirements*.
2. Install a single MongoDB replica set to be used for the MMS Application database, as described in *Preparing Backing MongoDB Instances*. You **cannot** use a replica set that is being used to store other data.
3. Install an SMTP email server as appropriate for your environment.
4. Install the MMS Application Package.
5. Configure the MMS server's URL, email addresses, and Mongo URI connection strings. See *Configuration* for more information.
6. Start up the MMS Application. See *Start and Stop MMS Application* for more information.
7. Optionally install the Backup Service. See *Optional: Install On Prem MMS Backup* for more information.

Prerequisites

Configure an MMS Application Server that meets the *hardware requirements and software requirements*.

Ensure that you have *deployed the backing MongoDB instances* that MMS uses to store application data. For additional information on installing MongoDB, see the MongoDB Installation tutorials.

Procedures

Install and Start the On Prem MMS Service

Step 1: Download On Prem MMS Monitoring. Download the latest On Prem Monitoring releases from the downloads page.

Step 2: Install On Prem MMS Monitoring. You can install On Prem MMS Monitoring from the provided `tar.gz` or `zip` archive without making any changes to the underlying system (i.e. without creating users). To install, extract the package, as in the following command:

```
tar -zxf mongodb-mms-<version>.x86_64.tar.gz
```

When complete, On-Prem MongoDB Management Service is installed in the `http://mms.mongodb.com/helpopt/mongodb/mms` directory.

Step 3: Create a symlink (optional). Optionally create a symlink in `/etc/init.d` to the included control script for convenience, as in the following:

```
sudo ln -s <install_dir>/bin/mongodb-mms /etc/init.d/
```

Note, when the app is first started, it will create and store an encryption key in `$HOME/.mongodb-mms` for the app user.

Step 4: Configure On Prem MMS Monitoring. In the `conf-mms.properties` file, ensure that the following required settings are correct:

- `mms.centralUrl`
- `mms.backupCentralUrl`
- `mms.fromEmailAddr`
- `mms.replyToEmailAddr`
- `mms.adminFromEmailAddr`
- `mms.adminEmailAddr`
- `mms.bounceEmailAddr`
- `mongo.mongoUri`
- `mongo.replicaSet`

Consider the following example configuration:

```
mms.centralUrl=http://<public_ip>:8080
mms.backupCentralUrl=http://<public_ip>:8081

mms.fromEmailAddr=mms-admin@example.net
mms.replyToEmailAddr=mms-admin@example.net
mms.adminFromEmailAddr=mms-admin@example.net
mms.adminEmailAddr=mms-admin@example.net
mms.bounceEmailAddr=mms-admin@example.net

mongo.mongoUri=mongodb://<mms_mongod_ip>:27017/
mongo.replicaSet=rs0
```

At this point, you can also configure authentication, email, and optional Kerberos integration, as described in the *Configuration*.

If you would like to run the MMS application in a highly available configuration, please consider *Configure a Highly Available MMS Application Server*.

Step 5: Start On Prem MMS Monitoring. To start MMS, issue the following command:

```
sudo /etc/init.d/mongodb-mms start
```

Upgrade On-Prem MongoDB Management Service from 1.3 and Later If you have an existing On Prem MMS deployment, use the following procedure to upgrade to the latest release. There are no supported downgrade paths for On Prem MMS.

To upgrade a tarball installation, backup the configuration file and logs, and then re-install the On Prem MMS server.

Important: It is crucial that you back up the existing configuration because the upgrade process will delete existing data.

In more detail:

Step 1: Shutdown the MMS server and take a backup of your existing configuration and logs. For example:

```
sudo /etc/init.d/mongodb-mms stop
sudo cp -a <install_dir>/conf ~/mms_conf.backup
sudo cp -a <install_dir>/logs ~/mms_logs.backup
```

Step 2: If you are running MMS Backup, shutdown the MMS Backup Daemon. The daemon may be installed on a different server. If is *critical* that this is also shut down. To shut down, issue a command similar to the following:

```
sudo /etc/init.d/mongodb-mms-backup-daemon stop
```

Step 3: Remove your existing MMS server installation entirely and extract latest release in its place. For example:

```
cd <install_dir>/../
sudo rm -rf <install_dir>
sudo tar -zxf -C . /path/to/mongodb-mms-<version>.x86_64.tar.gz
```

Step 4: Compare and reconcile any changes in configuration between versions. For example:

```
diff -u ~/mms_conf.backup/conf-mms.properties <install_dir>/conf/conf-mms.properties
diff -u ~/mms_conf.backup/mms.conf <install_dir>/conf/mms.conf
```

Step 5: Edit your configuration to resolve any conflicts between the old and new versions. Make any changes as appropriate. Changes to `mms.centralUri`, email addresses, and MongoDB are the most common configuration changes.

Step 6: Restart the On Prem MMS server. For example:

```
sudo /etc/init.d/mongodb-mms start
```

Step 7: Update all Monitoring Agents and Backup Agents. See *Getting Started with MMS Monitoring* for more information.

Step 8: Update the Backup Daemon Package and any Backup Agent, as appropriate. If you are running MMS Backup, update the Backup Daemon Package and any Backup Agent.

See *Optional: Install On Prem MMS Backup* and *Getting Started with MMS Backup* for more information.

Upgrade On-Prem MongoDB Management Service from 1.2 and Earlier Due to the company name change, the name of the MMS package changed between versions 1.2 and 1.3. Therefore, to upgrade the On-Prem MongoDB Management Service server from *any* version before 1.3, use the following procedure:

1. *Recommended.* Take a full backup of the MMS database before beginning the upgrade procedure.
2. Shut down MMS, using the following command:

   ```
   /etc/init.d/10gen-mms stop
   ```

3. Download the latest package from the downloads page and proceed with the instructions for a fresh install. Do not attempt to use your package manager to do an upgrade. See *Install On Prem MMS* for more information.

   When complete, On-Prem MongoDB Management Service is installed in the `http://mms.mongodb.com/helpopt/mongodb/mms` directory.

4. Follow all procedures for a new install include configuring the options in `http://mms.mongodb.com/helpopt/mongodb/mms/conf/conf-mms.properties`. If you used encrypted authentication credentials you will need to regenerate these manually. *Do not copy the credentials from your old properties file. Old credentials will not work.*

5. Start MMS using the new package name:

   ```
   sudo /etc/init.d/mongodb-mms start
   ```

6. Update any Monitoring Agent. See *Getting Started with MMS Monitoring* for more information.

Additional Information

- See *Configuration* for documentation of all configuration options for the MMS application.
- For complete instructions on managing the MMS application process, see *Start and Stop MMS Application*.
- To configure the MMS application for high availability, see *Configure a Highly Available MMS Application Server*.

Install On Prem MMS for Development and Testing

Overview

MMS On Prem is a package that lets you run the MongoDB Management Service (MMS) on site. MMS monitors and backs up your MongoDB infrastructure.

On Prem MMS uses the components described in *On Prem MMS Components*. In a test deployment, you can run an entire On Prem MMS deployment on a single system as in the *Minimal Deployment* diagram. You will deploy a replica set for testing on a separate system.

The minimal deployment is for testing and development purposes only. Running on a single server is *not** suitable for production deployments.

Procedures

Set up the On Prem Service

Step 1: Set up a server and firewall. Prepare the server that will run the MMS On Prem components, the Application Database, and Backup Blockstore Database.

Use a RHEL 6+ or Amazon Linux Server with at least:

- 15 GB of memory
- 50 GB of disk space for the root partition

For example, you can meet the size requirements by using an AWS EC2 `m3.xlarge` instance and changing the size of the root partition from 8 GB to 50 GB. When you log into the instance, execute "`df -h`" to verify the root partition has 50 GB of space.

Set up a firewall or EC2 Security Group that:

- Allows administrators to SSH into the server.
- Allows MMS users to connect from browsers to ports `8080` and `8081` through the server's public IP address.

Step 2: Configure `ulimits`. Remove the default `ulimit` settings that come with the operating system:

```
sudo rm /etc/security/limits.d/90-nproc.conf
```

Edit the `/etc/security/limits.conf` file to configure the following settings:

```
* soft nofile 64000
* hard nofile 64000
* soft nproc 32000
* hard nproc 32000
```

Step 3: Install MongoDB. Use the following series of commands to install MongoDB, which you use for the MMS Application Database and the MMS Backup Blockstore Database.

Set up a repository definition by issuing the following command:

```
echo "[MongoDB]
name=MongoDB Repository
baseurl=http://downloads-distro.mongodb.org/repo/redhat/os/x86_64
gpgcheck=0
enabled=1" | sudo tee -a /etc/yum.repos.d/mongodb.repo
```

Install MongoDB by issuing the following two commands:

```
sudo yum install -y mongodb-org mongodb-org-shell
```

Step 4: Download the MMS Application Package. Download the latest version of the MMS Application Package from the MMS downloads page. The MMS Application Package is listed as "Monitoring and Core" on the downloads page.

Download the On Prem MMS Package by issuing the following command. Substitute the MMS version for `<version>`:

```
sudo curl -OL https://downloads.mongodb.com/on-prem-mms/rpm/mongodb-mms-<version>.x86_64.rpm
```

For example, for version `1.4.2.73-1` issue:

```
curl -OL https://downloads.mongodb.com/on-prem-mms/rpm/mongodb-mms-1.4.2.73-1.x86_64.rpm
```

Step 5: Install the MMS Application Package. Install the package using the following command, where `<version>` is the MMS version:

```
sudo rpm --install mongodb-mms-<version>.x86_64.rpm
```

For example, for version `1.4.2.73-1` issue:

```
sudo rpm --install mongodb-mms-1.4.2.73-1.x86_64.rpm
```

Step 6: Download the Backup Daemon Package. Alternately, you can download the package by issuing the following command, where `<version>` is the MMS version:

```
curl -OL https://downloads.mongodb.com/on-prem-mms/rpm/mongodb-mms-backup-daemon-<version>.x86_64.rpm
```

For example, for version `1.4.2.73-1` issue:

```
curl -OL https://downloads.mongodb.com/on-prem-mms/rpm/mongodb-mms-backup-daemon-1.4.2.73-1.x86_64.rp
```

Step 7: Install the Backup Daemon Package. Install the package using the following command, where `<version>` is the MMS version:

```
sudo rpm --install mongodb-mms-backup-daemon-<version>.x86_64.rpm
```

For example, for version `1.4.2.73-1`, you would issue the following:

```
sudo rpm --install mongodb-mms-backup-daemon-1.4.2.73-1.x86_64.rpm
```

Step 8: Set up the two backing databases. Create a data directory for each backing database and set `mongod.mongod` as each data directory's owner. The *backing databases* are the MMS Application Database and MMS Backup Blockstore Database.

The following command creates two data directories, one for each backing database. You can use different directory names:

```
sudo mkdir -p /data /data/mmsdb /data/backupdb
```

The following command sets `mongod.mongod` as owner of the new directories:

```
sudo chown mongod:mongod /data /data/mmsdb /data/backupdb
```

Step 9: Start the MongoDB instances for the two backing databases. Start each MongoDB instance using the `mongod` daemon and specifying `mongod` as the user. Start each instance on its own dedicated port number and with the data directory you created in the last step.

The following two commands start separate instances for the MMS Application Database and for the Backup Blockstore Database:

```
sudo -u mongod mongod --port 27017 --dbpath /data/mmsdb --logpath /data/mmsdb/mongodb.log --fork

sudo -u mongod mongod --port 27018 --dbpath /data/backupdb --logpath /data/backupdb/mongodb.log --fo
```

Step 10: Configure the MMS Application Service. Edit `http://mms.mongodb.com/helpopt/mongodb/mms/conf/con` Set values for the following properties, substituting your install's values for *<public_ip>* and *mms-admin@example.net*:

```
mms.centralUrl=http://<public_ip>:8080
mms.backupCentralUrl=http://<public_ip>:8081

mms.fromEmailAddr=mms-admin@example.net
mms.replyToEmailAddr=mms-admin@example.net
mms.adminFromEmailAddr=mms-admin@example.net
mms.adminEmailAddr=mms-admin@example.net
mms.bounceEmailAddr=mms-admin@example.net

mongo.mongoUri=mongodb://127.0.0.1:27017/
mongo.backupdb.mongoUri=mongodb://127.0.0.1:27018/
```

Step 11: Start the MMS Application Service. Issue the following command:

```
sudo service mongodb-mms start
```

Step 12: Open the MMS On Prem home page. Enter the following URL in a browser, where `<public_ip>` is the public IP address of the server:

```
http://<public_ip>:8080
```

Step 13: Configure the MMS Backup Daemon. Edit `http://mms.mongodb.com/helpopt/mongodb/mms-backup-dae` to configure the the following settings:

```
mongo.mongoUri=mongodb://127.0.0.1:27017/

mongo.backupdb.mongoUri=mongodb://127.0.0.1:27018/
```

Step 14: Start the MMS Backup Daemon. Issue the following command:

```
sudo service mongodb-mms-backup-daemon start
```

Begin Monitoring and Backing Up a Replica Set The following procedure creates a three-member *replica set*, populates it with data, and then uses the On Prem Service to monitor and backup the replica set.

Step 1: Set up the server that will run the MongoDB replica set. Use a RHEL 6+ or Amazon Linux Server with at least:

- 3 GB of memory.
- 50 GB of disk space for the root partition.

For example, you can meet the size requirements by using an AWS EC2 `m3.medium` instance and changing the size of the root partition from 8 to 50 gigabytes. When you log into the instance, use "`df -h`" to verify the root partition has 50 gigabytes of space.

Step 2: Configure `ulimits`. Remove the default `ulimit` settings that come with the operating system:

```
sudo rm /etc/security/limits.d/90-nproc.conf
```

Edit the `/etc/security/limits.conf` file to configure the following settings:

```
* soft nofile 64000
* hard nofile 64000
* soft nproc 32000
* hard nproc 32000
```

Step 3: Install MongoDB. Use the following series of commands to install MongoDB.

Set up a repository definition by issuing the following command:

```
echo "[MongoDB]
name=MongoDB Repository
baseurl=http://downloads-distro.mongodb.org/repo/redhat/os/x86_64
gpgcheck=0
enabled=1" | sudo tee -a /etc/yum.repos.d/mongodb.repo
```

Install MongoDB by issuing the following two commands:

```
sudo yum install -y mongodb-org mongodb-org-shell
```

Step 4: Create the data directories for the replica set. Create a data directory for each replica set member and set `mongod.mongod` as each data directory's owner.

The following command creates the directory `/data` and then creates a data directory for each member of the replica set. You can use different directory names:

```
sudo mkdir -p /data /data/nodea /data/nodeb /data/nodec
```

The following command sets `mongod.mongod` as owner of the new directories:

```
sudo chown mongod:mongod /data /data/nodea /data/nodeb /data/nodec
```

Step 5: Start a separate MongoDB instance for each replica set member. Start each `mongod` instance on its own dedicated port number and with the data directory you created in the last step. For each instance, specify `mongod` as the user. Start each instance with the replSet command-line option specifying the name of the replica set.

The following three commands start separate instances for each member of a new replica set named `example`:

```
sudo -u mongod mongod --port 27017 --dbpath /data/nodea --replSet example --logpath /data/nodea/mongo

sudo -u mongod mongod --port 27018 --dbpath /data/nodeb --replSet example --logpath /data/nodeb/mongo

sudo -u mongod mongod --port 27019 --dbpath /data/nodec --replSet example --logpath /data/nodec/mongo
```

Step 6: Initiate the replica set. Connect to one of the members and initiate the replica set using the rs.initiate() method. Add the other members using the rs.add() method. The rs.add() method requires the hostame of the server: if necessary, first execute the `hostname` command to determine the name.

Use the following sequence of commands to initiate the replica set and add members. Replace `<hostname>` with the hostname of your server. The commands use the `mongod` running on port `27017` to initiate the set and add the other members.

```
mongo --port 27017 --eval "rs.initiate()";

mongo --port 27017 --eval "rs.add("<hostname>:27018")";

mongo --port 27017 --eval "rs.add("<hostname>:27019")";
```

Step 7: Connect to the replica set and verify the replica set configuration. To connect to the replica set, issue the `mongo` command:

```
mongo
```

To verify the configuration, issue the rs.status() method:

```
rs.status()
```

Verify that the `members` array lists the three members. For a full description of the output, see the explanation in replSetGetStatus.

Step 8: Add data to the replica set. While still connected to the replica set, issue the following `for` loop to create a collection titled `testData` and populate it with 25,000 documents, each with an `_id` field and a field `x` set to a random string.

```
for (var i = 1; i <= 25000; i++) {
  db.testData.insert( { x : Math.random().toString(36).substr(2, 15) } );
  sleep(0.1);
}
```

Step 9: Register the first user. Click the *Register* link and enter the user's information. When you finish, you will be logged into the MMS application.

The registration email will be sent to the address you specified in # `conf-mms.properties`.

For more information on creating and managing users, see *User and Environment Management*.

Step 10: Set up the Monitoring Service for the replica set. On the `Welcome` page, click the *Get Started* button for Monitoring. If the `Welcome` page is not visible, click the *Settings* tab to refresh MMS and then click the *Setup* tab to display the `Welcome` page.

After you click *Get Started*, follow the instructions, which will take you through steps to download, configure, and run the Monitoring Agent.

When the agent is running, the instructions prompt you to add a host. On the *Add a Host* page, enter the hostname and port of the replica set member running on port `27017`. For example, enter `<hostname>:27017`, replacing `<hostname>` with the hostname of the server running the replica set.

When you finish the instructions, the Monitoring Agent is running and monitoring the replica set.

Step 11: Set up the Backup Service for the replica set. On the `Welcome` page, click the *Get Started* button for Backup. If the `Welcome` page is not visible, click the *Settings* tab to refresh MMS and then click the *Setup* tab to display the `Welcome` page.

After you click *Get Started*, follow the instructions to set up the Backup Service. When you reach the *Enable Backup* page, select the `example` replica set, which is the set you created when initializing the replica set members. If you chose a different name when initializing the members, choose that name here instead.

When you finish the instructions, the Backup Agent is running and backing up the replica set. It may take up to 30 minutes for the first snapshot to complete, even for a very small replica set.

While you are waiting, this is a good time to take a tour of your MMS Backup environment.

2.4 Optional: Install On Prem MMS Backup

Install with DEB Packages Install the On Prem MMS Backup software on Ubuntu using `deb` packages..

Install with RPM Packages Install the On Prem MMS Backup software on Red Hat Enterprise, CentOS, Amazon, or SUSE Enterprise Linux using `rpm` packages.

Install On Prem MMS Backup with `deb` Packages

Overview

On Prem Backup provides continuous backup of your data. Use this tutorial to install On Prem MMS Backup on Ubuntu systems.

To arrange a guided install of the MMS Backup components, contact your MongoDB Account Manager. A MongoDB representative can assist you sizing the components and installing the software.

Prerequisites

Please see *On Prem MMS Hardware and Software Requirements* for complete description of the requirements for running On Prem MMS.

Procedures

Install Core Component

Step 1: Download On Prem MMS Monitoring. Download the latest On Prem Monitoring releases from the downloads page.

Step 2: Install On Prem MMS Monitoring. Install the `.deb` package by issuing the following command, where `<version>` is the version of the `.deb` package:

```
sudo dpkg --install mongodb-mms_<version>_x86_64.deb
```

When installed, the base directory for the MMS software is `http://mms.mongodb.com/helpopt/mongodb/mms/`. The `.deb` package creates a new system user `mongodb-mms` under which the server will run.

Step 3: Configure On Prem MMS Monitoring. In the `conf-mms.properties` file, ensure that the following required settings are correct:

- `mms.centralUrl`
- `mms.backupCentralUrl`
- `mms.fromEmailAddr`
- `mms.replyToEmailAddr`
- `mms.adminFromEmailAddr`
- `mms.adminEmailAddr`
- `mms.bounceEmailAddr`
- `mongo.mongoUri`
- `mongo.replicaSet`

Consider the following example configuration:

```
mms.centralUrl=http://<public_ip>:8080
mms.backupCentralUrl=http://<public_ip>:8081

mms.fromEmailAddr=mms-admin@example.net
mms.replyToEmailAddr=mms-admin@example.net
mms.adminFromEmailAddr=mms-admin@example.net
mms.adminEmailAddr=mms-admin@example.net
mms.bounceEmailAddr=mms-admin@example.net

mongo.mongoUri=mongodb://<mms_mongod_ip>:27017/
mongo.replicaSet=rs0
```

At this point, you can also configure authentication, email, and optional Kerberos integration, as described in the *Configuration*.

If you would like to run the MMS application in a highly available configuration, please consider *Configure a Highly Available MMS Application Server*.

Step 4: Start On Prem MMS Monitoring. To start MMS, issue the following command:

```
sudo service mongodb-mms start
```

Install Backup Component

Step 1: Stop any currently running instance. If you are upgrading an existing installation, please stop the currently running instance:

```
sudo service mongodb-mms-backup-daemon stop
```

Step 2: Download the Backup Daemon Package software. To download the Backup Daemon Package for use on Ubuntu, run the following, replacing `<version>` with the software version number:

```
sudo dpkg -i mongodb-mms-backup-daemon_<version>_x86_64.deb
```

The software is installed to `http://mms.mongodb.com/helpopt/mongodb/mms-backup-daemon`.

Step 3: Configure the MMS monitoring application for connection to MMS Backup. Ensure that the MMS Application server is configured to support backup. Edit the `http://mms.mongodb.com/helpopt/mongodb/mms/conf/conf-mms.properties` file and specify the following settings.

- `mongo.backupdb.mongoUri`
- `mongo.backupdb.replicaSet`

Step 4: Configure the back-end software package. Configure the Backup Daemon, by editing the `http://mms.mongodb.com/helpopt/mongodb/mms-backup-daemon/conf/conf-daemon.properties` file. Specify the required configuration options, . See *Configuration* for information about each value.

The following values of the following settings must correspond to the values set in the monitoring application:

- `mongo.mongoUri`
- `mongo.replicaSet`
- `mongo.backupdb.mongoUri`
- `mongo.backupdb.replicaSet`

Consider the following example configuration:

```
mongo.mongoUri=mongodb://<mms_mongod_ip>:27017/
mongo.replicaSet=rs0

mongo.backupdb.mongoUri=mongodb://<backup_mongod_ip>:27017/
mongo.backupdb.replicaSet=rs1
```

Additionally, ensure that the file system that holds the `rootDirectory` has sufficient space to accommodate the current snapshots of all backed up instances.

Step 5: Synchronize the gen.key file Synchronize the `/etc/mongodb-mms/gen.key` file from a MMS Application Server. This is only required if the Backup Daemon Package was installed on a different server then the MMS Application Package.

Step 6: Start the back-end software package. To start the Backup Daemon Package run:

```
sudo service mongodb-mms-backup-daemon start
```

If everything worked the following displays:

```
Start Backup Daemon                                        [  OK  ]
```

If you run into any problems, the log files are at `http://mms.mongodb.com/helpopt/mongodb/mms-backup-daemon/lo`

Install On Prem MMS Backup with `rpm` Packages

Overview

On Prem Backup provides continuous backup of your data. Use this tutorial to install On Prem MMS Backup on RHEL, CentOS, Amazon Linux, or SLES systems.

To arrange a guided install of the MMS Backup components, contact your MongoDB Account Manager. A MongoDB representative can assist you sizing the components and installing the software.

Prerequisites

Please see *On Prem MMS Hardware and Software Requirements* for complete description of the requirements for running On Prem MMS.

Procedures

Install Core Components

Step 1: Download On Prem MMS Monitoring. Download the latest On Prem Monitoring releases from the downloads page.

Step 2: Install On Prem MMS Monitoring. Install the `.rpm` package by issuing the following command, where `<version>` is the version of the `.rpm` package:

```
sudo rpm -ivh mongodb-mms-<version>.x86_64.rpm
```

When installed, the base directory for the MMS software is `http://mms.mongodb.com/helpopt/mongodb/mms/`. The RPM package creates a new system user `mongodb-mms` under which the server runs.

Step 3: Configure On Prem MMS Monitoring. In the `conf-mms.properties` file, ensure that the following required settings are correct:

- `mms.centralUrl`
- `mms.backupCentralUrl`
- `mms.fromEmailAddr`

- mms.replyToEmailAddr
- mms.adminFromEmailAddr
- mms.adminEmailAddr
- mms.bounceEmailAddr
- mongo.mongoUri
- mongo.replicaSet

Consider the following example configuration:

```
mms.centralUrl=http://<public_ip>:8080
mms.backupCentralUrl=http://<public_ip>:8081

mms.fromEmailAddr=mms-admin@example.net
mms.replyToEmailAddr=mms-admin@example.net
mms.adminFromEmailAddr=mms-admin@example.net
mms.adminEmailAddr=mms-admin@example.net
mms.bounceEmailAddr=mms-admin@example.net

mongo.mongoUri=mongodb://<mms_mongod_ip>:27017/
mongo.replicaSet=rs0
```

At this point, you can also configure authentication, email, and optional Kerberos integration, as described in the *Configuration*.

If you would like to run the MMS application in a highly available configuration, please consider *Configure a Highly Available MMS Application Server*.

Step 4: Start On Prem MMS Monitoring. To start MMS, issue the following command:

```
sudo service mongodb-mms start
```

Install Backup Components

Step 1: Stop any currently running instance. If you are upgrading an existing installation, please stop the currently running instance:

```
sudo service mongodb-mms-backup-daemon stop
```

Step 2: Download the Backup Daemon Package software. To download the Backup Daemon Package for use on RHEL, CentOS, Amazon Linux, or SLES, run the following, replacing <version> with the software version number:

```
sudo rpm -U mongodb-mms-backup-daemon-<version>.x86_64.rpm
```

The software is installed to http://mms.mongodb.com/helpopt/mongodb/mms-backup-daemon.

Step 3: Configure the MMS monitoring application for connection to MMS Backup. Ensure that the MMS Application server is configured to support backup. Edit the http://mms.mongodb.com/helpopt/mongodb/mms/conf/conf-mms.properties file and specify the following settings.

- mongo.backupdb.mongoUri

- mongo.backupdb.replicaSet

Step 4: Configure the back-end software package. Configure the Backup Daemon, by editing the http://mms.mongodb.com/helpopt/mongodb/mms-backup-daemon/conf/conf-daemon.properties file. Specify the required configuration options, . See *Configuration* for information about each value.

The following values of the following settings must correspond to the values set in the monitoring application:

- mongo.mongoUri
- mongo.replicaSet
- mongo.backupdb.mongoUri
- mongo.backupdb.replicaSet

Consider the following example configuration:

```
mongo.mongoUri=mongodb://<mms_mongod_ip>:27017/
mongo.replicaSet=rs0

mongo.backupdb.mongoUri=mongodb://<backup_mongod_ip>:27017/
mongo.backupdb.replicaSet=rs1
```

Additionally, ensure that the file system that holds the rootDirectory has sufficient space to accommodate the current snapshots of all backed up instances.

Step 5: Synchronize the gen.key file Synchronize the /etc/mongodb-mms/gen.key file from a MMS Application Server. This is only required if the Backup Daemon Package was installed on a different server then the MMS Application Package.

Step 6: Start the back-end software package. To start the Backup Daemon Package run:

```
sudo service mongodb-mms-backup-daemon start
```

If everything worked the following displays:

```
Start Backup Daemon                                        [  OK  ]
```

If you run into any problems, the log files are at http://mms.mongodb.com/helpopt/mongodb/mms-backup-daemon/lo

2.5 Optional On Prem MMS Configuration

HTTPS with Jetty Configure the Jetty server that runs the core MMS application to use HTTPS.

Application High Availability Outlines the process for achieving a highly available MMS deployment.

Backup High Availability Make the Backup system highly available.

Manage Backup Daemon Jobs Manage job assignments among the backup daemon

Backup Daemon Binaries Configure how MMS downloads versions of MongoDB to manage the backup filesystems.

LDAP Authentication Configure On Prem MMS to use LDAP to store user data and permissions.

Manage Two-Factor Authentication Configure two-factor authentication.

Configure On Prem MMS Monitoring Jetty Instances to use HTTPS

Overview

You can *optionally* configure the Jetty instances that serve the On-Prem MongoDB Management Service application to use HTTPS to encrypt connections between the MMS application and the MMS agent as well as the web interface. Alternately, you can provide access to the MMS application using a load balancer that provides HTTPS access.

Before you can configure On Prem MMS Monitoring Jetty instances to use HTTPS, you must have a valid SSL certificate prepared in the right format. If you do not, this page provides procedures for creating and preparing a valid SSL certificate.

Create and Prepare a Valid SSL Certificate

Create the certificate either through a 3rd-party authority or as a self-signed certificate. If you have an existing certificate, you can use that instead but still must prepare it. Preparing a certificate can involve converting its format and concatenating it with other certificates in a certificate chain.

Use the appropriate procedures in this section to generate and prepare the certificate. To generate certificates must have access to the `openssl` utility.

Create a New Certificate and Signing Request for a 3rd-party Certificate Authority

Step 1: Create a new certificate and certificate signing request (CSR). Issue the following command at the system prompt:

```
openssl req -new -out mms-ssl.csr -newkey rsa:2048 -keyout mms-ssl.key
```

Step 2: Enter answers for the certificate's meta data. `openssl` prompts you to answer questions for the certificate's meta data. Complete all prompts. The *Common Name* **must** have the same hostname value as the `mms.centralUrl` configuration.

Refer to the instructions provided by the certificate authority to ensure that they do not have any more requirements for the certificate signing authority or the certificate meta data.

Step 3: Submit your new CSR to the 3rd-party certificate authority. The certificate authority will return a signed certificate. Each certificate authority may have a different certificate signing procedure.

Create a Self-Signed Certificate

Step 1: Create a self-signed certificate. To generate a self-signed certificate, issue the following command at the system prompt:

```
openssl req -x509 -days 3650 -newkey rsa:2048 -keyout mms-ssl.key -out mms-ssl.crt
```

Step 2: Enter answers for the certificate's meta data. `openssl` prompts for a private key passphrase, and for the answers to questions for the certificate's meta data. Complete all prompts. The *Common Name* **must** have the same hostname value as the `mms.centralUrl` configuration.

Prepare the Certificate as a PEM Certificate

Step 1: If the certificate is in DER format, convert it to PEM. If the signed certificate is in DER format, convert the certificate to PEM format with the following command:

```
openssl x509 -in mms-ssl.cer -inform DER -outform PEM -out mms-ssl.crt
```

Step 2: If the CA uses a certificate chain, concatenate the certificates. If the certificate authority uses a certificate chain, concatenate the certificates together to create a unified certificate, with a command that resembles the following:

```
cat mms-ssl.crt <intermediate-certificate> <root-certificate> > mms-ssl-unified.crt
```

Replace `<intermediate-certificate>` with the intermediate certificate chain and `<root-certificate>` with the certificate authority's root certificate.

Prepare the Certificate as a PKCS12 Certificate

Step 1: Create a PKCS12-formatted keystore. Combine the private key and signed certificate, or certificate chain, into a PKCS12-formatted keystore with the following command:

```
openssl pkcs12 -inkey mms-ssl.key -in mms-ssl-unified.crt -export -out mms-ssl.pkcs12
```

Step 2: Enter answers for the certificate's meta data. `openssl` prompts you to enter the private key passphrase as well as a new passphrase for the PKCS12 keystore.

Configure Jetty Instances to use HTTPS

Once you have created and prepared a valid SSL certificate, use the following sequence of procedures to configure the Jetty instances to use HTTPS to encrypt connections between the MMS application and the MMS agent.

Create Java Truststore

Step 1: Import the PEM certificate into a Java truststore. Import the PEM certificate into a Java truststore, so that the MMS server trusts its own `mms.centralUrl` when making HTTP requests. The default installation directory for the MMS server is `http://mms.mongodb.com/helpopt/mongodb/mms`. If your installation uses a different directory, replace `http://mms.mongodb.com/helpopt/mongodb/mms` with that path.

```
/opt/mongodb/mms/jdk/bin/keytool -import -keystore mms-truststore.jks -file mms-ssl-unified.crt
```

Step 2: Enter a Java keystore passphrase. `keytool` prompts you to specify a Java keystore passphrase. Enter it and type `yes` to confirm import of the certificate.

Create Java Keystore

Step 1: Convert the PKCS12 keystore into a Java Keystore. Convert the PKCS12 keystore into a Java Keystore, so that the MMS server can access the required SSL infrastructure. The default installation directory for the MMS server is `http://mms.mongodb.com/helpopt/mongodb/mms`. If your installation uses a different directory, replace `http://mms.mongodb.com/helpopt/mongodb/mms` with that path.

```
/opt/mongodb/mms/jdk/bin/keytool -importkeystore \
                                 -srckeystore mms-ssl.pkcs12 \
                                 -srcstoretype PKCS12 \
                                 -destkeystore mms-keystore.jks
```

Step 2: Enter the PKCS12 keystore passphrase. You must use the same passphrase for the Java keystore as for the PKCS12 key.

Step 3: Enter a passphrase for the new Java keystore. You must use the same passphrase for the Java keystore as for the PKCS12 key.

Set Truststore and Keystore Location and Permissions

Step 1: Move the Java keystore and truststore files to the `/etc/mongodb-mms` directory. Issue the following command to move the Java keystore and truststore files to the `/etc/mongodb-mms` directory:

```
sudo mv mms-truststore.jks mms-keystore.jks /etc/mongodb-mms/
```

Step 2: Set permissions. Issue the following sequence of commands to set the appropriate permissions on the Java keystore and truststore files. If the MMS application server runs as a different user, change `mongodb-mms` in the `chown` command as needed.

```
sudo chown mongodb-mms:root /etc/mongodb-mms/*.jks
sudo chmod 600 /etc/mongodb-mms/*.jks
```

Generate Credentials

Step 1: Generate a credential pair for the MMS application to use to access the Java Keystore. Issue the following command, replacing `http://mms.mongodb.com/helpopt/mongodb/mms` with the path of the installation directory for the MMS server:

```
/opt/mongodb/mms/bin/credentialstool --username keystore --password
```

`credentialstool` returns output that resembles the following:

```
Your encrypted credentials pair:
Username: abcdef1234567890-76d41ae0a98c
Password: abcdef1234567890-2cc28e525d1f543464
```

Step 2: Copy the credential pair.

Configure MMS Application to use SSL

Step 1: Edit the `mms.conf` file to enable SSL. Edit the `mms.conf` (e.g. `http://mms.mongodb.com/helpopt/mongodb/mms/conf/mms.conf`) file to add the following options:

```
JAVA_MMS_SSL_OPTS="${JAVA_MMS_SSL_OPTS} -Dxgen.webServerSslEnabled=true"
JAVA_MMS_SSL_OPTS="${JAVA_MMS_SSL_OPTS} -Dxgen.webServerSslKeyStorePath=/etc/mongodb-mms/mms-keystore
JAVA_MMS_SSL_OPTS="${JAVA_MMS_SSL_OPTS} -Dxgen.webServerSslKeyStoreEncryptedPassword=abcdef123456789(
```

Step 2: Edit the `conf-mms.properties` file to change the `mms.centralUrl` value to the new HTTPS information. For example:

```
mms.centralUrl=https://mms.example.net:8443
```

Step 3: Configure MMS Application to use SSL. For example:

```
JAVA_MMS_SSL_OPTS="${JAVA_MMS_SSL_OPTS} -Dxgen.webServerSslTrustStorePath=/etc/mongodb-mms/mms-trusts
JAVA_MMS_SSL_OPTS="${JAVA_MMS_SSL_OPTS} -Dxgen.webServerSslTrustStoreEncryptedPassword=f6a5a6b19603c(

JAVA_MMS_SSL_OPTS="${JAVA_MMS_SSL_OPTS} -Dxgen.webServerSslKeyStorePath=/etc/mongodb-mms/mms-keystore
JAVA_MMS_SSL_OPTS="${JAVA_MMS_SSL_OPTS} -Dxgen.webServerSslKeyStoreEncryptedPassword=f6a5a6b19603c0c(
```

Configure the Monitoring Agent to Trust the MMS Server Starting with On Prem MMS 1.4, the Monitoring Agent validates the SSL certificate of the MMS server by default. This means the Monitoring Agent must be configured to trust the MMS server if not using a certificate signed by a trusted 3rd party.

To specify a self-signed certificate of the MMS server that the Monitoring Agent should trust, do the following:

Step 1: Copy your PEM certificate to `/etc/mongodb-mms/`. Issue the following sequence of commands:

```
sudo cp -a mms-ssl-unified.crt /etc/mongodb-mms/
sudo chown mongodb-mms-agent:mongodb-mms-agent /etc/mongodb-mms/mms-ssl-unified.crt
sudo chmod 600 /etc/mongodb-mms/mms-ssl-unified.crt
```

Step 2: Edit the following parameter in `/etc/mongodb-mms/monitoring-agent.config`. For example:

```
sslTrustedMMSServerCertificate=/etc/mongodb-mms/mms-ssl-unified.crt
```

Step 3: Restart the Monitoring Agent for the configuration update to take effect. For example:

```
sudo /etc/init.d/mongodb-mms-monitoring-agent restart
```

Restart MMS Application Server Before you can access MMS using an HTTPS connection you must restart the MMS application server.

Step 1: Restart the MMS application server.

```
sudo /etc/init.d/mongodb-mms start
```

Step 2: You can now connect to MMS by accessing the following URL in a web browser:

`https://mms.example.net:8443`

Configure a Highly Available MMS Application Server

Overview

The On-Prem MMS Application Server provides high availability through horizontal scaling and through use of a *replica sets* for the *backing MongoDB instance* that hosts the *MMS Application Database*.

Horizontal Scaling The MMS Application Servers are stateless between requests. Any server can handle requests as long as all the servers read from the same backing MongoDB instance. If one Application Server becomes unavailable, another fills requests.

To take advantage of this for high availability, configure a load balancer to balance between the pool of MMS Application Servers. Use the load balancer of your choice. Configure each Application Server's `conf-mms.properties` file to point its `mms.centralUrl` property to the load balancer. The load balancer then manages the *MMS HTTP Service* and *Backup HTTP Service* that runs on each Application Server.

Replica Set for the Backing Instance Deploy a *replica set* rather than a standalone as the *backing MongoDB instance* for monitoring. Replica sets have automatic failover if the *primary* becomes unavailable.

When deploying a replica set with members in multiple facilities, ensure that a single facility has enough votes to elect a *primary* if needed. Choose the facility that hosts the core application systems. Place a majority of voting members and all the members that can become primary in this facility. Otherwise, network partitions could prevent the set from being able to form a majority. For details on how replica sets elect primaries, see Replica Set Elections.

You can create backups the replica set using file system snapshots. File system snapshots use system-level tools to create copies of the device that holds replica set's data files.

Prerequisites

Deploy a replica set for the *backing instance* for the *MMS Application Database*. To deploy a replica set, see Deploy a Replica Set.

Procedure

To configure multiple Application Servers with load balancing:

Step 1: Configure a load balancer with the pool of MMS Application Servers. This configuration depends on the general configuration of your load balancer and environment.

Step 2: Update each MMS Application Server with the load balanced URL. On each Application Server, edit the `conf-mms.properties` file to configure the `mms.centralUrl` property to point to the load balancer URL.

The `conf-mms.properties` file is located in the `<install_dir>/conf/` directory.

Step 3: Update each MMS Application Server with the replication hosts information. On each Application Server, edit the `conf-mms.properties` file to define the replication hosts used as the *backing MongoDB instances*.

Set the `mongo.mongoUri` property to the connection string of the backing instance used for monitoring data. For example:

```
mongo.mongoUri=mongodb://<mms0.example.net>:<27017>,<mms1.example.net>:<27017>,<mms2.example.net>:<27
mongo.replicaSet=<mmsReplSet0>
```

If you use On Prem MMS Backup, set the `mongo.backupdb.mongoUri` property to the connection string of the the backing instance used for backup. For example:

```
mongo.backupdb.mongoUri=mongodb://<mms3.example.net>:<27017>,<mms4.example.net>:<27017>,<mms5.example
mongo.backupdb.replicaSet=<mmsBackupReplSet0>
```

Step 4: Synchronize the `gen.key` file across all the MMS Application Servers. Synchronize the `/etc/mongodb-mms/gen.key` file across all Application Servers. The MMS Application Server uses this file to encrypt sensitive information before storing the data in a database.

Additional Information

For information on making MMS Backup highly available, see *Configure a Highly Available MMS Backup Service*.

Configure a Highly Available MMS Backup Service

Overview

The *Backup Daemon* maintains copies of the data from your backed up `mongod` instances, and creates snapshots used for restoring data. The file system that the Backup Daemon uses must have sufficient disk space and write capacity to store the backed up instances.

For replica sets, the local copy is equivalent to an additional secondary replica set member. For sharded clusters the daemon maintains a local copy of each shard as well as a copy of the *config database*.

Multiple Backup Daemons To increase your storage and to scale horizontally, you can run multiple instances of the Backup Daemon. With multiple daemons,, MMS binds each backed up replica set or shard to a particular Backup Daemon. For example, if you run two Backup Daemons for a cluster that has three shards, and if MMS binds two shards to the first daemon, then that daemon's server replicates only the data of those two shards. The server running the second daemon replicates the data of the remaining shard.

Multiple daemons allow for **manual** failover should the daemon become unavailable. You can instruct MMS to transfer the daemon's backup responsibilities to another Backup Daemon. MMS reconstructs the data on the new daemon's server and binds the associated replica sets or shards to the new daemon. See *Move Jobs from a Lost Backup Service to another Backup Service* for a description of this process.

MMS reconstructs the data using a snapshot and the oplog from the *Backup Blockstore Database*. To deploy Backup Daemons, see *Optional: Install On Prem MMS Backup*.

Replica Set for the Backing Instance Deploy a *replica set* rather than a standalone as the *backing MongoDB instance* for backup. Replica sets have automatic failover if the *primary* becomes unavailable.

When deploying a replica set with members in multiple facilities, ensure that a single facility has enough votes to elect a *primary* if needed. Choose the facility that hosts the core application systems. Place a majority of voting members

and all the members that can become primary in this facility. Otherwise, network partitions could prevent the set from being able to form a majority. For details on how replica sets elect primaries, see Replica Set Elections.

To deploy a replica set, see Deploy a Replica Set.

Additional Information

To move jobs from a lost Backup server to another Backup server, see *Move Jobs from a Lost Backup Service to another Backup Service*.

For information on making the MMS Application Server highly available, see *Configure a Highly Available MMS Application Server*.

Move Jobs from a Lost Backup Service to another Backup Service

Overview

If the server running a *Backup Daemon* fails, and if you *run multiple Backup Daemons*, then an administrator with the `global owner` or `global backup admin` *role* can move all the daemon's jobs to another Backup Daemon. The new daemon takes over the responsibility to back up the associated *shards* and *replica sets*.

When you move jobs, the destination daemon reconstructs the data using a snapshot and the *oplog* from the *Backup Blockstore Database*. Reconstruction of data takes time, depending on the size of the databases on the source.

During the time it takes to reconstruct the data and reassign the backups to the new Backup Daemon:

- MMS Backup does not take new snapshots of the jobs that are moving until the move is complete. Jobs that are not moving are not affected.
- MMS Backup *does* save incoming oplog data. Once the jobs are on the new Backup Daemon's server, MMS Backup takes the missed snapshots at the regular snapshot intervals.
- Restores of previous snapshots are still available.
- MMS can produce restore artifacts using existing snapshots with *point-in-time recovery* for *replica sets* or *checkpoints* for *sharded clusters*.

Procedure

With administrative privileges, you can move jobs between Backup daemons using the following procedure:

Step 1: Click the *Admin* link at the top of MMS.

Step 2: Select *Backup* and then select *Daemons*. The *Daemons page* lists all active Backup Daemons.

Step 3: Locate the failed Backup Daemon and click the *Move all heads* link. MMS displays a drop-down list from which to choose the destination daemon. The list displays only those daemons with more free space than there is used space on the source daemon.

Step 4: Move the the jobs to the new daemon. Select the destination daemon and click the *Move all heads* button.

Optional LDAP Authentication

Overview

On-Prem MongoDB Management Service can use a Lightweight Directory Access Protocol (LDAP) service to store users and manage user authentication. Users continue to log in using the standard MMS interface. After successful LDAP authentication, MMS synchronizes the `firstName`, `lastName`, and `email` attributes in a LDAP user record with their MMS user record. Integration requires adding LDAP user record values in the MMS configuration file.

Upon submission of login form data, MMS authenticates in two steps:

1. First, MMS searches the LDAP server for a matching LDAP user record with the MMS `mms.ldap.bindDn` and `mms.ldap.bindPassword` configuration values to match the username.
2. With a match, MMS searches the LDAP server with the user record attribute defined for the `mms.ldap.user.searchAttribute` parameter to authenticate for MMS access.

Upon successful login with LDAP, the first user completes a welcome form to create the initial MMS group.

Prerequisites

- LDAP server installed, configured, and accessible to MMS.
- An LDAP group name used to populate the `mms.ldap.global.role.owner` configuration value used to match LDAP records with MMS data.
- MMS server installed and configured.

It's also possible to create LDAP groups to assign to users with read only or other non-administrative roles, then update the global roles property settings as needed. In this case, create one or more additional LDAP groups.

Considerations

For successful integration, each user record sent by the LDAP server must contain the list of LDAP groups assigned to the user.

The first user to login with LDAP authentication must have the LDAP Owner role assigned to their account. The `mms.ldap.global.role.owner` property setting in the MMS configuration file must match an LDAP owner group.

For example, if LDAP has an `admin` group for use by MMS admins, set the `mms.ldap.global.role.owner` property to `admin` in the MMS configuration file.

There is no method to transition an existing MMS deployment with independent user management to use LDAP for user management. You will need to need to start over with a fresh installation of the latest version of MMS.

Procedure

1. Define LDAP record schema attributes and values.
2. Update the LDAP server configuration values in the MMS `conf-mms.properties` file.
3. Update the LDAP user configuration values in the MMS `conf-mms.properties` file.
4. Update the Global Role configuration values in the MMS `conf-mms.properties` file.

The sections below define configuration values to update for each step.

Configuration Files

Configuration parameters connect one or more LDAP groups to roles used in MMS, as well as retrieve user data to authenticate users.

LDAP Server Configuration Parameter Update this LDAP server property in the MMS `conf-mms.properties` configuration file.

Property	Value
`mms.userSvcClass`	`com.xgen.svc.mms.svc.user.UserSvcLdap`

LDAP User Configuration Parameters Update these LDAP directory schema properties in the MMS `conf-mms.properties` configuration file:

Property	Example	Description
mms.ldap.url	`ldap://174.129.1.15:389`	The URI for the LDAP server
mms.ldap.bindDn	`cn_search_`	The LDAP user used to execute searches for other users
mms.ldap.bindPassword	`dI8OrFFnj7WMmc`	The credentials for the search user
mms.ldap.user.baseDn	`ou\=users,d\=idm,o\=xyz`	The base dn used for searching for users
mms.ldap.user.searchAttribute	`cn`	The LDAP user record attribute MMS uses to search then authenticate users when a user types their username into the MMS login form.
mms.ldap.user.firstName	`givenName`	The LDAP user attribute that contains the user's first name.
mms.ldap.user.lastName	`sn`	The LDAP user attribute that contains the user's last name.
mms.ldap.user.email	`mail`	The LDAP user attribute that contains the user's email address.
mms.ldap.user.group	`memberOf`	The LDAP user attribute that contains the list of groups that the user belongs to. These can be `cn s` or `dn s`. It doesn't matter as long as they are consistent with those provided in the MMS global role configuration, explained below.

Global Role Configuration Global parameters can be in any format for an LDAP group. They can be a `cn` (Common Name) or a `dn` (Distinguished Name). The format must match the property specified by the `mms.ldap.user.group` configuration property defined in the table above.

Update these LDAP directory schema properties in the MMS `conf-mms.properties` configuration file:

Property	Example	Description
mms.ldap.global.role.readOnly	`readOnlyLdapGroup`	The LDAP group attribute name for users assigned the global read-only role in MMS. This role can only view data in MMS.
mms.ldap.global.role.monitoringAdmin	`monitoringLdapGroup`	The LDAP group attribute name for users assigned the global monitoring administrative role in MMS. This role can view hosts, charts, and other data, as well as monitor hosts, manage monitoring settings, download the Monitoring Agent, and other tasks.
mms.ldap.global.role.backupAdmin	`backupLdapGroup`	The LDAP group attribute name for users assigned the global backup administrative role in MMS. This role can view backup status, snapshot lists, and modify backup settings, as well as start/stop/terminate backups, request restores, view/edit host passwords, and other tasks.
mms.ldap.global.role.owner	`ownerLdapGroup`	The LDAP group attribute name for users assigned the global owner role in MMS. This role can perform all administrative tasks in MMS.

Manage Two-Factor Authentication for On Prem MMS

Overview

When enabled, two-factor authentication requires a user to enter a verification code to log in and to perform certain protected operations. Operations that require two-factor authentication include:

- restoring and deleting snapshots,
- stopping and terminating Backup for a *sharded cluster* or *replica set*,
- inviting and adding users,
- generating new two-factor authentication backup codes, and
- saving phone numbers for two-factor authentication.

Administrators with access to the MMS Application's `<install_dir>/conf/conf-mms.properties` file on your servers can enable two-factor authentication through the file's `mms.multiFactorAuth.require` setting. Administrators can also enable two-factor authentication to use Twilio, through the file's *Twilio settings*.

Users configure two-factor authentication on their accounts through their *MMS user profiles*, where they select whether to receive their verification codes through voice calls, text messages (SMS), or the Google Authenticator application. If your organization does not use Twilio, then users can receive codes only through Google Authenticator.

Administrators can reset accounts for individual users as needed. Reseting a user's account clears out the user's existing settings for two-factor authentication. When the user next performs an action that requires verification, MMS forces the user to re-enter settings for two-factor authentication.

Procedures

Enable Two-factor Authentication

Step 1: Open the MMS Application Server's `conf-mms.properties` file. The `conf-mms.properties` file is located in the `<install_dir>/conf/` directory.

Step 2: Set the `mms.multiFactorAuth.require` property to `true`.

```
mms.multiFactorAuth.require=true
```

Step 3: Restart the MMS Application.

```
sudo service mongodb-mms start
```

Enable Twilio Integration

Reset a User's Two-factor Authentication Account Reseting the user's account clears out any existing two-factor authentication information. The user will be forced to set it up again at the next login.

You must have the `global user admin` or `global owner` *role* to perform this procedure.

Step 1: Open On Prem MMS Administration. To open Administration, click the *Admin* link in the On Prem MMS banner.

Step 2: Select the *Users* page.

Step 3: Locate the user and click the pencil icon on the user's line.

Step 4: Select the *Clear Two Factor Auth* checkbox.

2.6 Start and Stop MMS Application

Start the On Prem MMS Server

Note: If you installed from a `tar.gz` or `zip` archive, you must create a symlink located at the path `/etc/init.d/mongodb-mms` that points to the `<install_dir>/bin/mongodb-mms`.

After configuring your On Prem MMS Monitoring deployment, you can start the MMS server with this command:

```
sudo /etc/init.d/mongodb-mms start
```

In some situations, starting MongoDB *may* take several minutes to pre-allocate the journal files. This is normal behavior.

You can now use the On Prem MMS Monitoring instance by visiting the URL specified in the `mms.centralUrl` parameter (e.g. http://mms.example.com:8080) to continue configuration:

Unlike the SaaS version of MMS, On Prem MMS Monitoring stores user accounts in the local MongoDB instance. When you sign into the On Prem MMS Monitoring instance for the first time, the system will prompt you to register and create a new "group" for your deployment.

After completing the registration process, you will arrive at the "MMS Hosts," page.

Because there are no Monitoring Agents attached to your account, the first page you see in On Prem MMS Monitoring will provide instructions for downloading the Monitoring Agent. Click the "download agent" link to download a pre-configured agent for your account. Continue reading this document for installation and configuration instructions for the MMS agent.

Stop the On Prem MMS Server

Enter the following command:

```
sudo /etc/init.d/mongodb-mms stop
```

Startup Log File Output

The On Prem MMS server logs its output to a `logs` directory inside the installation directory. You can view this log information with the following command:

```
sudo less <install_dir>/logs/mms0.log
```

If the server starts successfully, you will see content in this file that resembles the following:

```
[main] INFO  ServerMain:202 - Starting mms...
[main] WARN  AbstractConnector:294 - Acceptors should be <=2*availableProcessors: SelectChannelConnec
[null] LoginService=HashLoginService identityService=org.eclipse.jetty.security.DefaultIdentityServic
[main] INFO  AppConfig:46 - Starting app for env: hosted
```

```
[main] INFO  MmsAppConfig:67 - Not loading backup components
[main] INFO  GraphiteSvcImpl:67 - Graphite service not configured, events will be ignored.
[main] INFO  TwilioSvcImpl:48 - Twilio service not configured, SMS events will be ignored.
[main] INFO  OpenDMKSnmpTrapAgentSvcImpl:91 - SNMP heartbeats hosts not configured, no heartbeat trap
[main] INFO  ServerMain:266 - Started mms in: 24979 (ms)
```

Optional: Run as Different User

1. Edit `<install_dir>/conf/mms.conf`:

   ```
   MMS_USER=foo_user
   ```

2. Change Ownership of `<install_dir>` for new user:

   ```
   sudo chown -R foo_user:foo_group <install_dir>
   ```

3. Restart MMS server:

   ```
   sudo <install_dir>/bin/mongodb-mms restart
   ```

Optional: MMS Application Server Port Number

1. Edit `<install_dir>/conf/conf-mms.properties`:

   ```
   mms.centralUrl=http://mms.acmewidgets.com:<newport>
   ```

2. Edit `<install_dir>/conf/mms.conf`

   ```
   BASE_PORT=<newport>
   ```

3. Restart MMS server:

   ```
   sudo <install_dir>/bin/mongodb-mms restart
   ```

3 On Prem MMS Administration

Administration Interface Lists and explains the MMS *Administration* page.

Application Settings Lists and explains the MMS *Settings* page.

Two-Factor Authentication How to use and manage two-factor authentication.

User and Environment Management Details how to manage users and groups in MMS.

Manage Events Events, alerts, and related configuration.

Create an Alert Configuration Outlines procedures for creating alerts in MMS.

Manage Alerts Outlines procedures for manipulating alerts in MMS.

Backup Alerts Describes alert messages concerning the MMS Backup system.

Configure Kerberos Authentication Configure the MMS application to connect to MongoDB using the Kerberos authentication mechanism.

MMS Public API Principles An overview of the MMS HTTP API.

Use MMS Public API An introduction to using the HTTP API for MMS.

3.1 Administration Interface

The *MMS Administration* section of the On-Prem MongoDB Management Service application provides access to user management, system status, and system-wide messaging.

Overview

On-Prem MongoDB Management Service includes a tool to view system status and perform administrative tasks, for example, set cron jobs and manage users and groups.

If you have administrative privileges for the MMS deployment, click the *Admin* link in the top right corner of MMS to access the system administration tool.

The *Admin* page includes the following sections and tabs:

- The default tab is the *Overview* tab which provides a system overview. *Messages*, Reports, *Users*, and *Groups* tabs provide additional functionality.
- The *Backup section* includes tabs for Jobs, Logs, Daemons, Blockstores, Grooms, and Job Timeline.
- The *Control Panel section* includes tabs for *Send Test Email* and *Send Test SMS* pages.

General Tabs

Overview

The *Overview* tab provides reports on system use and activity.

This tab displays the summary table and *Total Pages Viewed* and *Total Chart Requests* charts.

A summary table reports totals for:

- groups
- active groups
- active hosts
- users
- users active
- ping spillover queue
- increment spillover queue

Additionally two charts report:

- total page views
- total chart requests

Messages

You can display a short message on any page of the MMS application to notify or remind users, such as impending maintenance windows. Messages may be *active*, i.e. visible, on all pages or a subset of pages.

The *Messages* tab provides information on existing messages as well as an interface to add new messages or manage existing messages.

Message Table The *UI Messages* page holds a table of all available messages. Use the search box on the top right to filter the list of messages.

For each message, the message table reports:

- which page or page prefix the message will appear on.
- the text of the message.
- whether the message is active or inactive. Active messages are also highlighted in orange.
- the creation date and time for the message.
- the date and time of the last modification for the message.

Add a Message To add a message which appears on one page, or on all pages,

1. Click the:guilabel:*Add Message* button.
2. In the *Add Message* interface, enter
   - the message, and
   - the page URL or optionally, page prefix in the *Add Message* interface.

     The page prefix allows you to specify a path of a single page or the URL prefix of a group of pages. The prefix must begin with a `http://mms.mongodb.com/help` character.

     For example, entering the page prefix `http://mms.mongodb.com/helpsettings/profile` will display a message on the default *Settings* page and *Profile* tab but not on any other page or tab in the application.
3. Click the *Active* checkbox to make the message live. Optionally, you can leave the box unchecked to disable the message.
4. Click the *Add* button to add the message.

Once added, active messages take 60 seconds before they display.

Disable a Message To disable a message, click the orange square button to the right of any alert listed on the *Messages* tab. The orange button will change to a grey button.

To re-enable a disabled message, click the grey button; the grey button will change to back to an orange button.

Delete a Message To delete a message, click the garbage can icon to the right of any alert listed on the *Messages* tab.

Users

The *Users* tab displays a list of user information for all people with permission to use the On-Prem MongoDB Management Service application as well as provides an interface to manage users. Use the search box on the upper right corner to find a user record.

Users Information For each user, the users table reports:

- the username
- the available administrative roles, if any
- the date and time of the last login.

- the date and time of the last access event.
- the total number of login.
- the user's configured timezone.
- the creation date and time.

Edit a User Record To edit a user record,

1. Click the pencil icon at the far right of the user record.
2. In the *Edit User* interface, you can:
    - Edit the user's email.
    - Select the user's groups and roles.
    - Lock or unlock the account.
    - Specify the user's global roles.
3. When finished, click the *Save* button to save any changes and exit.

Groups

The *Groups* tab lists all groups created with their date created and the last date and time an agent for the group pinged MMS. At the top right of the page is a search box to find a group by name.

Click a group name to go directly to the group pages in MMS. To return to the system administration tool, click the browser back button or click the *Admin* link in the top right of any page.

Backup Tabs

Jobs

You can see all active and stopped Backup jobs on the *Jobs* tab. For each backup job, the tab lists job group, name, last agent conf, last oplog, head time, last snapshot, what the agent is working on, the daemon, and the blockstore.

Click the group name to go directly to the backup job page for the group. To return to the system administration tool, click the browser back button or click the *Admin* link in the top right of any page.

Click the job name to see the results of the job on a detail page. On the job detail page, click links to see the logs, resource usage, conf call, and download a zipped archive file containing complete diagnostic information for the specified job.

Logs

The *Logs* tab lists backup logs by job and class with messages grouped and counted by last 2 hours, last day, and last 3 days. Click a count number to see all messages in the group.

Daemons

This tab lists all active backup daemons, as well as the ability to pre-configure a new Backup Daemon.

For each Backup Daemon, this tab lists the server name, configuration, head space used, head space free, the number of replica sets backed up, the percentage of time the Backup Daemon Service was busy, and job runtime counts by 1 minute, 10 minutes, 60 minutes, less than or equal to 180 minutes, and greater than 180 minutes.

Click the *Show JSON* link to view the Backup Daemon JSON. When JSON displays, click the *View raw runtime data* link under the code to view raw data. To hide the daemon JSON, click the *Hide JSON* link.

Click the *Move all heads* link to move the Backup Daemon head location. Select the location and click the *Move all heads* button to move all jobs that live on this daemon to a new daemon.

For each Backup Daemon, you can select or deselect the assignment, backup jobs, restore jobs, resource usage, and garbage collection configuration, as well as indicate whether the disk type is SSD or HDD. If selected, the algorithm that assigns jobs to daemons uses disk type to assign jobs with high oplog churn to HHD daemons.

To pre-configure a new Backup Daemon, scroll to the bottom of this tab and type the `<machine>:<root head path>` in the text field above the *Pre-Configure New Daemon* button. Click the button to pre-configure the new Backup Daemon.

Blockstores

The *Blockstores* tab lists all backup blockstores with the ability to add a new blockstore.

To update an existing blockstore, edit the `<hostname>:<port>`, *MongoDD Auth Username*, and *MongoDB Auth Password*. Click the *Encrypted Credentials* checkbox if the username and password entered above used the `credentialstool` for encryption.

To add a blockstore, enter the `<id>` and `<hostname:port>` in the text fields above the *Add New Blockstore* button then click the button to save and add the blockstore.

For all MongoDB connections, enter a single `<hostname>:<port>` for a MongoDB standalone or a comma separated list of all replica set members. A newly configured blockstore only becomes active when you select the assignment checkbox.

Grooms

This tab lists all active backup groom jobs, bytes saved from recent groom jobs, and recently finished groom jobs. A groom is the process of garbage collecting dead bytes from the Blockstore Database. *Dead* blocks are blocks no longer referenced by a non-expired snapshot and, therefore, eligible for garbage collection.

For active and recently finished backup groom jobs, the tab lists the group, job name, savings, running time, stage, total blocks, scanned blocks as a percent of total for active jobs, copied blocks as a percent of total for finished jobs, direction, blockstore, and machine.

Job Timeline

The *Job Timeline* tab displays the present and future scheduled backup status for active jobs, as well as the ability to search backup jobs by group, replica set, and machine.

To update the list automatically every 10 seconds, click the *Auto refresh* checkbox at the top left of the list. Or click the *Refresh* button to refresh data manually as needed.

To view the backup job JSON, click the *Show JSON* link under the *Group* heading for any backup job. When JSON displays, click the *View raw runtime data* link under the code to view raw data. To hide the daemon JSON, click the *Hide JSON* link.

To move the job to a new Backup Daemon, click the *Move head* link under the *Machine* heading for a backup job. Select the location and click the *Move head* button to move the job to a new Backup Daemon.

You can bind a backup job to a head by clicking the Set binding button under the *Machine* heading for any backup job.

To break the backup job, click the *Break Job* link under the *Machine* heading for any backup job. To unbreak the backup job, click the *Unbreak Job* link.

Control Panel Tabs

Send Test Email

Use this tab to send a test email with MMS. Fill out the *To* and *Body* message then click the *Send Email* button.

Send Test SMS

Use this tab to send an SMS message. Fill out the *To* and *Body* message then click the *Send Email* button.

3.2 Application Settings

The "Settings" section of the On-Prem MongoDB Management Service (MMS) console enables users to personalize their console and activate or deactivate a variety of features. The following sections correspond to a tab on the *Settings* page.

My Settings

These settings are specific to the logged in user, and will only affect their MMS experience.

Profile

The *Profile* page allows users to update their personal information.

The username, email address, and password are also used for jira.mongodb.org. Changing your email address or password in MMS will also change the email address and password you use to log into Jira.

- **User Name**: displays the user's name. You cannot change your username.
- **Email Address**: displays the email address MMS associates with your account. You can change your email address by clicking on the "*pencil*" icon.
- **Password**: allows you to change your MMS password. Passwords must fulfill MMS's *password requirements*.
- **Two-Factor Authentication**: MMS requires two factor authentication for login. For details, see *Two-Factor Authentication*.

  To delete or reset two-factor authentication, contact your MMS system administrator.

Personalization

The *Personalization* page allows users to configure the console to suit their needs and preferences.

- **My Dashboard**: sets the default dashboard on the "Dashboard" page. You can select from a list of all of your dashboards.
- **My Time Zone**: sets your local time zone.
- **My Date Format**: allows you to select your preferred date format.

- **Page Shown When Switching Groups** sets which page of the MMS console you will see when you select a different group. You can select the "Hosts" page or the "Dashboard" page. Alternatively, select "Current" and MMS will not change pages when you select a different group.
- **Display Opcounters On Separate Charts**: allows you to control the presentation of Opcounter Charts. If enabled, MMS charts each opcounter type separately. Otherwise, each opcounter type is overlaid together in a single chart.
- **Display Chart Annotations**: toggles the presence of chart annotations. Chart annotations overlay information about significant system events on the charts. For example, with chart annotations MMS will draw a red vertical line over the charts.
- **Email Notifications**: allows you to opt-in to, or opt-out of receiving e-mail newsletters about MMS.

Group Settings

These settings are general settings that apply to all users in the current group.

Group Settings

- **Group Time Zone**: sets your group's time zone.
- **Collect Logs For All Hosts**: activates or deactivates the collection of log data for all hosts. This overwrites the statuses set on the individual hosts. On Prem MMS Monitoring displays log data in the "Logs" tab of the "Host Statistics" page.
- **Collect Profiling Information for All Hosts**: activates or deactivates the collection of profiling information for all hosts. MMS Monitoring can collect data from MongoDB's profiler to provide statistics about performance and database operations. Ensure exposing profile data to MMS Monitoring is consistent with your information security practices. Also be aware the profiler can consume resources which may adversely affect MongoDB performance.
- **Collect Database Specific Statistics**: allows you to enable or disable the collection of database statistics. For more information, see "*How does MMS gather database statistics?*".
- **Monitoring Agent's Log Level**: allows adjustment of the log level of the Monitoring Agent from the settings menu without restarting the agent. The *Default* level maintains the level specified when starting the Monitoring Agent. Setting the log level on the agent requires Monitoring Agent version 2.0.0+.
- **Preferred Hostnames**: allows you to specify the hostname to use for servers with multiple aliases. This prevents servers from appearing multiple times under different names. By default, the Monitoring Agent tries to connect by resolving hostnames. If the agent cannot connect by resolving a hostname, you can force the Monitoring Agent to prefer an IP address over its corresponding hostname for a specific IP address. To override this default behavior, set an IP address as a preferred hostname. If your IP addresses have a common prefix, create a preferred hostname with the *ends-with* button or click the *regexp* button to use a regular expression.
- **Reset Duplicates**: allows you to reset and remove all detected duplicate hosts. This is useful if your server environment has drastically changed and you believe a host is incorrectly marked as a duplicate.

API Settings

API Settings displays the On Prem MMS Monitoring API Key for your MMS group. Keep this key private. Use the API key to support automated installation of your Monitoring Agent with scripts included with the agent installation files.

Group Alerts

Group Alerts defines the default settings for HipChat, PagerDuty, and other third-party groups used to send alerts. See *Activity Alert Settings page* page for details about using these third-party services to manage alerts.

Agents

Monitoring Agent

Monitoring Agent provides links for downloading the pre-configured Monitoring Agent in both `.zip` and `.tar.gz` formats.

On Prem MMS Backup Settings

The following settings will only be visible to users of On Prem MMS Backup.

Backup Agent

Backup Agent provides links for downloading the pre-configured Backup Agent in both `.zip` and `.tar.gz` formats, on a variety of platforms. *On Prem MMS Backup* requires this agent.

Public Key for SCP Restores

Public Key for SCP Restores, or "*MMS Backup Public Key*" enables users who have signed up for *On Prem MMS Backup* to generate a new public key that MMS will use to connect via SSH and transmit a snapshot of your data.

Backup Agent

Backup Agent provides links to download the pre-configured Backup Agent in both `.zip` and `.tar.gz` formats. The software is dynamically assembled with your API key. Instructions are included to set up and start the Backup Agent, as well as create a new user for the agent if MongoDB authentication is used.

Backup

Restore Settings

Restore Settings provides the ability to generate a public key for SCP backup restoration through the Backup and Restore Service.

3.3 Two-Factor Authentication

Overview

When enabled, MMS requires two-factor authentication to help users control access to their MMS accounts. To log into MMS, a user must provide their password (i.e. "something you know"), as well as a second time-sensitive verification code, delivered during authentication (i.e. "something you have"). By requiring both factors, MMS can grant authentication requests with a higher degree of confidence.

MMS users receive verification codes through text messages (SMS), automated voice calls or an application that implements the Time-based One-time Password Algorithm (TOTP), such as the Google Authenticator application. Users can configure two-factor authentication mechanisms when signing up for MMS or in the *Settings* section of the MMS application.

Authentication with Text or Voice Messages

Users can receive verification codes through text or voice by providing phone numbers when setting up their MMS profiles. When a user needs a code, MMS sends the code using text (SMS) or through an automated phone call that reads out the code.

Certain network providers and countries may impose delays on SMS messages. Users who experience delays should consider Google Authenticator for verification codes.

Note: From India, use Google Authenticator for two-factor authentication. Google Authenticator is more reliable than authentication with SMS text messages to Indian mobile phone numbers (i.e. country code 91).

Authentication with Applications

Authentication using Google Authenticator Google Authenticator is a smartphone application that uses TOTP to generate verification codes. When a user needs a code, the application generates a time-based code based on a private key that was shared between MMS and the user's Google Authenticator application during the initial pairing process.

The Google Authenticator application **does not** require a Google account and does not connect a user's MMS account to Google in any way. The has both iOS and Android versions, and the user does not need to associate the application with a Google account. MMS two-factor authentication using Google Authenticator is not in any way integrated with Google's own account authentication mechanisms, and MMS does **not** provide two-factor authentication codes to Google.

Authentication using Another Implementation of TOTP There are implementations of the Time-based One-time Password Algorithm (TOTP) other than Google Authenticator. For example, the Authenticator application for Windows Phones.

Ensure that whichever devices runs the TOTP application has it's own set of robust authentication requirements.

For other implementations of TOTP, consider the list of TOTP implementations on Wikipedia.

Two-Factor Authentication on a Shared Account

A global team that shares the same MMS account can use Google Authenticator and use the same seed code for all team members. To generate a common seed code that all team members can use, select the *Can't scan the barcode?* link when *Configuring Two-Factor Authentication with Google Authenticator*.

Procedures

To enable or disable two-factor authentication for the entire On-Prem MongoDB Management Service environment, see *Manage Two-Factor Authentication for On Prem MMS*.

Configure Two-Factor Authentication with Text or Voice

Step 1: In MMS, select the *Settings* tab and then *Profile*.

Step 2: Select the pencil icon for *Two Factor Authentication*. Or, if this is the first time you are setting up an account, click the *Configure* button to the right side of the *Profile* page and follow the instructions.

Step 3: Select *Use Voice/SMS*.

Step 4: Enter the phone number for the phone that will receive the codes. If you are outside of the United States or Canada, you must include `011` and your country code. Alternatively, you can sign up for a Google Voice number and use that number for your authentication.

Step 5: Select how to receive the codes. Select either *Text message (SMS)* or *Voice call (US/Canada only)*.

Step 6: Click *Send Code*. MMS sends the codes to your phone.

Step 7: Enter the code in the box provided in MMS and click *Verify*.

Step 8: Click *Save Changes*.

Configure Two-Factor Authentication with Google Authenticator

Step 1: Install Google Authenticator from either the Google Play store or the iOS Apple Store, depending on your device.

Step 2: Run Google Authenticator.

Step 3: Click `Begin Setup`.

Step 4: When prompted, select how you will enter the shared private key. Under `Manually Add an Account`, select either `Scan a barcode` or `Enter provided key`. Stay on this screen while you use the next steps to access the barcode or key in MMS.

Step 5: In MMS, select the *Settings* tab and then *Profile*.

Step 6: Select the pencil icon for *Two Factor Authentication*. Or, if this is the first time you are setting up an account, click the *Configure* button to the right side of the *Profile* page and follow the instructions.

Step 7: Select *Use Google Authenticator*. MMS provides a barcode and a *Can't scan the barcode?* link.

Step 8: Scan or enter the shared private key. If your smartphone can scan barcodes, then scan the barcode. Otherwise, click *Can't scan the barcode?* and type the provided *Key* into your smartphone.

Step 9: Enter the Google Authenticator code in MMS. After you scan the barcode or enter the key, Google Authenticator generates a 6-digit code. Enter that in the box provided in MMS and click *Verify*.

Step 10: Click *Save Changes*.

Generate New Recovery Codes

As a backup, you can generate recovery codes to use in place of a sent code when you do not have access to a phone or your Google Authenticator application. Each recovery code is single-use, and you should save these codes in a secure place. When you generate new recovery codes, you invalidate previously generated ones.

Step 1: In MMS, select the *Settings* tab and then *Profile*.

Step 2: Select the pencil icon for *Two Factor Authentication*. Or, if this is the first time you are setting up an account, click the *Configure* button to the right side of the *Profile* page and follow the instructions.

Step 3: Select *Generate New Recovery Codes*. Keep the codes in a safe place. Each code can be used in conjunction with your username and password to not only access MMS but to reset your security settings on MMS.

3.4 User and Environment Management

You can manage the users that have access to your On-Prem MongoDB Management Service groups, create and manage groups, and assign roles to users to provide controlled access to the MMS application.

There is no planned upgrade path from existing MMS user authentication to using LDAP. You will need to recreate users, groups, and roles manually with your LDAP service, as described in the *Optional LDAP Authentication* document.

User Management

View Users

To view users, click the *Users* tab and then select the *Users* page. The *Users* page lists users who have access to your MMS group, their roles, their time zones, and other information.

Add Users

Note: With MongoDB Management Service On Prem, user accounts and groups are independent from JIRA. This is in contrast to the MongoDB Management Service, which shares account and group information with the MongoDB JIRA instance.

Users can create accounts using the account registration page of your MMS installation.

See *Assigning Roles to Users* for details about roles and privileges, as well as adding users and assigning roles with LDAP integration.

Step 1: Click the *Users* tab.

Step 2: Click the *Add/Invite User* button.

Step 3: Enter the new user's email address and select their role.

For more information on roles, see *Assigning Roles to Users* and *User Roles*. When you have entered all information, click *Add/Invite*.

Step 4: If prompted, enter the two-factor verification code.

There might be a delay of a few seconds before you receive the prompt. MMS will prompt you for a two-factor verification code if you have not verified recently.

Step 5: Send the invitation.

Click the *Send Email Invitation* button.

View Requests

To view requests, click the *Users* tab and then select the *Requests* page. The *Requests* page lists pending requests to join your group. Users can request access when they create their MMS account, as on the registration page.

View Invitations

To view invitations, click the *Users* tab and then select the *Invitations* page. The *Invitations* page lists pending invitations to your group. When you invite a user, MMS then sends an email to the prospective new user and lists the invitation until the user accepts.

Remove Users

Step 1: Click the guilabel:*Users* tab and then select the *Users* page.

Step 2: Remove the user.

Locate the user and click the garbage can on the user's line.

Working with Multiple Environments

If you have multiple MongoDB systems in distinct environments and cannot monitor all systems with a single agent, you will need to add a new group. Having a second group makes it possible to run two agents.

You may also use a second group and agent to monitor a different set of MongoDB instances in the same environment if you want to segregate the hosts within the MMS console. A user can only view data from the hosts monitored in a single group at once.

After adding a second group, the MMS interface will have a drop down list that will allow you to change groups. Selecting a new group will refresh the current page with the data available from the servers in this group.

Create Group

Step 1: In MMS, select the *Users* tab.

Step 2: Click the *Add New Group* button.

Step 3: Add the group.

In the *Group Name* box, type a name for the new group and then click *Add New Group*. For security and auditing reasons, you cannot use a name used earlier. Once you name a group, the group's name cannot be changed.

Step 4: Open the group.

To access the new group, select the *Group* box at the top of the MMS interface, type the group's name, and select the group. You are the first user added to the new group.

Step 5: Assign hosts.

In the *Monitoring* section, click *Get Started*. Follow the prompts to download the agent, if you have not already, and to assign hosts to the group.

Remove Group

Please contact your MMS administrator to remove a company or group from your MMS account.

Assigning Roles to Users

MMS or an LDAP server can assign roles to individual users to limit actions users can perform, as well as data users see in the application. With LDAP integration, follow the steps to setup *Optional LDAP Authentication* then create LDAP groups for each available MMS role.

Users must have User Admin or Global User Admin roles assigned to them to assign roles to users. A person with the User Admin role can assign roles to users in their group. A person with the Global User Admin role can assign roles to any user in any group. You cannot assign roles for yourself.

Upon successful login, the first user completes a welcome form to create the initial MMS group. This form includes assigning roles. For LDAP authentication, the welcome form includes the ability to assign LDAP groups to the MMS group-level and global roles.

See *User Roles* for roles available for a group.

Assign a Role with MMS

To assignroles with On-Prem MongoDB Management Service, go to the *Users* page, click the *Users* tab, and click the pencil icon to the far right of the name of the user record to edit. Click the checkboxes to assign roles.

Assign Roles with LDAP

First, create groups on your LDAP server for each of the available MMS group-level and global roles.

To assign LDAP groups to MMS roles, click the Admin link at the top right of any MMS page, then click Monitoring, which displays the Groups page. Click the pencil icon at the far right of a group name. Edit the Roles interface by adding the appropriate LDAP group name to its corresponding MMS group name.

Because MMS does not update role assignments stored in your LDAP server, assign roles by assigning users to groups in your LDAP server.

Configure global roles in `conf-mms.properties` file.

See *Optional LDAP Authentication* for more details about LDAP integration with MMS.

3.5 Manage Events

Overview

MMS provides event tracking. The *Activity* tab displays a feed of all events tracked by MMS, including alerts.

View All Events

To view all events, click the *Activity* tab and then select *All Activity*. The *All Activity* page displays an activity feed of all events tracked by MMS. If you have open alerts, the page displays them above the feed.

The events tracked by MMS include:

- Events generated by users through interaction with the user interface, such as adding a host, changing a user's role, or starting a backup.
- Events generated in the monitored environment, such as restarting a host or electing a new primary in a replica set.
- Events generated internally by the system, such as opening an alert or deactivating a host.

Filter the Event Feed

You can filter the event feed by date.

Step 1: Select the *Activity* tab and then select *All Activity*.

Step 2: Click the gear icon and specify a date range.

Download the Event Feed

You can download the event feed as a CSV file (comma-separated values).

Step 1: Select the *Activity* tab and then select *All Activity*.

Step 2: Click the gear icon and select *Download as CSV File*.

You can download all events or choose to filter the feed before downloading. MMS limits the number of events returned to 10,000.

Alerts and Alert Configurations

To manage alerts and alert configurations, see *Manage Alerts*.

3.6 Create an Alert Configuration

Overview

You can create an alert configuration from scratch or clone it from an existing alert. This section describes both.

To implement alert escalation, you can create multiple alert configurations with different minimum frequencies. MMS processes alerts on a 5-minute interval. Therefore, the minimum frequency for an alert is 5 minutes. The time between re-notifications increases by the frequency amount every alert cycle (e.g. 5 minutes, 10 minutes, 15 minutes, 20 minutes, etc.) up to a maximum of 24 hours. The default frequency for a new alert configuration is 60 minutes.

When an alert state triggers, you can set a time to elapse before MMS will send alert messages at the specified interval. This helps eliminate false positives. Type in the *after waiting* field the number of minutes to wait before sending the alert at the specified interval for each recipient.

Costs to send alerts depend on your telephone service contract. Many factors may affect alert delivery, including do not call lists, caps for messages sent or delivered, delivery time of day, and message caching.

Procedures

You can create a new alert configuration or clone an existing one. This section provides both procedures.

Create an Alert

Step 1: Select the *Activity* tab and then select *Alert Settings*.

Step 2: Click the *Add Alert* button.

Step 3: Select the component to monitor and the condition that triggers the alert. In *Alert if*, select the target component. If you select `Host`, you must also select the type of host.

Next, select the condition and, if applicable, specify the threshold for the metric. For explanations of alert conditions and metrics, see *Alert Conditions*.

In *For*, you can optionally filter the alert to apply only to a subset of the monitored targets. This option is available only if the targets are hosts or replica sets.

Step 4: Select the alert recipients and choose how they receive the alerts. In *Send to*, specify the alert interval and distribution method for each alert recipient. Click *Add* to add more recipients.

For a user to receive an `SMS` alert, the user must have correctly entered their telephone number in their *Alerts* window. MMS removes all punctuation and letters and only uses the digits for the telephone number.

If you are outside of the United States or Canada, you will need to include '011' and your country code. For instance, for New Zealand (country code 64), you would need to enter '01164', followed by your phone number. Alternately, you can sign up for a Google Voice number, and use that number for your authentication.

Step 5: Select the alert recipients and choose how they receive the alerts. In *Send to*, specify the alert interval and distribution method for each alert recipient. Click *Add* to add more recipients.

For a user to receive an `SMS` alert, the user must have correctly entered their telephone number in their *Alerts* window. MMS removes all punctuation and letters and only uses the digits for the telephone number.

If you are outside of the United States or Canada, you will need to include '011' and your country code. For instance, for New Zealand (country code 64), you would need to enter '01164', followed by your phone number. Alternately, you can sign up for a Google Voice number, and use that number for your authentication.

For HipChat alerts, enter the HipChat room name and API token. Alerts will appear in the HipChat room message stream. See the *Settings page* to define default group alerts settings for HipChat.

For PagerDuty alerts, enter only the service key. Define escalation rules and alert assignments in PagerDuty. See the *Settings page* to define default group alerts settings for PagerDuty.

For SNMP alerts, specify the hostname that will receive the v2c trap on standard port `162`.

The MIB file for SNMP is available for download here.

Users must ensure that they have entered the correct number into the *Alerts* window. MMS removes all punctuation and letters and only uses the digits for the telephone number.

If you are outside of the United States or Canada, you will need to include '011' and your country code. For instance, for New Zealand (country code 64), you would need to enter '01164', followed by your phone number. Alternately, you can sign up for a Google Voice number, and use that number for your authentication.

Step 5: Click *Save*.

Clone an Alert

You can create new alerts by cloning an existing alert then editing it.

Step 1: Select the *Activity* tab and then select *Alert Settings*.

Step 2: Click the *gear icon* to the right of an alert and then select *Clone*.

Step 3: Select the component to monitor and the condition that triggers the alert. In *Alert if*, select the target component. If you select `Host`, you must also select the type of host.

Next, select the condition and, if applicable, specify the threshold for the metric. For explanations of alert conditions and metrics, see *Alert Conditions*.

In *For*, you can optionally filter the alert to apply only to a subset of the monitored targets. This option is available only if the targets are hosts or replica sets.

Step 4: Select the alert recipients and choose how they receive the alerts. In *Send to*, specify the alert interval and distribution method for each alert recipient. Click *Add* to add more recipients.

For a user to receive an `SMS` alert, the user must have correctly entered their telephone number in their *Alerts* window. MMS removes all punctuation and letters and only uses the digits for the telephone number.

If you are outside of the United States or Canada, you will need to include '011' and your country code. For instance, for New Zealand (country code 64), you would need to enter '01164', followed by your phone number. Alternately, you can sign up for a Google Voice number, and use that number for your authentication.

For HipChat alerts, enter the HipChat room name and API token. Alerts will appear in the HipChat room message stream. See the *Settings page* to define default group alerts settings for HipChat.

For PagerDuty alerts, enter only the service key. Define escalation rules and alert assignments in PagerDuty. See the *Settings page* to define default group alerts settings for PagerDuty.

For SNMP alerts, specify the hostname that will receive the v2c trap on standard port `162`.

The MIB file for SNMP is available for download here.

Users must ensure that they have entered the correct number into the *Alerts* window. MMS removes all punctuation and letters and only uses the digits for the telephone number.

If you are outside of the United States or Canada, you will need to include '011' and your country code. For instance, for New Zealand (country code 64), you would need to enter '01164', followed by your phone number. Alternately, you can sign up for a Google Voice number, and use that number for your authentication.

Step 5: Click *Save*.

3.7 Manage Alerts

Overview

You can manage alerts and alert configurations from the *Activity* tab. An alert configuration defines the conditions that trigger an alert and defines the notifications to be sent.

When a condition triggers an alert, users receive the alert at regular intervals until the alert is resolved or canceled. Users can mark the alert as acknowledged for a period of time but will again receive notifications when the acknowledgment period ends if the alert condition still exists.

Alerts end when the alert is resolved or canceled. An alert is resolved, also called "closed," when the condition that triggered the alert has been corrected. MMS sends users a notification at the time the alert is resolved.

An alert is canceled if the alert configuration that triggered the alert is deleted or disabled, or if the target of the alert is removed from the system. For example, if you have an open alert for "Host Down" and you delete that host from MMS, then the alert is canceled. When an alert is canceled, MMS does not send a notification and does not record an entry in the *activity feed*.

Manage Alert Configurations

View Alert Configurations

Alert configurations define the conditions that trigger alerts and the notifications sent when alerts are triggered. MMS creates certain alert configurations automatically when a new group is created.

To view alert configuratons, click the *Activity* tab and then select the *Alert Settings* page.

Create or Clone an Alert Configuration

To create or clone an alert configuration, see *Create an Alert Configuration*.

Modify an Alert Configuration

Each alert configuration has a distribution list, a frequency for sending the alert, and a waiting period after an alert state triggers before sending the first alert.

By default, an alert configuration sends alerts at 60-minute intervals. You can modify the interval. The minimum interval is 5 minutes.

Step 1: Select the *Activity* tab and then select *Alert Settings*.

Step 2: Click the *gear icon* to the right of an alert and then select *Edit*.

Step 3: Select the component to monitor and the condition that triggers the alert.

In *Alert if*, select the target component. If you select `Host`, you must also select the type of host.

Next, select the condition and, if applicable, specify the threshold for the metric. For explanations of alert conditions and metrics, see *Alert Conditions*.

In *For*, you can optionally filter the alert to apply only to a subset of the monitored targets. This option is available only if the targets are hosts or replica sets.

Step 4: Select the alert recipients and choose how they receive the alerts.

In *Send to*, specify the alert interval and distribution method for each alert recipient. Click *Add* to add more recipients.

For a user to receive an `SMS` alert, the user must have correctly entered their telephone number in their *Alerts* window. MMS removes all punctuation and letters and only uses the digits for the telephone number.

If you are outside of the United States or Canada, you will need to include '011' and your country code. For instance, for New Zealand (country code 64), you would need to enter '01164', followed by your phone number. Alternately, you can sign up for a Google Voice number, and use that number for your authentication.

For HipChat alerts, enter the HipChat room name and API token. Alerts will appear in the HipChat room message stream. See the *Settings page* to define default group alerts settings for HipChat.

For PagerDuty alerts, enter only the service key. Define escalation rules and alert assignments in PagerDuty. See the *Settings page* to define default group alerts settings for PagerDuty.

For SNMP alerts, specify the hostname that will receive the v2c trap on standard port `162`.

The MIB file for SNMP is available for download here.

Users must ensure that they have entered the correct number into the *Alerts* window. MMS removes all punctuation and letters and only uses the digits for the telephone number.

If you are outside of the United States or Canada, you will need to include '011' and your country code. For instance, for New Zealand (country code 64), you would need to enter '01164', followed by your phone number. Alternately, you can sign up for a Google Voice number, and use that number for your authentication.

Step 5: Click *Save*.

Delete an Alert Configuration

When you delete an alert configuration that has open alerts associated to it, MMS cancels the open alerts and sends no further notifications. This is true whether users have acknowledged the alerts or not.

Step 1: Select the *Activity* tab and then select *Alert Settings*.

Step 2: Click the *gear icon* to the right of an alert and then select *Delete*.

Step 3: Click *Confirm*.

Disable or Enable an Alert Configuration

When you disable an alert configuration it remains visible in a *grayed out* state. MMS automatically cancels active alerts related to a disabled alert configuration. You can reactivate disabled alerts.

For example, if you have an alert configured for *Host Down* and you currently have an active alert telling you a host is down, MMS automatically cancels active *Host Down* alerts if you disable the default *Host Down* configuration. MMS will send no further alerts of this type unless the disabled alert is re-enabled.

Step 1: Select the *Activity* tab and then select *Alert Settings*.

Step 2: Click the *gear icon* to the right of an alert and then select either *Disable* or *Enable*.

Manage Alerts

View Open Alerts

To view open alerts, click the *Activity* tab and then select *All Activity*. The *All Activity* page displays a feed of all events tracked by MMS. If you have open alerts, the page displays them above the feed.

Acknowledge an Open Alert

After you acknowledge the alert, MMS sends no further notifications to the alert's distribution list until the acknowledgement period has passed or until the alert is resolved. The distribution list receives *no* notification of the acknowledgment.

If the alert condition ends during the acknowledgment period, MMS sends a notification of the resolution. For example, if you acknowledge a host-down alert and the host comes back up during the acknowledgement period, MMS sends you a notification that the host is up.

If you configure an alert with PagerDuty, a third-party incident management service, you can only acknowledge the alert on your PagerDuty dashboard.

Step 1: Select the *Activity* tab.

The *All Activity* page appears.

Step 2: On the line item for the alert, click *Acknowledge*.

Step 3: Select the time period for which to acknowledge the alert.

MMS will send no further alert messages for the period of time you select.

Step 4: Click *Acknowledge*.

Unacknowledge an Acknowledged Alert

Step 1: Select the *Activity* tab.

The *All Activity* page appears.

Step 2: On the line item for the alert, click *Unacknowledge*.

Step 3: Click *Confirm*.

If the alert condition continues to exist, MMS will resend alerts.

View Closed Alerts

To view closed alerts, click the *Activity* tab and then select *Closed Alerts*. The *Closed Alerts* page displays alerts that users have closed explicitly or where the metric has dropped below the threshold of the alert.

3.8 Backup Alerts

If a problem with the MMS Backup system occurs, MMS sends an alert to system administrators. This page describes possible alerts and provides steps to resolve them.

Backup Agent Down

This alert is triggered if a Backup Agent for a group with at least one active replica set or cluster is down for more than 1 hour.

To resolve this alert:

1. Open the group in MMS by typing the group's name in the *GROUP* box.
2. Select the *Backup* tab and the *Backup Agents* page to see what server the Backup Agent is hosted on.
3. Check the Backup Agent log file on that server.

Backups Broken

If MMS On Prem Backup detects an inconsistency, the Backup state for the replica set is marked as "broken."

To debug the inconsistency:

1. Check the corresponding Backup Agent log. If you see a "Failed Common Points" test, one of the following may have happened.

- A significant rollback event occurred on the backed-up replica set.
- The oplog for the backed-up replica set was resized or deleted.

If either is the case, you must resync the replica set. See Resync a Member of a Replica Set.

2. Check the corresponding job log for an error message explaining the problem. In MMS, click *Admin*, then *Backup*, then *Jobs*, then the name of the job, then *Logs*. Contact MongoDB Support if you need help interpreting the error message.

Clustershot Failed

This alert is generated if MMS Backup cannot successfully take a snapshot for a sharded cluster backup. The alert text should contain the reason for the problem. Common problems include the following:

- There was no reachable mongos. To resolve this issue, ensure that there is at least one mongos showing in the MMS *Monitoring* tab's *Hosts* page.
- The balancer could not be stopped. To resolve this issue, check the log files for the first config server to determine why the balancer will not stop.
- Could not insert a token in one or more shards. To resolve this issue, ensure connectivity between the Backup Agent and all shards.

Bind Failure

This alert is generated if a new replica set cannot be bound to a Backup Daemon. The alert test should contain a reason for the problem. Common problems include:

- No primary is found. At the time the binding occurred, no primary could be detected by the Monitoring Agent. Ensure that the replica set is healthy.
- Not enough space is available on any Backup Daemon.

In both cases, resolve the issue and then re-initiate the initial sync. Alternatively, the job can be manually bound through the MMS Admin interface. In MMS, click *Admin*, then *Backup*, and then *Job Timeline*.

For information on initial sync, see Replica Set Data Synchronization.

Snapshot Behind Snitch

This alert is triggered if the latest snapshot for a replica set is significantly behind schedule. Check the job log in the MMS Admin interface for any obvious errors. In MMS, click *Admin*, then *Backup*, then *Jobs*, then the name of the job, and then *Logs*.

3.9 Connect to Hosts with Kerberos Authentication

Enterprise Feature

Only MongoDB Enterprise provides support for Kerberos.

Kerberos is a generic authentication protocol available in MongoDB Enterprise after version 2.4. The Monitoring and Backup Agents can authenticate to hosts using Kerberos in addition to the default MongoDB authentication protocol.

Considerations

Install the *Monitoring Agent* and/or *Backup Agent* and all requirements before beginning to configure Kerberos.

You must configure the Kerberos Key Distribution Center (KDC) to grant tickets that are valid for at least four hours. The Monitoring Agent takes care of periodically renewing the ticket. The KDC service provides session tickets and temporary session keys to users and computers.

Install Required Operating System Packages

Debian and Ubuntu Linux

Install the following required packages:

```
sudo apt-get install krb5-user libsasl2-modules-gssapi-mit
```

Red Hat Enterprise, CentOS and Fedora Linux

Install the following required packages:

```
sudo yum install krb5-workstation.x86_64 cyrus-sasl-gssapi.x86_64
```

Configure Kerberos Environment

Step 1: Create or configure the `/etc/kerb5.conf` file on the system to integrate this host into your Kerberos environment.

Step 2: Ensure the `kinit` binary is available at the `/user/bin/kinit` path.

Create Kerberos Principal and MongoDB User

Tip

If you are running both the Monitoring Agent and the Backup Agent on the same server, then both agents must connect as the same Kerberos Principal.

Step 1: Create or choose a Kerberos principal for the On Prem MMS Monitoring and/or On Prem MMS Backup agent.

Step 2: Generate a keytab for the Kerberos principal and copy it to the system where the agent runs.

Ensure the user that will run the agent is the same user that owns the keytab file.

Step 3: Create a MongoDB user for the new Kerberos principal.

See *MMS Agent Authentication Requirements* for more information about required user roles.

Step 4: Edit the agent's `/etc/mongodb-mms/monitoring-agent.config` file to inform the agent about the keytab and principal identifier.

Set the `krb5Principal` to the name of the Kerberos principal:

```
krb5Principal=<id>
```

For example:

```
krb5Principal=mmsagent/someserver.example.com@EXAMPLE.COM
```

Set the `kerb5Keytab` value to the complete absolute path of the keytab file:

```
krb5Keytab=None
```

For example:

```
krb5Keytab=/etc/mongodb-mms/mmsagent.keytab
```

Specify Kerberos for Hosts

For each host that connects with Kerberos, go to the *Hosts* page, and the *Hosts*, *Mongos*, or *Configs* tab to display the lists of available hosts. Click the *gear icon* on the right of any host and select *Kerberos (GSSAPI)* as the Auth Mechanism on the *Credentials* tab of the Edit Host interface.

3.10 MMS Public API Principles

The MMS Public API follows the principles of the REST architectural style to expose a number of internal resources which enable programmatic access to MMS's features.

Concepts

HTTP Methods

All resources support a subset of these common HTTP Methods:

- `GET` - Retrieve the JSON representation of a resource.
- `POST` - Create a new resource using the provided JSON representation.
- `PUT` - Replace a resource with the provided JSON representation.
- `PATCH` - Update the specified fields in a resource using the provided JSON representation.
- `DELETE` - Remove a resource.

JSON

All entities are represented in JSON. The following rules and conventions apply:

- When sending JSON to the server via `POST` or `PUT`, make sure to specify the correct content type request header: `Content-Type: application/json`
- Invalid fields will be *rejected* rather than *ignored*. If, for example, you attempt to create a new entity and misspell one of the fields, or if you attempt to update an existing entity and include a field that cannot be modified, the server will respond with a 400 status code and an error message stating which field was invalid.

- All dates are returned as ISO-8601 formatted strings designated in UTC. When sending dates to the server (ie, as query parameters or fields in `POST` or `PATCH` request entities), use ISO-8601 formatted dates. If you do not specify a time zone, UTC is assumed. However, it is highly recommended that you include a time zone designator to avoid any ambiguity.
- In some cases, a timestamp will be returned as a BSON timestamp, most notably in the backup resources. These are represented in JSON documents as an object with two fields: `date`, an ISO-8601 formatted date string in UTC with granularity to the second, and `increment` a 32-bit integer.
- Fields that contain numeric values in a particular unit will be named so as to disambiguate the unit being used. For example, a host's uptime is returned in millseconds, so the name of the host entity field is `uptimeMsec`.
- Fields that do not have a current value will be returned with an appropriate default value. For example, MMS will not have any statistics for a newly discovered host, so any statistics-related fields will have a value of zero. Fields that do not have a sensible default value will be omitted from the entity. For example, a host that is not using authentication will omit the `username` field from the returned entity.
- The fields in the JSON documents returned by the server are in no particular order, and it may change. Do not depend on the order of the fields.

Linking

Each resource includes one or more links to sub-resources and/or related resources. For example, a host has a link to the group it belongs to, the replica set it belongs to, and so on. Links are placed in the `links` field of an entity, which is an array of link relation objects. Each link relation has two fields:

- `rel` - Name (or type) of the relation. Many of these are considered Extension Relation Types and will be prefixed by *http://mms.mongodb.com*.
- `href` - The target URL.

All entities include at least one link relation called `self`, which is simply its own URL. When an entity is part of a list (ie, when requesting all hosts in a group), then it only includes the `self` link relation. Here's an example of a portion of a **host** resource with a few links:

```
{
  "id": "xxx",
  "groupId": "yyy",
  "hostname": "mongodb.foo.com",
  "port": 27017,
  // additional host properties...
  "links": [
    {
      "rel": "self",
      "href": "https://mms.mongodb.com/api/public/v1.0/groups/xxx/hosts/yyy"
    },
    {
      "rel": "http://mms.mongodb.com/group",
      "href": "https://mms.mongodb.com/api/public/v1.0/groups/xxx"
    }
  ]
}
```

For more information, refer to the **'Web Linking Specification'_**. Note that although the specification describes a format for including links in the HTTP response headers, doing so is not a requirement. To make the API easily browsable, it includes the links in the response body rather than in the response headers.

Lists

Some resources return a list of entities. For example, you can request a list of all **hosts** in a **group**. When a list of entities is expected in a response, the results will be returned in batches bounded by two query parameters:

- `pageNum` - Page number (1-based). Defaults to 1 if not specified.
- `itemsPerPage` - Maximum number of items to return, up to a maximum of 100. Defaults to 100 if not specified.

The response entity contains three fields:

- `totalCount` - The total number of items in the entire result set. For example, if a group has a total of 57 hosts, and you make a request with `pageNum=6` and `itemsPerPage=10`, then `totalCount` will be 57.
- `results` - The result set, which is an array of entity documents.
- `links` - Contains one to three link relations: `previous` for the previous page of results (omitted for the first page); `next` for the next page of results (omitted for the last page); `self` for the current page (always present).

If you make a request for a list of entities and there are no results, then the API will respond with a 200 status code and the `results` array will be empty. It does *not* respond with a 404 in this case, since the list of entities may not be empty at some point in the future. However, had you requested a list of entities in a context that does not exist (ie, the list of hosts for a non-existent group), then this *will* result in a 404 response status.

Here's an example respoonse for the second page of 10 hosts in a group with a total of 57 hosts:

```
{

  "totalCount": 57,
  "results": [
    {
      "id": "yyy",
      "groupId": "xxx",
      // additional host properties...
    },
    // additional host documents...
  ],
  "links": [
    {
      "rel": "previous",
      "href": "https://www.mongodb.com/api/public/v1.0/groups/xxx/hosts?itemsPerPage=10&pageNum=1"
    },
    {
      "rel": "next",
      "href": "https://www.mongodb.com/api/public/v1.0/groups/xxx/hosts?itemsPerPage=10&pageNum=3"
    }
  ]
}
```

Envelopes

Some clients may not be able to access the HTTP response headers and/or status code. In that case, you can request that the response include an "envelope," which is simply an extra layer of information in the JSON document that contains any relevant details that would normally be in the response headers. By default, the API will *not* include the response in an envelope. To request one, simply add the query parameter `envelope=true`.

For responses that contain a single entity, the envelope will contain two fields:

- `status` - The HTTP status code.

- `content` - The requested entity.

For responses that contain a list of entities, there is already an envelope that wraps the results, so specifying `envelope=true` in this case will only add the `status` field to the existing envelope.

Pretty Printing

By default, extraneous whitespace is stripped from the JSON returned by the server. To ask for pretty-printed JSON, simply append the `pretty=true` query parameter to any request. Note that all the examples in this document show pretty-printed JavaScript for clarity, although the example URLs do not contain this additional query parameter.

Response Codes

Responses utilize the standard HTTP response codes, including:

Code	Meaning	Notes
200	OK	The request was successful. This is typically the response to a successful `GET` request.
201	Created	A new resource was created. This is typically the response to a succcessful `POST` request.
202	Accepted	A request for an asynchronous operation was accepted.
400	Bad Request	Something was wrong with the client request.
401	Unautho-rized	Authentication is required but was not present in the request. Typically this means that the digest authentication information was omitted from the request.
403	Forbidden	Access to the specified resource is not permitted. Usually means that the user associated with the given API Key is not allowed to access the requested resource.
404	Not Found	The requested resource does not exist.
405	Method Not Allowed	The HTTP method is not supported for the specified resource. Keep in mind that each resource may only support a subset of HTTP methods. For example, you are not allowed to `DELETE` the root resource.
409	Conflict	This is typically the response to a request to create or modify a property of an entity that is unique when an existing entity already exists with the same value for that property. For example, attempting to create a group with the same name as an existing group is not allowed.
5xx	Various server errors	Something unexpected went wrong. Try again later and consider notifying MMS Support.

Errors

When a request results in an error, the response body will contain a document with additional details about what went wrong. The document contains three fields:

- `error` - The error code, which is simply the HTTP status code.
- `reason` - A short description of the error, which is simply the HTTP status phrase.
- `detail` - A more detailed description of the error.

For example, here is the response body for a request for a host that does not exist:

```
{
  "error": 404,
  "reason": "Not Found",
  "detail": "No host exists with ID yyy in group xxx."
}
```

Authentication

As previously mentioned, the MMS API uses HTTP Digest Authentication. The details of digest authentication are beyond the scope of this document, but it essentially requires a username and a password which are hashed using a unique server-generated value called a **nonce**. The username is the username of a registered MMS account, and the password is an API Key associated to that username.

Keep the following points in mind:

- The server-generated nonce is used by the client to hash the username and password before sending them back to the server to authenticate a request. The nonce is only valid for a short amount of time as per the digest authentication specification. This is to prevent replay attacks, so you can't cache a nonce and use it forever.
- Some resource methods require even more security and are additionally protected by a whitelist, which is a list of client IP addresses associated to a user account that are permitted to access these protected resources.
- The MMS UI has a concept of **roles**, which allow more fine-grained control of the operations a user is allowed to perform. The API resources also enforce the same authorization rules, so the resources and methods that can be accessed by an API Key are governed by the roles granted to the associated user. For example, to `DELETE` a host, the user that owns the API key used to make the request must be a `Group Monitoring Admin` or `Group Owner` in the group that the host belongs to.
- Many resources are tied to a group, as evidenced by URLs of the form `.../api/public/v1.0/groups/<GROUP-ID>/...`. For these resources, the user tied to the API key must be a member of the group *or* must be assigned to one of the `GLOBAL` roles. Otherwise the server will respond with a 403 (Forbidden) status.

Additional Information

See the *MMS Public API Principles* for an background on the use and operation of the MMS public API, and *MMS Public API* for a complete reference of all resources available in the MMS public API.

3.11 Use MMS Public API

Overview

The MMS Public API follows the principles of the REST architectural style to expose a number of internal resources which enable programmatic access to MMS's features. This document describes the procedure for getting started with the MMS Public API.

Procedure

Before using the API, you must:

1. *Generate an API Key* - Go to the **Settings** page in the MMS UI and click on the **Public API Settings** tab. Here, you can manage the API Keys associated to your MMS user account. Note the following:
   (a) You can have up to ten keys associated to your account. Each key can be either enabled or disabled, but be aware that they both count towards the ten key limit.
   (b) When a new key is generated, it will be shown to you *one time only.* An API Key is like a password, so keep it secret. For that reason, we will not show the entire key in the MMS UI after it is initially created.
   (c) API Keys are associated to a user and therefore have the same level of access as that user.

2. *Define your whitelist* - Certain API operations require a higher level of security and are protected by a whitelist. Only client requests that originate from a whitelisted IP address will be permitted to invoke such operations. To define your whitelist, go to the **Settings** page in the MMS UI and click on the **Public API Settings** tab. Here, you can manage the IP addresses in your whitelist. Currently, you must enter each permitted IP address individually; CIDR notation is not supported.
3. *Enable the Public API for each Group* - The Public API is enabled on a per-group basis, so make sure to enable it for all the Groups that need to use it. To enable it, go to the **Settings** page in the MMS UI and click on the **API Settings** tab. You will see an ON/OFF switch for turning the API on or off. Note that this setting is only visible to users with the `Group Owner` role.

Additional Information

See the *MMS Public API Principles* for an background on the use and operation of the MMS public API, and *MMS Public API* for a complete reference of all resources available in the MMS public API.

4 On Prem MMS Monitoring

On Prem MMS Monitoring is a service for monitoring MongoDB deployments, and an integral part of of the On-Prem MongoDB Management Service. On Prem MMS Monitoring collects statistics on all key server and hardware indicators and presents this data through an intuitive web interface.

This manual describes the installation of the Monitoring Agent and operation of the On Prem MMS Monitoring web console. You can find answers to common questions in the FAQs, but for all other inquiries please feel free to open a JIRA ticket.

Getting Started with MMS Monitoring Register for backup, install the Backup Agent, and begin using MMS backup to capture real-time back ups of MongoDB deployments.

Using MMS Monitoring Use the MMS monitoring interface to configure the Monitoring Agent and the operation of your MMS Monitoring deployment..

Monitoring Operations An outline of the interfaces for Monitoring data in MMS.

4.1 Getting Started with MMS Monitoring

Install or Update Agent with RPM Packages Install or update the MMS Monitoring Agent using an `rpm` pacakge.

Install or Update Agent with Debian Packages Install or update the MMS Monitoring Agent using a `deb` pacakge.

Install or Update Agent on OS X Install or update the MMS Monitoring Agent on OS X systems.

Install or Update Agent on Other Linux Systems Outlines the process for integrating MMS Monitoring with your MongoDB deployment to collect data and provide alerts to help you maintain the health of your system.

Install or Update Agent on Windows Install or update the MMS Monitoring Agent using on windows.

Add Hosts to MMS Monitoring Outlines the process for adding hosts to MMS Monitoring.

Install or Update the Monitoring Agent with `rpm` Packages

Overview

The MMS Monitoring Agent is a lightweight component that runs within your infrastructure, connects to your MongoDB processes, collects data about the state of your deployment, and then sends the data to the On Prem MMS

Monitoring service which processes and renders this data. The agent initiates all connections to the On Prem MMS Monitoring service, and communications between the agent and the On Prem MMS Monitoring service are encrypted. A single agent can collect data from multiple MongoDB processes. Consider the following diagram of an example deployment:

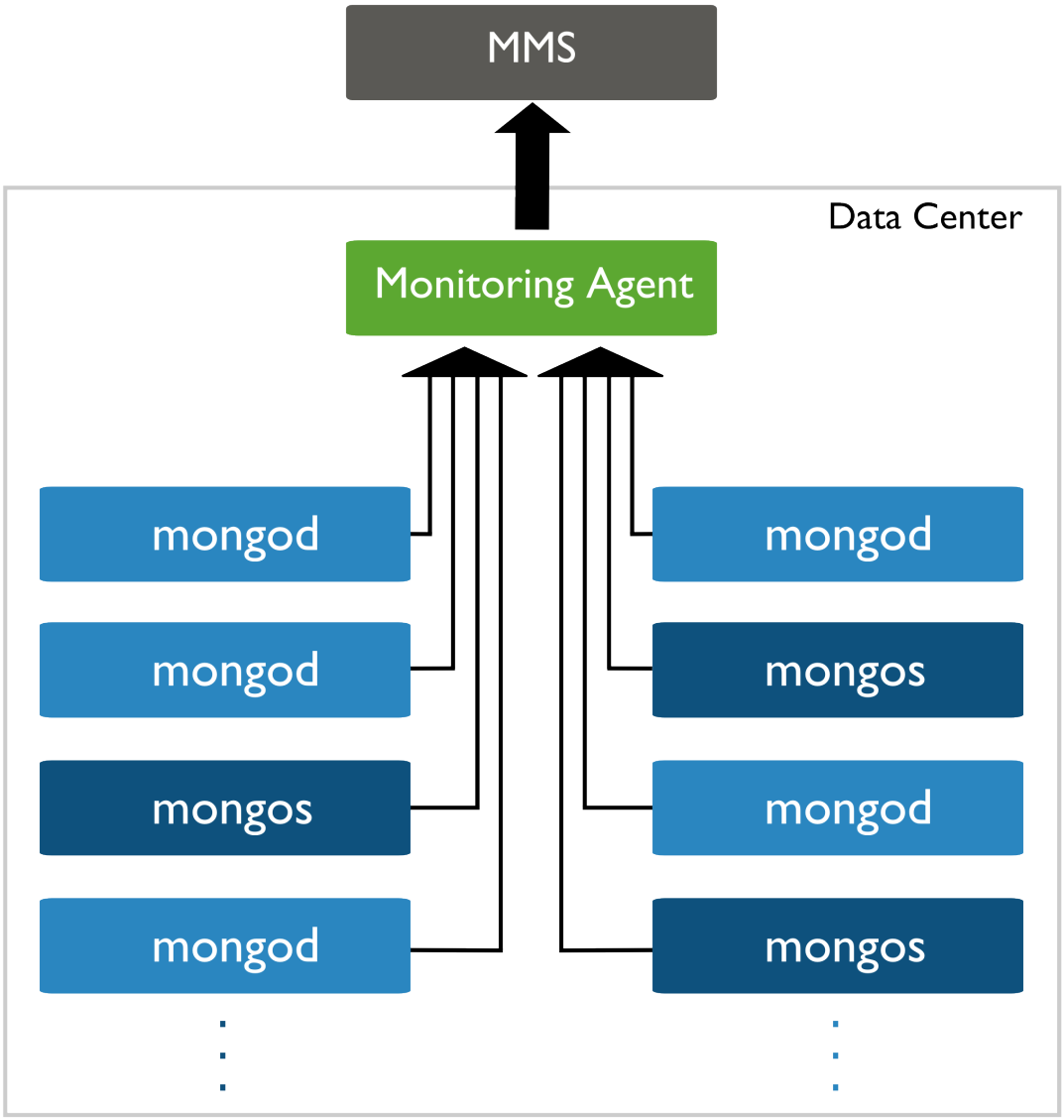


Figure 1: One Monitoring Agent can pull from multiple data centers and instances then push data to MMS.

This tutorial will guide you through the steps necessary to install or update On Prem MMS Monitoring on your system. You must install the On Prem Monitoring server itself before installing the Monitoring Agent.

See the *Frequently Asked Questions: Monitoring* page for additional information.

Considerations

Connectivity You must configure the networking rules of your deployment so that:

- the Monitoring Agent can connect to all `mongod` and `mongos` instances that you want to monitor.

- the Monitoring Agent can connect to On Prem MMS Monitoring server on port 443 (i.e. `https`.)

The On Prem MMS Monitoring server does not make *any* outbound connections to the agents or to MongoDB instances. If *Exposed DB Host Check is enabled,* the On Prem MMS Monitoring server will attempt to connect to your servers occasionally as part of a vulnerability check.

Ensure all `mongod` and `mongos` instances are not accessible to hosts outside your deployment.

Monitoring Agent Redundancy A single Monitoring Agent is sufficient and strongly recommended. However, you can run additional instances of the agent as hot standbys to provide redundancy. If the primary agent fails, a standby agent starts monitoring.

When you run multiple agents, only one Monitoring Agent per group or environment is the **primary agent**. The primary agent reports the cluster's status to MMS. The remaining agents are completely idle, except to log their status as standby agents and to periodically ask MMS whether they should become the primary.

To install additional agents, simply repeat the installation process.

Access Control If you are using MongoDB authentication, see the *Authentication Requirements documentation*.

Collection Interval If you are updating the agent, keep in mind that when the Monitoring Agent restarts, there is a five-minute delay before that agent begins collecting data and sending pings to On Prem MMS Monitoring. If you have multiple agents, this delay permits other agents in your infrastructure to become the primary agent and permits On Prem MMS Monitoring to determine which agent will be primary.

During this interval, the restarted Monitoring Agent will not collect data.

Procedures

This section includes procedures for both installing and updating the Monitoring Agent.

Install the Monitoring Agent with an `rpm` Package Use this procedure to install the agent on RHEL, CentOS, SUSE, Amazon Linux, and other systems that use `rpm` packages.

Step 1: Download the latest version of the Monitoring Agent package. In a system shell, issue the following command:

```
curl -OL <mmsUri>/download/agent/monitoring/mongodb-mms-monitoring-agent-latest.x86_64.rpm
```

Step 2: Install the Monitoring Agent package. Issue the following command:

```
sudo rpm -U mongodb-mms-monitoring-agent-latest.x86_64.rpm
```

Step 3: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Monitoring Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 4: Edit the `monitoring-agent.config` file to include your MMS API key. In the `<install-directory>/monitoring-agent.config` file, set the `mmsApiKey` property to your API key.

Step 5: Optional. For SUSE deployments only, configure the `sslTrustedMMSServerCertificate` property. If you're deploying on SUSE, you must configure the `sslTrustedMMSServerCertificate` setting. All other users should omit this step.

Enter the following property and value in the `/etc/mongodb-mms/monitoring-agent.config` file:

```
sslTrustedMMSServerCertificate=/etc/ssl/certs/UTN_USERFirst_Hardware_Root_CA.pem
```

Save and close the file.

Step 6: Start the Monitoring Agent. Issue the following command:

```
sudo service mongodb-mms-monitoring-agent start
```

Remember, that you only need to run **1** Monitoring Agent for each MMS group. A single Monitoring Agent can collect data from many MongoDB instances.

Update the Monitoring Agent with an `rpm` Package

Step 1: Download the latest version of the Monitoring Agent package. In a system shell, issue the following command:

```
curl -OL <mmsUri>/download/agent/monitoring/mongodb-mms-monitoring-agent-latest.x86_64.rpm
```

Step 2: Install the Monitoring Agent package. Issue the following command:

```
sudo rpm -U mongodb-mms-monitoring-agent-latest.x86_64.rpm
```

Install or Update the Monitoring Agent with `deb` Packages

Overview

The MMS Monitoring Agent is a lightweight component that runs within your infrastructure, connects to your MongoDB processes, collects data about the state of your deployment, and then sends the data to the On Prem MMS Monitoring service which processes and renders this data. The agent initiates all connections to the On Prem MMS Monitoring service, and communications between the agent and the On Prem MMS Monitoring service are encrypted. A single agent can collect data from multiple MongoDB processes. Consider the following diagram of an example deployment:

This tutorial will guide you through the steps necessary to install or update On Prem MMS Monitoring on your system. You must install the On Prem Monitoring server itself before installing the Monitoring Agent.

See the *Frequently Asked Questions: Monitoring* page for additional information.

Considerations

Connectivity You must configure the networking rules of your deployment so that:

- the Monitoring Agent can connect to all **mongod** and **mongos** instances that you want to monitor.
- the Monitoring Agent can connect to On Prem MMS Monitoring server on port 443 (i.e. `https`.)

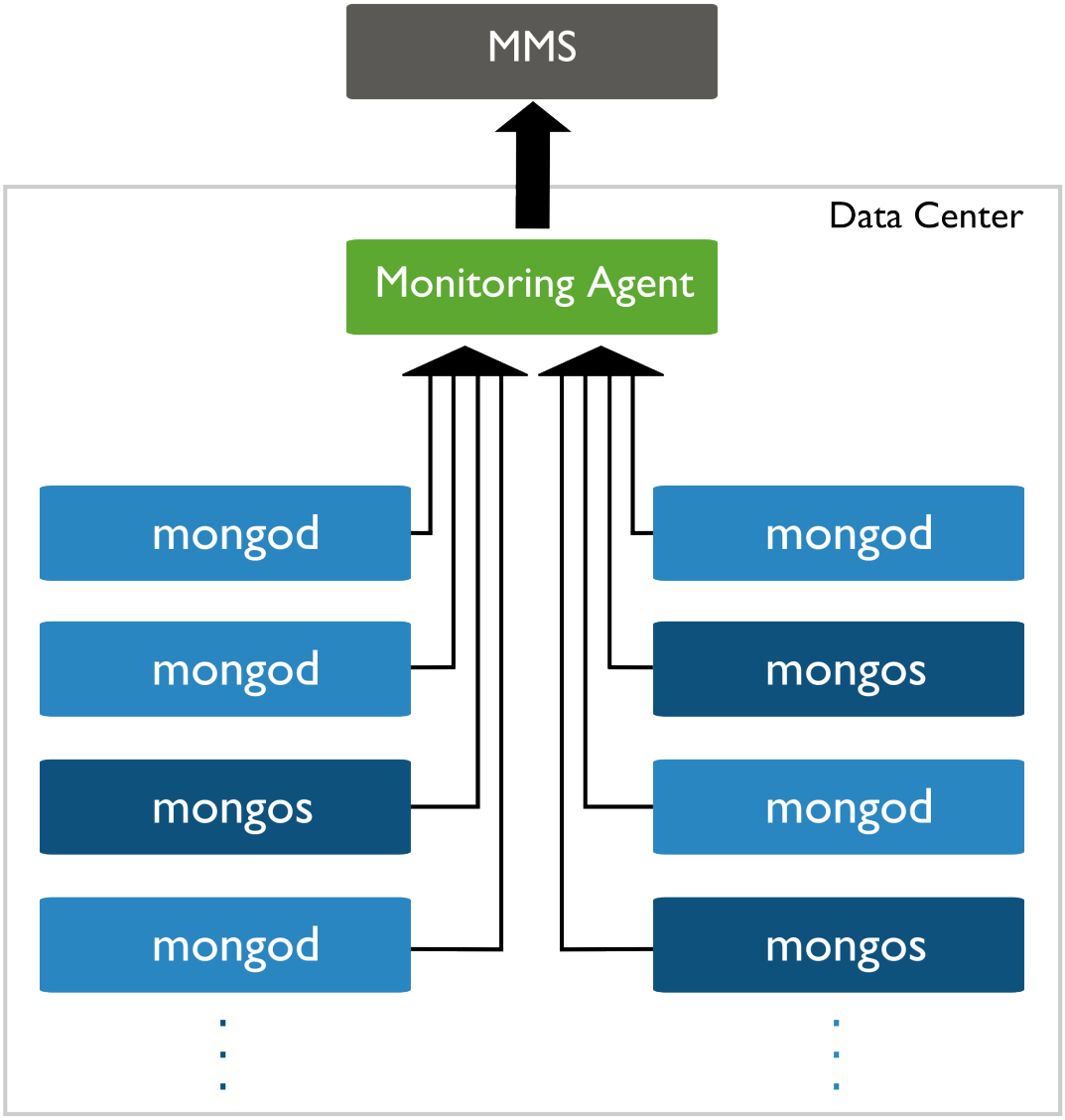


Figure 2: One Monitoring Agent can pull from multiple data centers and instances then push data to MMS.

The On Prem MMS Monitoring server does not make *any* outbound connections to the agents or to MongoDB instances. If *Exposed DB Host Check is enabled,* the On Prem MMS Monitoring server will attempt to connect to your servers occasionally as part of a vulnerability check.

Ensure all **mongod** and **mongos** instances are not accessible to hosts outside your deployment.

Monitoring Agent Redundancy A single Monitoring Agent is sufficient and strongly recommended. However, you can run additional instances of the agent as hot standbys to provide redundancy. If the primary agent fails, a standby agent starts monitoring.

When you run multiple agents, only one Monitoring Agent per group or environment is the **primary agent**. The primary agent reports the cluster's status to MMS. The remaining agents are completely idle, except to log their status as standby agents and to periodically ask MMS whether they should become the primary.

To install additional agents, simply repeat the installation process.

Access Control If you are using MongoDB authentication, see the *Authentication Requirements documentation*.

Collection Interval If you are updating the agent, keep in mind that when the Monitoring Agent restarts, there is a five-minute delay before that agent begins collecting data and sending pings to On Prem MMS Monitoring. If you have multiple agents, this delay permits other agents in your infrastructure to become the primary agent and permits On Prem MMS Monitoring to determine which agent will be primary.

During this interval, the restarted Monitoring Agent will not collect data.

Procedures

This section includes procedures for both installing and updating the Monitoring Agent.

Install the Monitoring Agent with a `deb` Package Use this procedure to install the agent on Ubuntu and other systems that use `deb` packages.

Step 1: Download the latest version of the Monitoring Agent package. Issue the following command using the system shell:

```
curl -OL <mmsUri>/download/agent/monitoring/mongodb-mms-monitoring-agent_latest_amd64.deb
```

Step 2: Install the Monitoring Agent package. Issue the following command using the system shell:

```
sudo dpkg -i mongodb-mms-monitoring-agent_latest_amd64.deb
```

Step 3: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Monitoring Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 4: Edit the `monitoring-agent.config` file to include your MMS API key. In the `<install-directory>/monitoring-agent.config` file, set the `mmsApiKey` property to your API key.

Step 5: Start the Monitoring Agent. Issue the following command:

```
sudo start mongodb-mms-monitoring-agent
```

Remember, that you only need to run **1** Monitoring Agent for each MMS group. A single Monitoring Agent can collect data from many MongoDB instances.

Update the Monitoring Agent with a `deb` Package

Step 1: Download the latest version of the Monitoring Agent package. Issue the following command using the system shell:

```
curl -OL <mmsUri>/download/agent/monitoring/mongodb-mms-monitoring-agent_latest_amd64.deb
```

Step 2: Install the Monitoring Agent package. Issue the following command using the system shell:

```
sudo dpkg -i mongodb-mms-monitoring-agent_latest_amd64.deb
```

Step 3: Start the Monitoring Agent. Issue the following command:

```
sudo start mongodb-mms-monitoring-agent
```

Remember, that you only need to run **1** Monitoring Agent for each MMS group. A single Monitoring Agent can collect data from many MongoDB instances.

Install or Update the Monitoring Agent on OS X

Overview

The MMS Monitoring Agent is a lightweight component that runs within your infrastructure, connects to your MongoDB processes, collects data about the state of your deployment, and then sends the data to the On Prem MMS Monitoring service which processes and renders this data. The agent initiates all connections to the On Prem MMS Monitoring service, and communications between the agent and the On Prem MMS Monitoring service are encrypted. A single agent can collect data from multiple MongoDB processes. Consider the following diagram of an example deployment:

This tutorial will guide you through the steps necessary to install or update On Prem MMS Monitoring on your system. You must install the On Prem Monitoring server itself before installing the Monitoring Agent.

See the *Frequently Asked Questions: Monitoring* page for additional information.

Considerations

Connectivity You must configure the networking rules of your deployment so that:

- the Monitoring Agent can connect to all **mongod** and **mongos** instances that you want to monitor.
- the Monitoring Agent can connect to On Prem MMS Monitoring server on port 443 (i.e. `https`.)

The On Prem MMS Monitoring server does not make *any* outbound connections to the agents or to MongoDB instances. If *Exposed DB Host Check is enabled,* the On Prem MMS Monitoring server will attempt to connect to your servers occasionally as part of a vulnerability check.

Ensure all **mongod** and **mongos** instances are not accessible to hosts outside your deployment.

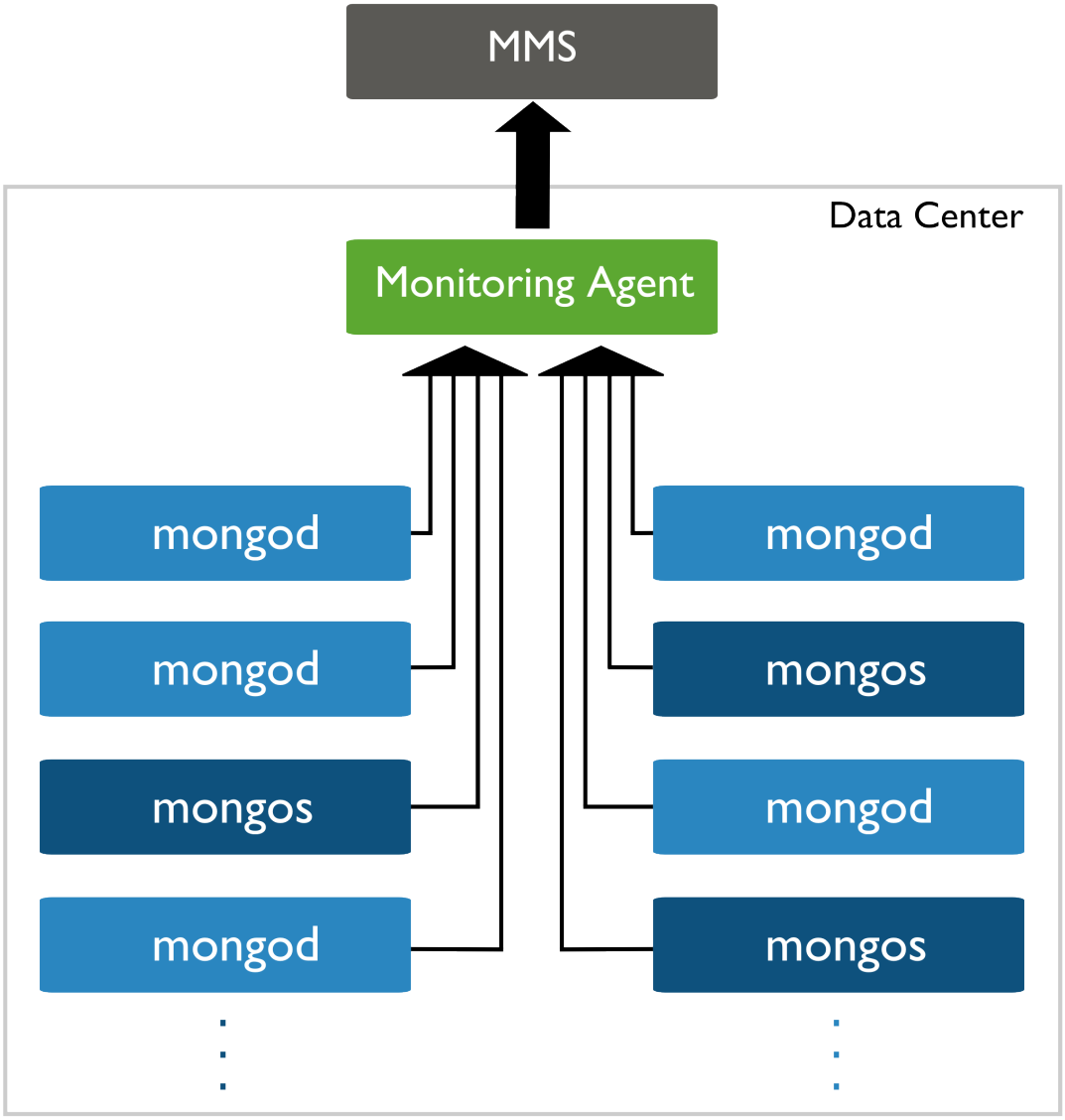


Figure 3: One Monitoring Agent can pull from multiple data centers and instances then push data to MMS.

Monitoring Agent Redundancy A single Monitoring Agent is sufficient and strongly recommended. However, you can run additional instances of the agent as hot standbys to provide redundancy. If the primary agent fails, a standby agent starts monitoring.

When you run multiple agents, only one Monitoring Agent per group or environment is the **primary agent**. The primary agent reports the cluster's status to MMS. The remaining agents are completely idle, except to log their status as standby agents and to periodically ask MMS whether they should become the primary.

To install additional agents, simply repeat the installation process.

Access Control If you are using MongoDB authentication, see the *Authentication Requirements documentation*.

Collection Interval If you are updating the agent, keep in mind that when the Monitoring Agent restarts, there is a five-minute delay before that agent begins collecting data and sending pings to On Prem MMS Monitoring. If you have multiple agents, this delay permits other agents in your infrastructure to become the primary agent and permits On Prem MMS Monitoring to determine which agent will be primary.

During this interval, the restarted Monitoring Agent will not collect data.

Procedures

This section includes procedures for both installing and updating the Monitoring Agent.

Install the Monitoring Agent on OS X Use this procedure to install the agent OS X systems.

Step 1: Download the latest version of the Monitoring Agent archive. In a system shell, issue the following command:

```
curl -OL <mmsUri>/download/agent/monitoring/mongodb-mms-monitoring-agent-latest.osx_x86_64.tar.gz
```

Step 2: Install the Monitoring Agent. To install the agent, extract the archive by issue the following command:

```
tar -xf mongodb-mms-monitoring-agent-latest.osx_x86_64.tar.gz
```

The Monitoring Agent is installed.

Step 3: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Monitoring Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 4: Edit the `monitoring-agent.config` file to include your MMS API key. In the `<install-directory>/monitoring-agent.config` file, set the `mmsApiKey` property to your API key.

Step 5: Optional. Configure the agent to use a proxy server. If the agent will connect to MMS via a proxy server, you must specify the server in the `http_proxy` environment variable. To specify the server, use the `export` command, as in the following example:

```
export http_proxy="http://proxy.example.com:9000"
```

To connect through a proxy, you must install the agent from a `.tar.gz` file, *not* from a `.deb` or `.rpm` file.

Step 6: Start the Monitoring Agent. Issue the following command:

```
nohup ./mongodb-mms-monitoring-agent >> monitoring-agent.log 2>&1 &
```

Remember, that you only need to run **1** Monitoring Agent for each MMS group. A single Monitoring Agent can collect data from many MongoDB instances.

Update the Monitoring Agent from a `tar.gz` Archive Use this procedure to update the agent on OS X systems.

Step 1: Stop any currently running Monitoring Agents. Issue the following command:

```
pkill -f mongodb-mms-monitoring-agent
```

Step 2: Download the latest version of the Monitoring Agent archive. In a system shell, issue the following command:

```
curl -OL <mmsUri>/download/agent/monitoring/mongodb-mms-monitoring-agent-latest.osx_x86_64.tar.gz
```

Step 3: Install the Monitoring Agent. To install the agent, extract the archive by issue the following command:

```
tar -xf mongodb-mms-monitoring-agent-latest.osx_x86_64.tar.gz
```

The Monitoring Agent is installed.

Step 4: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Monitoring Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 5: Edit the `monitoring-agent.config` file to include your MMS API key. In the `<install-directory>/monitoring-agent.config` file, set the `mmsApiKey` property to your API key.

Step 6: Start the Monitoring Agent. Issue the following command:

```
nohup ./mongodb-mms-monitoring-agent >> monitoring-agent.log 2>&1 &
```

Remember, that you only need to run **1** Monitoring Agent for each MMS group. A single Monitoring Agent can collect data from many MongoDB instances.

Install or Update the Monitoring Agent from Archive

Overview

The MMS Monitoring Agent is a lightweight component that runs within your infrastructure, connects to your MongoDB processes, collects data about the state of your deployment, and then sends the data to the On Prem MMS Monitoring service which processes and renders this data. The agent initiates all connections to the On Prem MMS Monitoring service, and communications between the agent and the On Prem MMS Monitoring service are encrypted. A single agent can collect data from multiple MongoDB processes. Consider the following diagram of an example deployment:

This tutorial will guide you through the steps necessary to install or update On Prem MMS Monitoring on your system. You must install the On Prem Monitoring server itself before installing the Monitoring Agent.

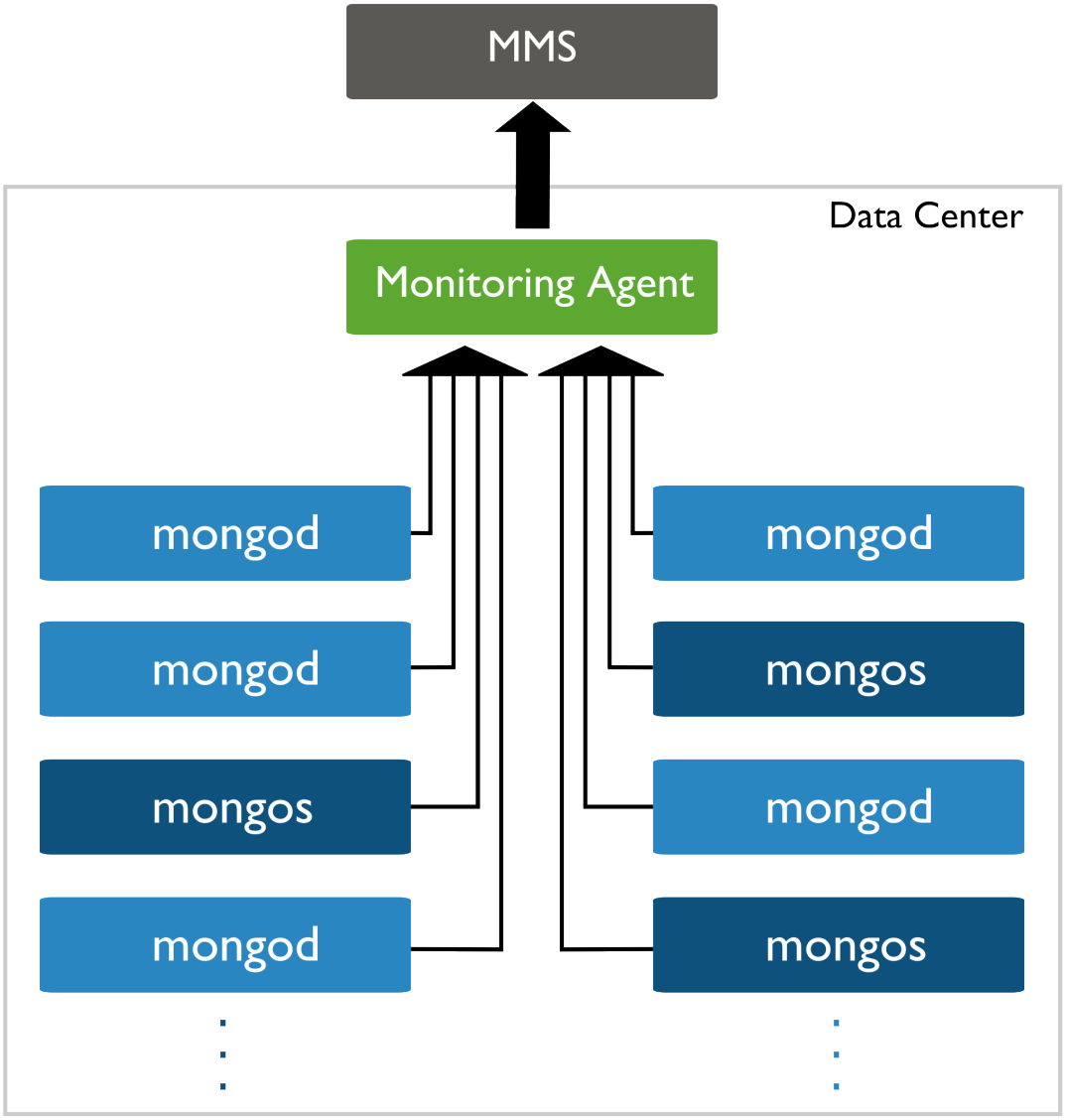


Figure 4: One Monitoring Agent can pull from multiple data centers and instances then push data to MMS.

See the *Frequently Asked Questions: Monitoring* page for additional information.

Considerations

Connectivity You must configure the networking rules of your deployment so that:

- the Monitoring Agent can connect to all **mongod** and **mongos** instances that you want to monitor.
- the Monitoring Agent can connect to On Prem MMS Monitoring server on port 443 (i.e. `https`.)

The On Prem MMS Monitoring server does not make *any* outbound connections to the agents or to MongoDB instances. If *Exposed DB Host Check is enabled,* the On Prem MMS Monitoring server will attempt to connect to your servers occasionally as part of a vulnerability check.

Ensure all **mongod** and **mongos** instances are not accessible to hosts outside your deployment.

Monitoring Agent Redundancy A single Monitoring Agent is sufficient and strongly recommended. However, you can run additional instances of the agent as hot standbys to provide redundancy. If the primary agent fails, a standby agent starts monitoring.

When you run multiple agents, only one Monitoring Agent per group or environment is the **primary agent**. The primary agent reports the cluster's status to MMS. The remaining agents are completely idle, except to log their status as standby agents and to periodically ask MMS whether they should become the primary.

To install additional agents, simply repeat the installation process.

Access Control If you are using MongoDB authentication, see the *Authentication Requirements documentation*.

Collection Interval If you are updating the agent, keep in mind that when the Monitoring Agent restarts, there is a five-minute delay before that agent begins collecting data and sending pings to On Prem MMS Monitoring. If you have multiple agents, this delay permits other agents in your infrastructure to become the primary agent and permits On Prem MMS Monitoring to determine which agent will be primary.

During this interval, the restarted Monitoring Agent will not collect data.

Procedures

This section includes procedures for both installing and updating the Monitoring Agent.

Install the Monitoring Agent from a `tar.gz` Archive Use this procedure to install the agent on Linux systems:

Step 1: Download the latest version of the Monitoring Agent archive. With a system shell, issue the following command:

```
curl -OL <mmsUri>/download/agent/monitoring/mongodb-mms-monitoring-agent-latest.linux_x86_64.tar.gz
```

Step 2: Install the Monitoring Agent. To install the agent, extract the archive by issue the following command:

```
tar -xf mongodb-mms-monitoring-agent-latest.linux_x86_64.tar.gz
```

The Monitoring Agent is installed.

Step 3: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Monitoring Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 4: Edit the `monitoring-agent.config` file to include your MMS API key. In the `<install-directory>/monitoring-agent.config` file, set the `mmsApiKey` property to your API key.

Step 5: Optional. Configure the agent to use a proxy server. If the agent will connect to MMS via a proxy server, you must specify the server in the `http_proxy` environment variable. To specify the server, use the `export` command, as in the following example:

```
export http_proxy="http://proxy.example.com:9000"
```

To connect through a proxy, you must install the agent from a `.tar.gz` file, *not* from a `.deb` or `.rpm` file.

Step 6: Start the Monitoring Agent. Issue the following command:

```
nohup ./mongodb-mms-monitoring-agent >> monitoring-agent.log 2>&1 &
```

Remember, that you only need to run **1** Monitoring Agent for each MMS group. A single Monitoring Agent can collect data from many MongoDB instances.

Update the Monitoring Agent from a `tar.gz` Archive Use this procedure to update the agent on Linux systems:

Step 1: Stop any currently running Monitoring Agents. Issue the following command:

```
pkill -f mongodb-mms-monitoring-agent
```

Step 2: Download the latest version of the Monitoring Agent archive. With a system shell, issue the following command:

```
curl -OL <mmsUri>/download/agent/monitoring/mongodb-mms-monitoring-agent-latest.linux_x86_64.tar.gz
```

Step 3: Install the Monitoring Agent. To install the agent, extract the archive by issue the following command:

```
tar -xf mongodb-mms-monitoring-agent-latest.linux_x86_64.tar.gz
```

The Monitoring Agent is installed.

Step 4: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Monitoring Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 5: Edit the `monitoring-agent.config` file to include your MMS API key. In the `<install-directory>/monitoring-agent.config` file, set the `mmsApiKey` property to your API key.

Step 6: Start the Monitoring Agent. Issue the following command:

```
nohup ./mongodb-mms-monitoring-agent >> monitoring-agent.log 2>&1 &
```

Remember, that you only need to run **1** Monitoring Agent for each MMS group. A single Monitoring Agent can collect data from many MongoDB instances.

Additional Information

If you installed the Monitoring Agent from the `tar.gz` archives, see `http://mms.mongodb.com/helptutorial/rotate-agent-log-files` to configure log rotation.

Install or Update the Monitoring Agent on Windows

Overview

The MMS Monitoring Agent is a lightweight component that runs within your infrastructure, connects to your MongoDB processes, collects data about the state of your deployment, and then sends the data to the On Prem MMS Monitoring service which processes and renders this data. The agent initiates all connections to the On Prem MMS Monitoring service, and communications between the agent and the On Prem MMS Monitoring service are encrypted. A single agent can collect data from multiple MongoDB processes. Consider the following diagram of an example deployment:

This tutorial will guide you through the steps necessary to install or update On Prem MMS Monitoring on your system. You must install the On Prem Monitoring server itself before installing the Monitoring Agent.

See the *Frequently Asked Questions: Monitoring* page for additional information.

Considerations

Connectivity You must configure the networking rules of your deployment so that:

- the Monitoring Agent can connect to all **mongod** and **mongos** instances that you want to monitor.
- the Monitoring Agent can connect to On Prem MMS Monitoring server on port 443 (i.e. `https`.)

The On Prem MMS Monitoring server does not make *any* outbound connections to the agents or to MongoDB instances. If *Exposed DB Host Check is enabled,* the On Prem MMS Monitoring server will attempt to connect to your servers occasionally as part of a vulnerability check.

Ensure all **mongod** and **mongos** instances are not accessible to hosts outside your deployment.

Monitoring Agent Redundancy A single Monitoring Agent is sufficient and strongly recommended. However, you can run additional instances of the agent as hot standbys to provide redundancy. If the primary agent fails, a standby agent starts monitoring.

When you run multiple agents, only one Monitoring Agent per group or environment is the **primary agent**. The primary agent reports the cluster's status to MMS. The remaining agents are completely idle, except to log their status as standby agents and to periodically ask MMS whether they should become the primary.

To install additional agents, simply repeat the installation process.

Access Control If you are using MongoDB authentication, see the *Authentication Requirements documentation*.

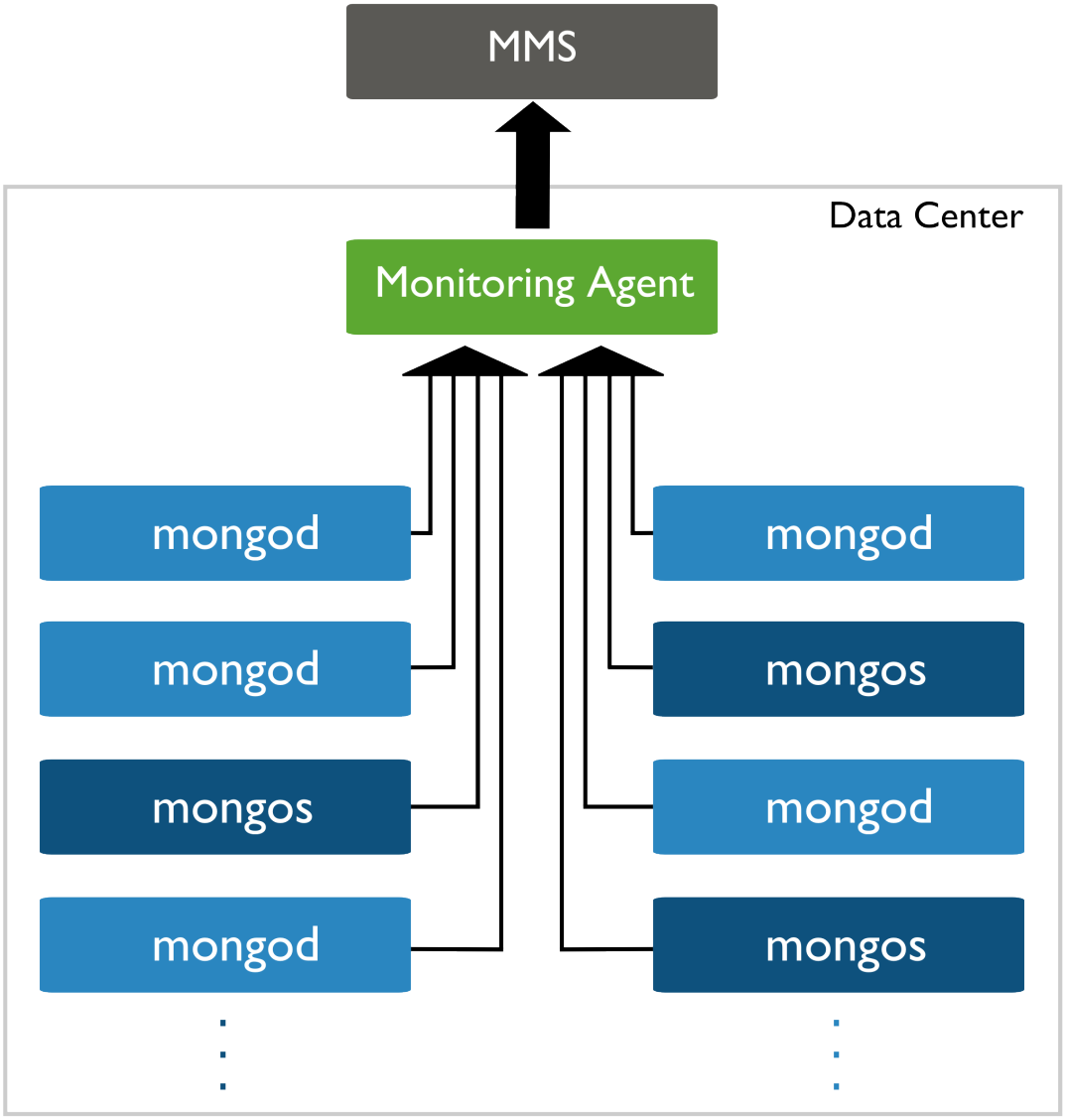


Figure 5: One Monitoring Agent can pull from multiple data centers and instances then push data to MMS.

Collection Interval If you are updating the agent, keep in mind that when the Monitoring Agent restarts, there is a five-minute delay before that agent begins collecting data and sending pings to On Prem MMS Monitoring. If you have multiple agents, this delay permits other agents in your infrastructure to become the primary agent and permits On Prem MMS Monitoring to determine which agent will be primary.

During this interval, the restarted Monitoring Agent will not collect data.

Procedures

This section includes procedures for both installing and updating the Monitoring Agent.

Install the Monitoring Agent on Windows Use this procedure to install the agent on Windows.

Step 1: Download and run the latest version of the Monitoring Agent MSI file. Download and run the 32-bit or 64-bit MSI file. During installation, the installer prompts you to specify the folder for storing configuration and log files. It is strongly advised that you encrypt or restrict access to this folder.

To download the 32-bit MSI file, use the following URL, where `<mms-server>` is the hostname of the Monitoring server:

```
<mms-server>/download/agent/monitoring/mongodb-mms-monitoring-agent-latest.windows_i386.msi
```

To download the 64-bit MSI file, use the following URL, where `<mms-server>` is the hostname of the Monitoring server:

```
<mms-server>/download/agent/monitoring/mongodb-mms-monitoring-agent-latest.windows_x86_64.msi
```

During installation, the installer prompts you to specify the folder for storing configuration and log files. It is strongly advised that you encrypt or restrict access to this folder.

Step 2: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Monitoring Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 3: Edit the `monitoring-agent.config` file to include your MMS API key. In the `<install-directory>/monitoring-agent.config` file, set the `mmsApiKey` property to your API key.

The default location for the agent configuration file is `C:\MMSData\Monitoring\monitoring-agent.config`.

Step 4: Edit the `monitoring-agent.config` file to include the hostname of the Monitoring server. Set the `mmsBaseUrl` property to the hostname of the Monitoring server.

Step 5: Start the Monitoring Agent. In Windows `Control Panel`, open `Administrative Tools`, and then open `Services`.

In the list of services, select the `MMS Monitoring Agent` service. Select the `Action` menu and select `Start`.

Remember, that you only need to run **1** Monitoring Agent for each MMS group. A single Monitoring Agent can collect data from many MongoDB instances.

Update the Monitoring Agent on Windows To update the agent on Windows systems:

Step 1: Stop any currently running Monitoring Agents. In Windows `Control Panel`, open `Administrative Tools` and then `Services`.

In the list of services, select `MMS Monitoring Agent`. Select the `Action` menu and select `Stop`.

Step 2: Download and run the latest version of the Monitoring Agent MSI file. Download and run the 32-bit or 64-bit MSI file. During installation, the installer prompts you to specify the folder for storing configuration and log files. It is strongly advised that you encrypt or restrict access to this folder.

To download the 32-bit MSI file, use the following URL, where `<mms-server>` is the hostname of the Monitoring server:

```
<mms-server>/download/agent/monitoring/mongodb-mms-monitoring-agent-latest.windows_i386.msi
```

To download the 64-bit MSI file, use the following URL, where `<mms-server>` is the hostname of the Monitoring server:

```
<mms-server>/download/agent/monitoring/mongodb-mms-monitoring-agent-latest.windows_x86_64.msi
```

During installation, the installer prompts you to specify the folder for storing configuration and log files. It is strongly advised that you encrypt or restrict access to this folder.

Step 3: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Monitoring Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 4: Edit the `monitoring-agent.config` file to include your MMS API key. In the `<install-directory>/monitoring-agent.config` file, set the `mmsApiKey` property to your API key.

Step 5: Start the Monitoring Agent. In Windows `Control Panel`, open `Administrative Tools`, and then open `Services`.

In the list of services, select the `MMS Monitoring Agent` service. Select the `Action` menu and select `Start`.

Remember, that you only need to run **1** Monitoring Agent for each MMS group. A single Monitoring Agent can collect data from many MongoDB instances.

Add Hosts to MMS Monitoring

The Monitoring Agent automatically discovers MongoDB processes based on existing cluster configuration. You'll have to manually seed at least one of these hosts from the MMS console.

To add a host to On Prem MMS Monitoring, click the *ADD HOST* button at the top of the Hosts page. This displays the *New Host* interface. Enter the hostname, port, and optionally an `admin` database username and password. Then select *Add* to submit data.

If the host is only accessible by specific hostname or IP address, or you need to specify the hostname to use for servers with multiple aliases, set up a preferred hostname. See the *Preferred Hostnames* section for details.

Once it has a seed host, the Monitoring Agent will discover any other nodes from associated clusters. These clusters, and their respective seed hosts, include:

- Master databases, after adding slave databases.
- Shard clusters, after adding `mongos` instances.
- Replica sets, after adding any member of the set.

Once you add these seed nodes, the Monitoring Agent will fetch this information *from* the MMS servers. Thus, when configuring the monitoring environment, you may need to wait for several update cycles (e.g. 5-10 minutes) to complete the auto-discovery process and host identification. During this period, you may see duplicate hosts in the MMS web console. This is normal.

The Monitoring Agent fetches configuration and reports to On Prem MMS Monitoring every minute, so, again, there may be a delay of several minutes before data and host information propagate to the MMS console.

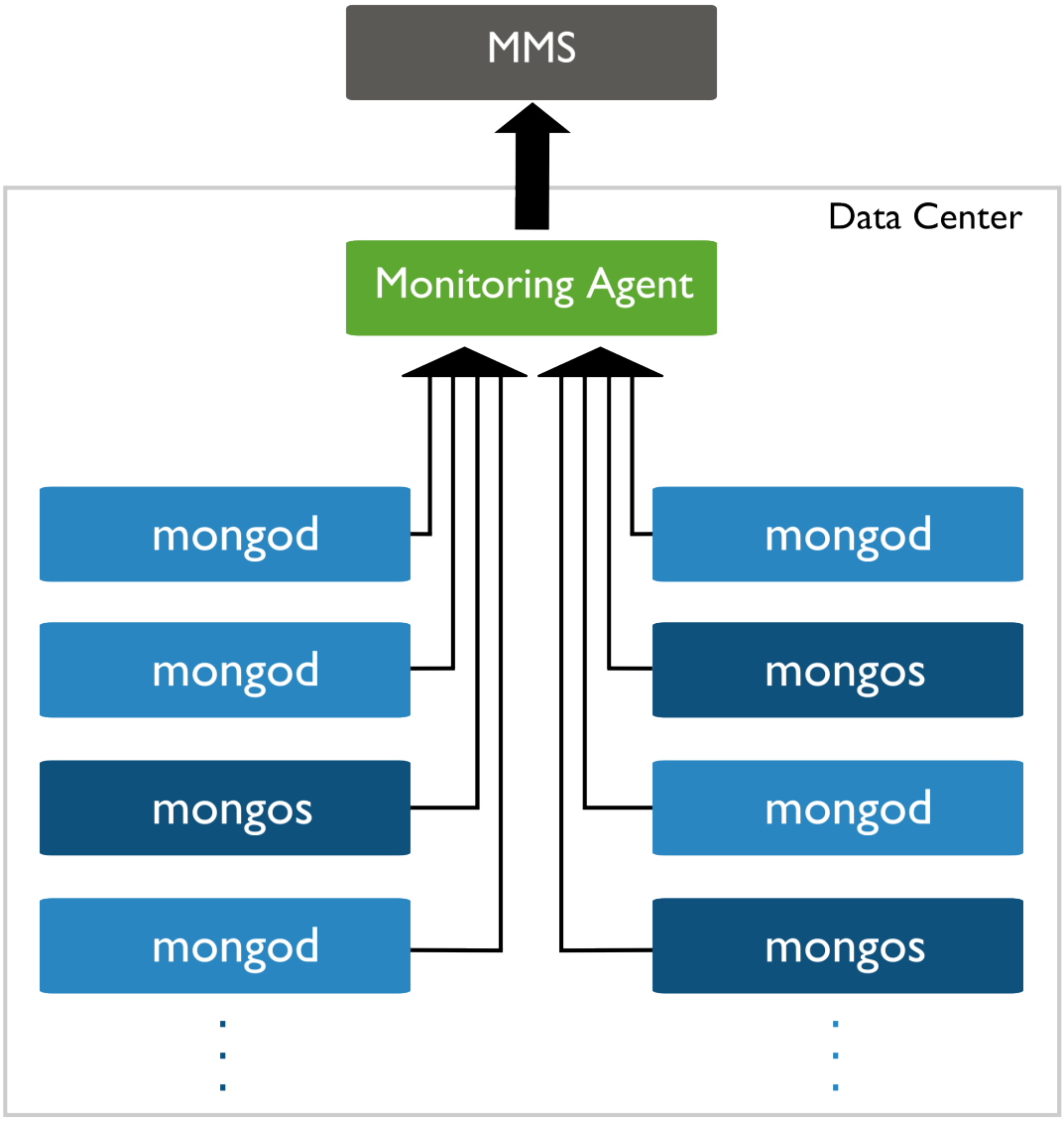


Figure 6: One Monitoring Agent can pull from multiple data centers and instances then push data to MMS.

You can find immediate evidence of a working installation in the agent output or logs. For more information, check the MMS console's *Monitoring Agent Log* tab on the *Hosts* page, as well as the *Last Ping* and *Daily Pings* tabs visible when you click a hostname on the *Hosts*, *Mongos*, and *Configs* tabs on the *Hosts* page. Once On Prem MMS Monitoring has data, you can view and begin using the statistics.

seealso:: *Manage Hosts*

4.2 Using MMS Monitoring

Manage Hosts Procedures to create host aliases as well as remove hosts.

View Aggregated Cluster Statistics Compare hosts dynamically across the cluster.

View Replica Set Statistics Compare hosts dynamically across a replica set.

Monitoring Configuration Discusses configurable options with On Prem MMS Monitoring: hardware monitoring with `munin-node` and using On Prem MMS Monitoring with MongoDB instances running with SSL.

Diagnostic and Troubleshooting Guide Troubleshooting advice for common issues encountered with On Prem MMS Monitoring.

Delete Monitoring Agents Remove an existing Monitoring Agents from an MMS group.

Manage Hosts

MMS provides access to hosts and host management through the *Monitoring* tab.

View Host Statistics

Host statistics display the data collected by the Monitoring Agent and provide tools for working with the data.

Step 1: Select the *Monitoring* tab and then select *Hosts*.

Step 2: Click the name of the host for which to view statistics. MMS displays the collected host statistics.

Manage Host Labels

You can label hosts to narrow the list of hosts in the MMS display. This helps if you monitor a large number of hosts and want to access a more limited set of hosts. Hosts may belong to none, one, or multiple labels.

Labels also are useful as aliases if your machines have existing hostnames that do not sufficiently describe the system in the context of MMS. In most cases, hosts are automatically aliased during auto-discovery.

Add or Edit Labels To add or edit a host label:

1. Select the *Monitoring* tab and then select either *Hosts*, *Mongos*, or *Configs*.
2. Click the *all hosts* link directly above the *Last Ping* column heading.
3. Select *Edit Labels* from the drop down lists. The *Host Labels* interface appears.
4. To edit or delete an existing host label, click the pencil icon or trash icon.
5. To add a new host label, type the name and click the *Add Label* button.

When you add or update a Host Label value, all previous hostname aliases are reset.

Associate Labels with Hosts To add or tag a host with a label:

1. Select the *Monitoring* tab and then select either *Hosts*, *Mongos*, or *Configs*.
2. To the right of any Host, click the gear icon in the far right column of the hosts table. Select *Edit Host*.
3. The *Edit Host* interface appears. Click the *Host Labels* tab.
4. Click to select one or more labels to assign to the host.

Display Hosts by Label To display hosts by their host label:

1. Select the *Monitoring* tab and then select either *Hosts*, *Mongos*, or *Configs*.
2. Click the *all hosts* link directly above the *Last Ping* column heading.
3. Select the appropriate host label. The list of hosts will display only hosts tagged with the selected label.

Reactivate Hosts

If the Monitoring Agent cannot collect information from a MongoDB instance, MMS deactivates the instance in the Monitoring Agent and in the console. MMS deactivates a `mongos` that is unreachable for 24 hours and a `mongod` that is unreachable for 7 days.

When the system deactivates an instance, the corresponding MMS *Monitoring* page marks the deactivated instance with an `x` in the *LAST PING* column. If the instance is a `mongod`, MMS displays a caution icon at the top of each *Monitoring* page.

You can reactivate a deactivated instance whether or not it is running. When you reactivate a host, the Monitoring Agent has an hour to reestablish contact and provide a ping to MMS. If a host is running and reachable, it appears marked with a green circle in the *LAST PING* column. If it is unavailable, it appears marked with a red square. If it remains unreachable for an hour, MMS deactivates it again.

You can optionally remove a host that you are no longer using. Removed hosts are permanently hidden from MMS. For more information, see *Remove Hosts*.

To reactivate a host:

Step 1: Select the *Monitoring* tab and then select either *Hosts*, *Mongos*, or *Configs*.

Step 2: Click the warning icon at the top of the page.

Step 3: Click *Reactivate ALL hosts*.

Step 4: Add the `mongos` instances.

Remove Hosts

You can remove hosts that you no longer use, but when you do they are hidden permanently. If you run the instance again, MMS will not discover it. If you choose to *add* the host again, MMS will **not** display it.

Only a global administrator can undelete the host so that it will again display if added. The administrator can add the host back through the *Deleted Hosts* tab on the MMS *Admin* interface.

Instead of removing a host, you can optionally disable alerts for the host, which does not remove it from the monitoring pages. See *Manage Host Alerts*.

To remove a host from MMS:

Step 1: Select the *Monitoring* tab and then select either *Hosts*, *Mongos*, or *Configs*.

Step 2: On the line listing the host, click the gear icon and select *Remove Host*.

Step 3: Click *Delete*.

Step 4: If prompted for a two-factor authentication code, enter it, click *Verify*, and then click *Delete* again.

Manage Host Alerts

Step 1: Select the *Monitoring* tab and then select either *Hosts*, *Mongos*, or *Configs*.

Step 2: On the line listing the host, click the gear icon and select *Edit Host*.

Step 3: Select *Alert Status* and modify alert settings.

Profile Databases

On Prem MMS Monitoring can collect data from MongoDB's profiler to provide statistics about performance and database operations.

Before enabling profiling, be aware of these issues:

- Profile data can include sensitive information, including the content of database queries. Ensure that exposing this data to On Prem MMS Monitoring is consistent with your information security practices.
- The profiler can consume resources which may adversely affect MongoDB performance. Consider the implications before enabling profiling.

To allow On Prem MMS Monitoring to collect profile data for a specific host:

Step 1: Select the *Monitoring* tab and then select either *Hosts*, *Mongos*, or *Configs*.

Step 2: On the line for any host, click the gear icon and select *Edit Host*.

Step 3: On the *Edit Host* interface, click the *Profiling* tab.

Step 4: Click the *On button* to enable transmission of database profile statistics to MMS.

Step 5: Start database profiling by using the mongo shell to modify the `setProfilingLevel` command. See the database profiler documentation for instructions for using the profiler.

Note: The Monitoring Agent attempts to minimize its effect on the monitored systems. If resource intensive operations, like polling profile data, begins to impact the performance of the database, On Prem MMS Monitoring will throttle the frequency that it collects data. See *How does MMS gather database statistics?* for more information about the agent's throttling process.

When enabled, On Prem MMS Monitoring samples profiling data from monitored instances: the agent only sends the most recent 20 entries from last minute.

With profiling enabled, all configuration changes made in MMS can take up to 2 minutes to propagate to the agent and another minute before profiling data appears in the MMS interface.

When profiling is on, the *Profile Data* tab on the *Host Statistics page* displays the profile levels used to collect data. For more information on profile levels, see /tutorial/manage-the-database-profiler.

If you have profiling data and wish to delete it from MMS, use the button on the bottom of the *Profile Data* tab for deleting profile data. When you click on this button, MMS raises a confirmation dialogue. When you confirm, On Prem MMS Monitoring will begin removing stored profile data from this server's record.

Note: If On Prem MMS Monitoring is storing a large amount of profile data for your instance, the removal process will not be instantaneous.

Additional Information

- *Add Hosts to MMS Monitoring*
- *View Aggregated Cluster Statistics*
- *View Replica Set Statistics*
- *Diagnostic and Troubleshooting Guide*

View Aggregated Cluster Statistics

Overview

Cluster statistics provide an interface to view data for an entire cluster at once. You can compare components dynamically across the cluster and view host-specific and aggregated data, as well as pinpoint moments in time and isolate data to specific components.

Procedure

To view cluster statistics:

Step 1: Select the *Monitoring* tab and then select *Hosts*.

Step 2: In the *CLUSTER* column, click the name of the *sharded cluster*. MMS displays a chart and table with an initial set of cluster statistics. At the top of the chart, the *DATA SIZE* field measures the cluster's data size on disk. For more information, see the explanation of dataSize on the dbStats page.

The bell icon shows if the cluster has alerts. Click the icon to view alerts.

If `Backup` </bacup> is enabled, hover the mouse pointer over the "clock" icon to view the time of the last snapshot and time of the next scheduled snapshot. Click the icon to view snapshots.

Step 3: Select the components to display. In the buttons above the chart, select whether to display the cluster's *shards*, *mongos* instances, or *config servers*.

If you select *shards*, select whether to display *Primaries*, *Secondaries*, or *Both* using the buttons at the chart's lower right.

The chart displays a different colored line for each component. The table below displays additional data for each component, using the same colors.

Step 4: Select the data to display. Select the type of data in the *CHART* drop-down list. MMS graphs the data for each component individually. You can instead graph the data as an average or sum by clicking the *Averaged* or *Sum* button at the chart's lower right.

Step 5: Change the granularity and zoom. To the right of the chart, select a *GRANULARITY* for the data. The option you select determines the available *ZOOM* options. Whenever you change the granularity, the selected zoom level changes to the closest zoom level available for that granularity.

To zoom further and isolate a specific region of data, click-and-drag on that region of the chart. To reset the zoom level, double-click anywhere on the chart.

Step 6: View metrics for a specific date and time. Move the mouse pointer over the chart to view data for a point in time. The data in the table below the chart changes as you move the pointer.

Step 7: Isolate certain components for display. To remove a component from the chart, click its checkmark in the table below the chart. To again display it, click the checkmark again.

To quickly isolate just a few components from a large number displayed, select the *None* button below the chart and then select the checkmark for the individual components to display. Alternatively, select the *All* button and then deselect the checkmark for individual components not to display.

Step 8: View statistics for a specific component. In the table below the chart, click a component's name to display its statistics page.

If your are viewing shards, you can click the replica set name in the *SHARDS* column to display *replica set statistics*, or you can click the *P* or *S* icon in the *MEMBERS* column to display *host statistics* </core/host-statistics> for a primary or secondary. Hover over an icon for tooltip information.

Step 9: Change the name of the cluster. If you want to change the name of the cluster, hover the mouse pointer over the cluster name. A pencil icon appears. Click the icon and enter the new name.

View Replica Set Statistics

Overview

The Replica set statistics interface makes is possible to view data from all replica set members at once.

Procedure

To view replica set statistics:

Step 1: Select the *Monitoring* tab and then select *Hosts*.

Step 2: In the *REPL SET* column, click the name of the *replica set*. MMS displays a separate chart for each replica set member.

Step 3: Select the members to display. At the top of the page, click *P* to display primaries and *S* to display secondaries. You can display both at once. Hover the mouse pointer over an icon to display tooltip information.

Step 4: Select the granularity and the zoom. At the top of the page, select a *GRANULARITY* for the data. The option you select determines the available *ZOOM* options. Whenever you change the granularity, the selected zoom level changes to the closest zoom level available for that granularity.

To zoom further and isolate a specific region of data, click-and-drag on that region of the chart. All other charts automatically zoom to the same region.

To reset the zoom level, double-click anywhere on the chart.

Step 5: Select the data to display. Select the type of data in the *Add Chart* drop-down list. The charts currently displayed are checked in the drop-down list. Click a chart to add or remove it from the display.

You can alternatively add and remove charts by clicking their buttons at the bottom of the page.

Step 6: View an explanation of a chart's data. Hover the mouse pointer over the name of the chart to display the *i* icon. Click the *i* icon.

Step 7: View metrics for a specific date and time. Move the mouse pointer over the chart to view data for a point in time.

Step 8: Re-order of the charts. To move a chart up or down in the display, hover the mouse pointer over the top left corner until a triangular grabber appears. Click and hold the grabber and move the chart to its new position.

Monitoring Configuration

This document discusses specific configuration options for On Prem MMS Monitoring, including hardware monitoring with `munin-node` and using On Prem MMS Monitoring with SSL.

Hardware Monitoring with `munin-node`

On Prem MMS Monitoring provides support for several plugins for charting hardware statistics collected with Munin. MMS supports the following munin-node plugins:

- `cpu` plugin, which creates the `cputime` chart.
- `iostat` plugin, which creates the `iostat` chart.
- `iostat_ios` plugin, which creates the `iotime` chart.

Install the `munin-node` Package You must install the `munin-node` package on all of the host systems that you wish to monitor. Ensure that the Monitoring Agent can connect to the `munin-node` process on port 4949 of the monitored host to collect data.

Note: `munin-node`, and hardware monitoring is only available for MongoDB instances running on Linux hosts.

On Debian and Ubuntu systems, issue the following command to install `munin-node`:

```
sudo apt-get install munin-node
```

To install `munin-node` on Red Hat, CentOS, and Fedora systems, issue the following command:

```
yum install munin-node
```

Note: For Red Hat and CentOS 6.8 systems, you will need to install the EPEL repository **before** installing `munin-node`. To install the EPEL repository, issue the following command:

```
rpm -Uvh http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
```

Configure `munin-node` When installation is complete, ensure that `munin-node`:

- is running. Use the command, "`ps -ef | grep "munin"`" to confirm. If the process is not running, issue the command "`/etc/init.d/munin-node start`".
- will start following the next system reboot. This is the default behavior on most Debian-based systems. Red Hat and related distributions should use the "`chkconfig`" command, to configure this behavior (i.e. "`chkconfig munin-node on`")
- is accessible from the system running the agent. `munin-node` uses port 4949, which needs to be open on the monitored system, so the agent can access this data source. Use the following procedure to test access:

  ```
  telnet [HOSTNAME] 4949
  fetch iostat
  fetch iostat_ios
  fetch cpu
  ```

  Replace `[HOSTNAME]` with the hostname of the monitored system. Run these commands from the system where the Monitoring Agent is running. If these "`fetch`" commands return data, then `munin-node` is running and accessible by the Monitoring Agent.

Note: On some platforms, `munin-node` does not have all required plugins enabled.

For CentOS and Ubuntu, the `munin-node` package does not have the `iostat` and `iostat_ios` plugins enabled. Use the following operation to enable these plugins:

```
sudo ln -s /usr/share/munin/plugins/iostat /etc/munin/plugins/iostat
sudo ln -s /usr/share/munin/plugins/iostat_ios /etc/munin/plugins/iostat_ios
sudo /etc/init.d/munin-node restart
```

If `munin-node` is running but inaccessible, make sure that you have access granted for the system running the Monitoring Agent and that no firewalls block the port between `munin-node` and the Monitoring Agent. You may find the `munin-node` configuration at `/etc/munin-node/munin-node.conf`, `/etc/munin/munin-node.conf`, or `/etc/munin-node.conf`, depending on your distribution.

Additional Considerations for `munin-node`

- If you have numbered disk devices (e.g. `http://mms.mongodb.com/helpdev/sda1` and `http://mms.mongodb.com/helpdev/sda2`) then you will need to configure support for numbered disk in the munin iostat plugin. Find the configuration file at `/etc/munin/plugin-conf.d/munin-node` or a similar path, and add the following value:

  ```
  [iostat]
      env.SHOW_NUMBERED 1
  ```

- If you have Munin enabled and do not have `iostat ios` data in your Munin charts, your `munin-node` may not have write access to required state files in its `munin/plugin-state/` directory. See the `munin-node` plugin log (i.e. `/var/log/munin/munin-node.log` or similar depending on your distribution) for more information.

  The full path of this state directory depends on the system, but is typically `/var/lib/munin/plugin-state/`. Run the following command sequence to correct this issue:

  ```
  touch /var/lib/munin/plugin-state/iostat-ios.state
  chown -R [username]:[group] /var/lib/munin/plugin-state/
  chmod -R 660 /var/lib/munin/plugin-state/
  ```

  Replace `[username]` and `[group]` with the username and group that the `munin-node` process runs with.

- Add the host running the Monitoring Agent to the *allow* directive in the `/etc/munin-node/munin-node.conf` file. The *allow* directive lists hosts allowed to query the munin-node process. Otherwise, traffic from the MMS host will be allowed via firewall but will not be collected by munin.

If you encounter any other problems, check the log files for `munin-node` to ensure that there are no errors with Munin. `munin-node` writes logs files in the `/var/log/` directory on the monitored system.

See also:

The Muinin Diagnostics section on the *Diagnostic and Troubleshooting Guide* page.

Using SSL with On Prem MMS Monitoring

On Prem MMS Monitoring can monitor MongoDB instances running with SSL. To use SSL with `mongod` and `mongos`, you must enable it at compile time, or use one of the subscriber builds. MongoDB added SSL support in version 2.0.

The Monitoring Agent must be configured with the trusted CA certificates used to sign the certificates used by any MongoDB instances running with SSL. Edit the `monitoring-agent.config` file in your agent installation to set `sslTrustedServerCertificates` to the path of a file containing one or more certificates in PEM format.

By default, the agent will only connect to MongoDB instances using a trusted certificate. For testing purposes the `sslRequireValidServerCertificates` setting can be set to `False` to bypass this check. This configuration is NOT recommended for production use as it makes the connection insecure.

To monitor a host with SSL enabled, you can either:

1. Edit the `monitoring-agent.config` file in your agent installation, so that the `useSslForAllConnections` value is `True`, as follows:

   ```
   useSslForAllConnections = True
   ```

   Then restart the Monitoring Agent. After restarting the agent you may observe a five minute delay before On Prem MMS Monitoring receives data from the agent.

2. Enable support on a per-host basis in the MMS console by clicking on the edit (i.e. "Pencil") button on the right hand-side of the "Hosts" page. In the dialogue that pops up, click the check-box on the SSL tab.

If you enable SSL support globally you **will not** be able to override this setting on a per-host basis.

Next Steps with On Prem MMS Monitoring

Take this opportunity to explore the MMS interface. For a detailed explanation of the pages that form the MMS console, continue to the *usage documentation*. You may also want to consult the *troubleshooting guide*.

Diagnostic and Troubleshooting Guide

This document provide troubleshooting advice for common issues encountered installing the Monitoring Agent. Begin by working through the checklist below to ensure issues are not easily resolved. Questions and answers also are listed below for issues not caused by easily fixed installation or connectivity problems.

For answers to other questions, see the *monitoring FAQ*.

Getting Started Checklist

Most problems with MMS are the result of issues with installation, connectivity, and other problems easily resolved. To begin troubleshooting, complete these tasks:

1. *Authentication Errors*
2. *Check Agent Output or Log*
3. *Confirm Only One Agent is Active and Running*
4. *Ensure Connectivity Between Agent and Monitored Hosts*
5. *Ensure Connectivity Between Agent and MMS Server*
6. *Allow Agent to Discover Hosts and Collect Initial Data*

Installation

Authentication Errors If your MongoDB instances run with authentication enabled, ensure MMS has these credentials. For new hosts, click the *Add Host* button on the *Hosts* page then specify credentials for every host with authentication enabled. For hosts already listed in MMS, click the *gear icon* to the right of a host name on the *Host* page then select *Edit Host* to provide credentials.

Please *consult the Authentication Requirements documentation* for details about how to use authentication.

Agent

Check Agent Output or Log If you continue to encounter problems, check the agent's output or logs for errors.

Confirm Only One Agent is Active and Running If *running multiple Monitoring Agents*, then to determine which agent is the primary agent look at the Last Ping value in the *Monitoring Agents* tab on the *Hosts* page. If there is no Last Ping value for a listed agent, the agent is a standby agent.

When you upgrade a Monitoring Agent, do not forget to kill the old hot standby agent.

If you run a primary agent and a hot standby agent, confirm both agents are the same version.

See *Frequently Asked Questions: Monitoring* and *Add Hosts to MMS Monitoring* for more information.

Ensure Connectivity Between Agent and Monitored Hosts Ensure the system running the agent can resolve and connect to the MongoDB instances. To confirm, log into the system where the agent is running and issue a command in the following form:

```
mongo [hostname]:[port]
```

Replace `[hostname]` with the hostname and `[port]` with the port that the database is listening on.

Ensure Connectivity Between Agent and MMS Server Verify that the Monitoring Agent can connect on TCP port 443 (outbound) to the MMS server (i.e. "`mms.mongodb.com`".)

Allow Agent to Discover Hosts and Collect Initial Data Allow the agent to run for 5-10 minutes to allow host discovery and initial data collection.

Alerts

How to Set Up Alerts In MMS, click the *Activity* tab then *Alert Settings*. Click the *Add Alert* button to add an alert. On the *Create a New Alert* window, select the alert conditions, for which hosts, and where to send alerts.

See *Activity: Alerts and Events* for details.

Cannot Turn Off Email Notifications There are at least two ways to turn off alert notifications:

Remove the host from your MMS account. Click the *Hosts* tab then click the *gear icon* to the right of a host name and select *Remove Host*.

Disable or delete the alert in MMS. Click the *Activity* tab then click *Alert Settings*. To the right of an alert, select the *gear icon* and select *Disable* or *Delete*.

Receive Duplicate Alerts If the notification email list contains multiple email-groups, one or more people may receive multiple notifications of the same alert.

Receive "Host has low open file limits" or "Too many open files" error messages These error messages appear on the *Hosts*, *Mongos*, and *Configs* pages under the Host name. They appear if the number of available connections does not meet an MMS-defined minimum value. These errors are not generated by the `mongos` instance and, therefore, will not appear in `mongos` log files.

On a host by host basis, the Monitoring Agent compares the number of open file descriptors and connections to the maximum connections limit. The max open file descriptors ulimit parameter directly affects the number of available server connections. The agent calculates whether or not enough connections exist to meet an MMS-defined minimum value.

In ping documents, for each node and its `serverStatus.connections` values, if the sum of the `current` value plus the `available` value is less than the `maxConns` configuration value set for a monitored host, the Monitoring Agent will send a *Host has low open file limits* or *Too many open files* message to MMS.

Ping documents are data sent by Monitoring Agents to MMS. To view ping documents, click the *Hosts* page, the *Host Name*, and the *Last Ping* tab on the charts page for the selected host.

To prevent this error, we recommend you set `ulimit` open files to 64000. We also recommend setting the `maxConns` command in the mongo shell to at least the recommended settings.

See the MongoDB ulimit reference page and the the MongoDB maxConns reference page for details.

Hosts

Hosts are not Visible Problems with the Monitoring Agent detecting hosts can be caused by a few factors.

Host not added to MMS: In MMS, click the *Hosts* tab then click the *Add Host* button. In the *New Host* window, specify the host type, internal hostname, and port. If appropriate, add the database username and password and whether or not MMS should use SSL to connect with your Monitoring Agent. Note it is not necessary to restart your Monitoring Agent when adding (or removing) a host.

Accidental duplicate mongods If you add the host after a crash and restart the Monitoring Agent, you might not see the hostname in the MMS *Mongos* page. MMS detects the host as a duplicate and suppresses its data. To reset, select *Settings* then *Group Settings*. Click the *Reset Duplicates* button.

Too many Monitoring Agents installed: Only one Monitoring Agent is needed to monitor all hosts within a single network. You can use a single Monitoring Agent if your hosts exist across multiple data centers and can be discovered by a single agent. Check you have only one Monitoring Agent and remove old agents after upgrading the Monitoring Agent.

A second Monitoring Agent can be set up for redundancy. However, the MMS Monitoring Agent is robust. MMS sends an *Agent Down* alert only when there are no available Monitoring Agents available. See *Monitoring FAQ* and *Monitoring Architecture* for more information.

Cannot Delete a Host In rare cases, the `mongod` is brought down and the replica set is reconfigured. The down host cannot be deleted and returns an error message, "This host cannot be deleted because it is enabled for backup." Contact MMS Support for help in deleting these hosts.

For more information on deleted hosts, see *Remove Hosts*.

Groups

Cannot Delete a Group Please contact your MMS administrator to remove a company or group from your MMS account.

How to Add a Group Create a group to monitor additional segregated systems or environments for servers, agents, users, and other resources. For example, your deployment might have two or more environments separated by firewalls. In this case, you would need two or more separate MMS groups.

API and shared secret keys are unique to each group. Each group requires its own agent with the appropriate API and shared secret keys. Within each group, the agent needs to be able to connect to all hosts it monitors in the group.

See *User and Environment Management* for more details.

Step 1: In MMS, select the *Users* tab.

Step 2: Click the *Add New Group* button.

Step 3: Add the group. In the *Group Name* box, type a name for the new group and then click *Add New Group*. For security and auditing reasons, you cannot use a name used earlier. Once you name a group, the group's name cannot be changed.

Step 4: Open the group. To access the new group, select the *Group* box at the top of the MMS interface, type the group's name, and select the group. You are the first user added to the new group.

Step 5: Assign hosts. In the *Monitoring* section, click *Get Started*. Follow the prompts to download the agent, if you have not already, and to assign hosts to the group.

Monitoring Server

Why doesn't the monitoring server startup successfully? If you use authentication, whether or not you enable backup, confirm these properties are in the `<install_dir>/conf/conf-mms.properties` file:

```
mongo.mongoUri=<SetToValidUri>
mongo.replicaSet=<ValidRSIfUsed>
```

Otherwise, MMS will fail while trying to connect to the default 127.0.0.1:27017 URL.

If you use the MMS `<install_dir>/bin/credentialstool` to encrypt the password used in the `mongo.mongoUri` value, also add the `mongo.encryptedCredentials` key to the `<install_dir>/conf/conf-mms.properties` file and set the value for this property to true:

```
mongo.encryptedCredentials=true
```

For more details, see *MongoDB Access Control Considerations*.

Munin

Install and configure the `munin-node` daemon on the monitored MongoDB server(s) before starting MMS monitoring. The MMS agent README file provides guidelines to install `munin-node`. However, new versions of Linux, specifically Red Hat Linux (RHEL) 6, can generate error messages. See *Configure MMS Monitoring* for details about monitoring hardware with `munin-node`.

Restart `munin-node` after creating links for changes to take effect.

"No package munin-node is available" Error To correct this error, install the most current version of the Linux repos. Type these commands:

```
sudo yum install http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
```

Then type this command to install `munin-node` and all dependencies:

```
sudo yum install munin-node
```

Non-localhost IP Addresses are Blocked By default, munin blocks incoming connections from non-localhost IP addresses. The `/var/log/munin/munin-node.log` file will display a "Denying connection" error for your non-localhost IP address.

To fix this error, open the `munin-node.conf` configuration file and comment out these two lines:

```
allow ^127\.0\.0\.1$
allow ^::1$
```

Then add this line to the `munin-node.conf` configuration file with a pattern that matches your subnet:

```
cidr_allow 0.0.0.0/0
```

Restart `munin-node` after editing the configuration file for changes to take effect.

Verifying iostat and Other Plugins/Services Returns "# Unknown service" Error The first step is to confirm there is a problem. Open a telnet session and connect to `iostat`, `iostat_ios`, and `cpu`:

```
telnet HOSTNAME 4949 <default/required munin port>
fetch iostat
fetch iostat_ios
fetch cpu
```

The `iostat_ios` plugin creates the `iotime` chart, and the `cpu` plugin creates the `cputime` chart.

If any of these telnet `fetch` commands returns an "# Unknown Service" error, create a link to the plugin or service in /etc/munin/plugins/ by typing these commands:

```
cd /etc/munin/plugins/
sudo ln -s /usr/share/munin/plugins/<service> <service>
```

Replace `<service>` with the name of the service that generates the error.

Disk names are not listed by Munin In some cases, Munin will omit disk names with a dash between the name and a numerical prefix, for example, `dm-0` or `dm-1`. There is a documented fix for Munin's iostat plugin.

Two-Factor Authentication

Missed SMS Authentication Tokens Unfortunately SMS is not a 100% reliable delivery mechanism for messages, especially across international borders. The Google authentication option is 100% reliable. Unless you must use SMS for authentication, use the Google Authenticator application for two-factor authentication.

If you do not receive the SMS authentication tokens:

1. Refer to the *Settings* page for more details about using two-factor authentication. This page includes any limitations which may affect SMS delivery times.
2. Enter the SMS phone number with country code first followed by the area code and the phone number. Also try 011 first followed by the country code, then area code, and then the phone number.

If you do not receive the authentication token in a reasonable amount of time contact MMS Support to rule out SMS message delivery delays.

How to Delete or Reset Two-Factor Authentication Contact your system administrator to remove or reset two-factor authentication on your account.

For administrative information on two-factor authentication, see *Manage Two-Factor Authentication for On Prem MMS*.

Delete Monitoring Agents

The On-Prem MongoDB Management Service adds a Monitoring Agent when the agent reports to the service upon startup. MMS does not send commands to a Monitoring Agent. Nor can the MMS stop a Monitoring Agent from reporting data.

The On-Prem MongoDB Management Service removes an agent when the agent does not report to the service for more than 24 hours. A stopped or inactive Monitoring Agent will not appear in the list of agents on the *Monitoring Agents* tab on the *Hosts* page.

There is no way to delete a Monitoring Agent from MMS except to remove a Monitoring Agent on your server then wait 24 hours.

If you delete a Monitoring Agent by removing it from your environment, thus removing it from MMS, also delete any alerts for the Monitoring Agent. For example, an **Agent Down** alert may trigger if MMS detects 0 Monitoring Agents, you have removed your Monitoring Agent, and have an active alert to notify you when a Monitoring Agent is down.

4.3 Monitoring Operations

Hosts Detailed description of the *Hosts* page of the MMS console, which lists all hosts that are currently being monitored.

Dashboards Instructions for modifying what dashboards are displayed on the *Dashboards* page of the MMS console.

Host Statistics In-depth guide to the *Host Statistics* and the options that you can specify to customize your view.

Hosts

The *Hosts* pages are the primary location for monitoring information in the MMS console. This tabbed interface provides access to all of your monitored objects.

Interface

Tabs The tabs you see depend on the types of processes in your deployment. The main tabs:

- **Hosts** displays all `mongod` instances.
- **Mongos** displays all `mongos` instances.
- **Configs** displays all database configuration servers.
- **Host Mapping** shows the mapping between system hostnames and the names provided by the monitored process (e.g. `mongod` and `mongos`). For more information about creating host aliases and labels, see *Creating Host Aliases*.

The **AGENTS** section of the *Hosts* page includes these tabs:

- **Monitoring Agents** lists the Monitoring Agents attached to this MMS account. For more information, see *Monitoring Agents*.
- **Monitoring Agent Log** displays a log of the Monitoring Agent's activity.

The Hosts Table The *Hosts*, *Mongos*, and *Configs* pages list monitored resources as a table with these headings:

- **Last Ping:** the last time this agent sent a ping to the MMS servers.
- **Host:** the name of the host running the agent. When clicked, charts of this host display.
- **Type:** the type of host, for example, `PRIMARY`, `SECONDARY`, `STANDALONE`, and `ARBITER`. When the host recovers, the rectangle flag turns yellow and displays `RECOVERING`. When the host returns a fatal error, the flag displays `FATAL`. The flag also can display `NO DATA`.
- **Cluster:** the cluster name. Click the cluster name to display aggregated information on the cluster's replica sets. See *View Aggregated Cluster Statistics* for details.
- **Shard:** the cluster name.
- **Repl Set:** the replica set name name. When clicked, charts dispaly for all members of that replica set.
- **Up Since:** the date the host first pinged MMS.
- **Version:** the version of the MongoDB running on this instance.

The Monitoring Agents Table The *Monitoring Agents* tab lists software agent information as a table with these headings:

- **Agent Hostname:** the hostname for the agent.
- **Agent IP:** the IP address of the running agent.
- **Hosts:** the number of MongoDB instances this agent monitors.
- **Ping Count:** the number of pings (i.e. data payloads) sent by the agent since midnight GMT. Typically agents send pings every minute.
- **Conf Count:** the number of configuration requests sent by the agent since midnight GMT. Typically agents request configuration updates every two minutes.
- **Version:** the version of the agent software running on this agent instance.
- **Last Ping:** the date and time of the last ping from the agent.
- **Last Conf:** the last time the agent made a configuration request of the MMS servers.

Remember, if you have more than one Monitoring Agent, only one agent actively monitors MongoDB instances at once. See *Monitoring Architecture* for more information.

Messages An orange yield sign icon displays under a host name on the *Hosts*, *Mongos*, or *Configs* pages when:

- On Prem MMS Monitoring has detected startup warnings for this host. You can see the warning in the last ping for the host.
- On Prem MMS Monitoring suspects the host has a low `ulimit` setting less than `1024`. On Prem MMS Monitoring infers the host's `ulimit` setting using the total number of available and current connections. See the UNIX ulimit Settings reference page.
- On Prem MMS Monitoring flags a deactivated host.

Important: If you have deactivated hosts, review all deactivated hosts to ensure that they are still in use, and remove all hosts that are not active. Then click on the warning icon and select **Reactive ALL hosts**.

Note: If your Monitoring Agent is out of date, it will be highlighted in red on the *Monitoring Agents* tab of the *Hosts* page.

Dashboards

With On Prem MMS Monitoring dashboards, you can create customized collections of charts for easier data analysis. You can configure On Prem MMS Monitoring to automatically load a dashboard rather than the Hosts page from the MMS settings page.

You can create multiple dashboards as your needs dictate. Use the plus icon at the top of the page to specify a name and create a new dashboard, or select "New Dashboard..." when adding a chart to a dashboard . You can rename or remove a dashboard from links on the bottom of a dashboard page.

Adding and Removing Charts from Dashboards

You can add charts from the *Host Statistics* page to your dashboards.

To add a chart to a dashboard, select the arrow above the chart and then select *Add to Dashboard*.

To remove a chart from a dashboard, click the *Dashboard* tab and select the dashboard. Then select the arrow above the chart and select *Remove from Dashboard*.

Advanced Dashboard Creation

When adding a new dashboard, you can select the "`ADD CHART`" button to create a dashboard that includes a custom selection of charts, or a collection of charts from a dynamically assembled list of hosts. From this page, you may create new dashboards or append new charts to existing dashboards. You can filter the included processes by *host type*.

Specify the list of hosts to include in this dashboard by selecting a replica set or shard cluster or writing a regular expression to match monitored processes' hostnames. If checked, the "Host Alias In Regexp" check box allows you to use the regular expression to select the *aliased* hostname you configured, rather than the actual hostname. Below the host configuration options you may toggle an option to "group hosts in chart," which creates a single composite chart for all matching charts.

Below this, there are 17 chart types that you can use to select charts for this dashboard. Below the chart selection, the final row of buttons allows you to: (optionally) test the "host regexp" to ensure that your regular expression is sufficiently selective; preview the charts that this operation will add to the dashboard; and submit these changes to the dashboards.

You can add and remove charts to these dashboards manually. You may also add additional charts using the "advanced create dashboard" functionality by specifying an existing dashboard in the first field.

Zoom to Navigate Chart Data

Click and drag on a chart, horizontally or vertically, to zoom and isolate a specific data region. When a horizontal data region is selected, other charts will automatically zoom to the same region. Double click on a chart to reset zoom level.

Host Statistics

The MMS web console provides an extensive set of features for analyzing the statistics collected by the Monitoring Agent. For a basic overview of the console, see the *usage* documentation. This document provides a more in-depth guide for the "Host Statistics" page of the MMS console.

Accessing the Host Statistics

Step 1: Select the *Monitoring* tab and then select *Hosts*.

Step 2: Click the name of the host for which to view statistics. MMS displays the collected host statistics.

The charts on the host statistics page are interactive and provide tools for navigating and working with On Prem MMS Monitoring data. Click on the "info" buttons with an `i` in a circle to raise informational boxes, to explore the functionality of the MMS console and charts.

Global Page Controls

There are four to seven columns in the line below the MMS console's menu bar. From left to right, these are:

- The time elapsed since the Monitoring Agent collected the last ping response from the host.
- The hostname and port of the mongod instance. Click the hostname to view a page with host performance charts.

- The *host type*. Possible types include: “primary,” “secondary,” “master,” “slave,” “standalone,” “recovering,” and “unknown.”
- The following optional columns appear as needed:
  - The name of the cluster to which this process belongs. Only cluster members display this value. Click the cluster name to view a page with performance charts ordered by shards and mongos within the cluster.
  - The name of the shard cluster to which this process belongs. Only shard cluster members display this value.
  - The name of the replica set to which this process belongs. Only replica set members display this value. Click the replica set name to view a page with performance charts from all members of a set.
- The version of MongoDB running on this process.

The second line contains nine links that control the host statistics page. On Prem MMS Monitoring displays the current selection in a larger font. In the second line there is a “window” selector that narrows the amount of data displayed. These options are:

- “by minute,” which is the default setting. All charts plot one data point per minute. The “window” options are:
  - “one hour,” which is the default window for this selection and charts 1 hour of data.
  - “six hours,” which charts 6 hours of data.
  - “twelve hours,” which charts 12 hours of data.
  - “twenty-four hours,” which charts 24 hours of data.
- “by 5min,” which re-plots all charts with five-minute averages. The “window” options are:
  - “six hours,” which charts 6 hours of data.
  - “twelve hours,” which charts 12 hours of data.
  - “twenty-four hours,” which is the default window for this selection and charts 24 hours of data.
  - “forty-eight hours,” which charts 24 hours of data.
- “by hour,” which re-plots all charts with hourly averages. The “window” options are:
  - “one day,” which charts 1 day of data.
  - “one week,” which charts 1 week of data.
  - “two weeks,” which charts 2 weeks of data.
  - “one month,” which is the default window for this selection and charts 1 month of data.
  - “two months,” which charts 2 months of data.
- “by day,” which re-plots the chart to display a period of time greater than 24 hours. The “window” options are:
  - “one week,” which charts 1 week of data.
  - “two weeks,” which charts 2 weeks of data.
  - “one month,” which is the default window for this selection and charts 1 month of data
  - “six months,” which charts 6 months of data.
  - “one year,” which charts 1 year of data.
- “range,” which allows you to specify a time rage for the charts to display.
- “avg/sec,” which is the default setting. When selected, charts display the average number of events per second.
- “total,” which allows you to re-plot the charts to display the total number of events.

- "gmt," which allows you to re-plot the charts to the GMT zone. Use this option when correlating MMS data with server logs in GMT rather than your local timezone.
- "refresh," which triggers a refresh of all charts.

On the next line, the "multi-chart zoom" slider allows you to change the scope of all charts at once. Move the sliders on either end of this bar to narrow all of the charts on this page at once.

Specific Chart Controls

You may also interact with the charts individually. Using the mouse you can:

- Click-and-drag to select a portion of the chart to zoom into.
- Double-click to revert to the default zoom setting.
- Shift-click-and-drag (i.e. hold the shift key while clicking and dragging) to scroll left and right.

You can control each On Prem MMS Monitoring chart using the buttons at the top right of the chart container in the "chart toolbar." From left to right, these controls are:

- "Add To Dashboard," a plus sign, takes you to a dashboard creation page where you can create a new dashboard and add a collection of charts to the new dashboard.
- "Expand Chart," an icon with two arrows expanding, raises a box with a larger version of the chart.
- "Chart Permalink," a chain, links to a page that only displays this chart.
- "Email Chart," an envelope, raises a dialogue box where you can input an email address and short message to send the chart to an arbitrary email address.
- "View Legend," the character `i` in a circle, raises a box with a key to the chart.

Chart Annotations

Annotations may appear as colored vertical lines on your charts to indicate server events. The following color/event combinations are:

- A *red bar* indicates a server restart.
- A *purple bar* indicates the server is now a primary.
- A *yellow bar* indicates the server is now a secondary.

If you do not wish to see the chart annotations, you can disable them on the "Setting" page.

5 On Prem MMS Backup

On Prem MMS Backup is a system that creates backups of MongoDB replica sets and sharded clusters. A lightweight agent runs within your infrastructure and captures data from your MongoDB deployment using the same mechanism as replication. On Prem MMS Backup hosts the backup data and snapshots using its own infrastructure.

This manual describes the process for registering and activating On Prem MMS Backup, the process for installing the Backup Agent, and the basic procedures for enabling backups and restoring from a backup.

Getting Started with MMS Backup Register for backup, install the Backup Agent, and begin using MMS backup to capture real-time back ups of MongoDB deployments.

Restore MongoDB Instances Use data captured by MMS backup to instantiate MongoDB instances and deployments.

Backup Use and Operation Using and administrating MMS Backup.

5.1 Getting Started with MMS Backup

Install or Update Agent with RPM Packages Install and start the Backup Agent using an `rpm` package.

Install or Update Agent with Debian Packages Install and start the Backup Agent using a `deb` package.

Install or Update Agent on Other Linux Systems Install and start the Backup Agent on other Linux systems using the `tar.gz` archive packages.

Install or Update Agent on OS X Install and start the Backup Agent on OS X.

Install or Update the Backup Agent on Windows Install and start the Backup Agent on Windows.

Activate Backup for a Replica Set Activate On Prem MMS Backup for a Replica Set.

Activate Backup for a Sharded Cluster Activate On Prem MMS Backup for a Sharded Cluster.

Backing up Clusters with Authentication Create a new Backup Agent user to connect the Backup Agent to `mongod` or `mongos` instances with authentication.

Stop, Start, or Disable the MMS Backup Service Suspend, restart or disable On Prem MMS Backup for a cluster or replica set.

Install or Update the Backup Agent with `rpm` Packages

Overview

The On Prem MMS Backup Agent polls the primary MongoDB instance of every backup-enabled replica set and transmits the operations to the On-Prem MongoDB Management Service service.

The Backup Agent relies on the MMS Monitoring Agent to populate the list of sharded clusters and replica sets eligible for backup. If the appropriate hosts are not added, or the Monitoring Agent is not being correctly run, the lists may be incomplete or out-of-date. If you have not already installed and configured On Prem MMS Monitoring, please refer to the *Getting Started with MMS Monitoring* documentation.

Considerations

MongoDB Requirements MMS only supports backing up replica sets and sharded cluster, and does *not* support backing up standalone instances.

MMS only supports backup for replica sets that run MongoDB 2.0 or later.

MMS only supports backup for sharded clusters that run MongoDB 2.4 or later.

All backed up replica sets and config servers must be able to maintain oplog entries, by default, for at least 3 hours over the last 24 hour period. This window is configurable with the `brs.minimumReplicationOplogWindowHr` setting in the `conf-mms.properties` file for the MMS Application server.

Agent Architecture To avoid resource contention, run the agent on a host other than the hosts where the MongoDB instances are running. Be sure the agent can access the MongoDB hosts.

Running on Amazon EC2 If you run the Backup Agent on Amazon EC2, do not use the `t1.micro` instance type, which has a CPU scheduling policy that does not typically provide sufficient capacity to support a Backup Agent for a production deployment. Use a larger instance type instead.

Prerequisites

Install and configure the On Prem MMS Monitoring, as described in the *Getting Started with MMS Monitoring* documentation.

If your MongoDB instances operate within a firewall, configure your network infrastructure to allow outbound connections on port `443` (SSL) to `api-backup.mongodb.com`.

If you use On Prem MMS Backup with a MongoDB deployment that uses authentication, see the following before installing the On Prem MMS Backup Agent:

- *Backing up Clusters with Authentication*
- *MMS Agent Authentication Requirements*

Procedures

This section includes procedures for both installing and updating the Backup Agent on RHEL, CentOS, SUSE, Amazon Linux, and other systems that use `rpm` packages.

Install the Backup Agent with an `rpm` Package Use this procedure to install the agent on RHEL, CentOS, SUSE, Amazon Linux, and other systems that use `rpm` packages.

Step 1: Download the latest version of the Backup Agent package.

```
curl -OL <mmsUri>/download/agent/backup/mongodb-mms-backup-agent-latest.x86_64.rpm
```

Step 2: Install the Backup Agent package. Issue the following command:

```
sudo rpm -U mongodb-mms-backup-agent-latest.x86_64.rpm
```

Step 3: Retrieve the API key for your MMS group. In the *Settings* tab on the *Backup Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your API key. The step displays the actual API key used by your MMS group. Copy the key.

Step 4: Configure the `backup-agent.config` file with the API key. In the `/etc/mongodb-mms/backup-agent.config` file, set the `mmsApiKey` property to your API key.

Step 5: Optional. For SUSE deployments only, configure the `sslTrustedMMSServerCertificate` property. Enter the following property and value in the `/etc/mongodb-mms/backup-agent.config` file:

```
sslTrustedMMSServerCertificate=/etc/ssl/certs/UTN_USERFirst_Hardware_Root_CA.pem
```

Save and close the file.

Step 6: Start the Backup Agent. Issue the following command:

```
sudo service mongodb-mms-backup-agent start
```

Update the Backup Agent with an `rpm` Package Use this procedure to update the agent on RHEL, CentOS, SUSE, Amazon Linux, and other systems that use `rpm` packages.

Step 1: Download the latest version of the Backup Agent package.

```
curl -OL <mmsUri>/download/agent/backup/mongodb-mms-backup-agent-latest.x86_64.rpm
```

Step 2: Install the Backup Agent package. Issue the following command:

```
sudo rpm -U mongodb-mms-backup-agent-latest.x86_64.rpm
```

Step 3: Start the Backup Agent. Issue the following command:

```
sudo service mongodb-mms-backup-agent start
```

Additional Information

The `README` included with the downloaded package also provides information about the Backup Agent.

For details about backup operations, see the *Frequently Asked Questions: Backup* page.

Install or Update the Backup Agent with `deb` Packages

Overview

The On Prem MMS Backup Agent polls the primary MongoDB instance of every backup-enabled replica set and transmits the operations to the On-Prem MongoDB Management Service service.

The Backup Agent relies on the MMS Monitoring Agent to populate the list of sharded clusters and replica sets eligible for backup. If the appropriate hosts are not added, or the Monitoring Agent is not being correctly run, the lists may be incomplete or out-of-date. If you have not already installed and configured On Prem MMS Monitoring, please refer to the *Getting Started with MMS Monitoring* documentation.

Considerations

MongoDB Requirements MMS only supports backing up replica sets and sharded cluster, and does *not* support backing up standalone instances.

MMS only supports backup for replica sets that run MongoDB 2.0 or later.

MMS only supports backup for sharded clusters that run MongoDB 2.4 or later.

All backed up replica sets and config servers must be able to maintain oplog entries, by default, for at least 3 hours over the last 24 hour period. This window is configurable with the `brs.minimumReplicationOplogWindowHr` setting in the `conf-mms.properties` file for the MMS Application server.

Agent Architecture To avoid resource contention, run the agent on a host other than the hosts where the MongoDB instances are running. Be sure the agent can access the MongoDB hosts.

Running on Amazon EC2 If you run the Backup Agent on Amazon EC2, do not use the `t1.micro` instance type, which has a CPU scheduling policy that does not typically provide sufficient capacity to support a Backup Agent for a production deployment. Use a larger instance type instead.

Prerequisites

Install and configure the On Prem MMS Monitoring, as described in the *Getting Started with MMS Monitoring* documentation.

If your MongoDB instances operate within a firewall, configure your network infrastructure to allow outbound connections on port `443` (SSL) to `api-backup.mongodb.com`.

If you use On Prem MMS Backup with a MongoDB deployment that uses authentication, see the following before installing the On Prem MMS Backup Agent:

- *Backing up Clusters with Authentication*
- *MMS Agent Authentication Requirements*

Procedures

This section includes procedures for both installing and updating the Backup Agent on Ubuntu and other systems that use `deb` packages.

Install the Backup Agent with a `deb` Package Use this procedure to install the agent on Ubuntu and other systems that use `deb` packages.

Step 1: Download the latest version of the Backup Agent package. With a system shell, issue the following command:

```
curl -OL <mmsUri>/download/agent/backup/mongodb-mms-backup-agent_latest_amd64.deb
```

Step 2: Install the Backup Agent package. Issue the following command:

```
sudo dpkg -i mongodb-mms-backup-agent_latest_amd64.deb
```

Step 3: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Backup Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 4: Configure the `backup-agent.config` file with the API key. In the `/etc/mongodb-mms/backup-agent.config` file, set the `mmsApiKey` property to your API key.

Step 6: Start the Backup Agent. Issue the following command:

```
sudo start mongodb-mms-backup-agent
```

Update the Backup Agent with a `deb` Package Use this procedure to update the agent on Ubuntu and other systems that use `deb` packages.

Step 1: Download the latest version of the Backup Agent package. With a system shell, issue the following command:

```
curl -OL <mmsUri>/download/agent/backup/mongodb-mms-backup-agent_latest_amd64.deb
```

Step 2: Install the Backup Agent package. Issue the following command:

```
sudo dpkg -i mongodb-mms-backup-agent_latest_amd64.deb
```

Additional Information

The `README` included with the downloaded package also provides information about the Backup Agent.

For details about backup operations, see the *Frequently Asked Questions: Backup* page.

Install or Update the Backup Agent from an Archive

Overview

The On Prem MMS Backup Agent polls the primary MongoDB instance of every backup-enabled replica set and transmits the operations to the On-Prem MongoDB Management Service service.

The Backup Agent relies on the MMS Monitoring Agent to populate the list of sharded clusters and replica sets eligible for backup. If the appropriate hosts are not added, or the Monitoring Agent is not being correctly run, the lists may be incomplete or out-of-date. If you have not already installed and configured On Prem MMS Monitoring, please refer to the *Getting Started with MMS Monitoring* documentation.

Considerations

MongoDB Requirements MMS only supports backing up replica sets and sharded cluster, and does *not* support backing up standalone instances.

MMS only supports backup for replica sets that run MongoDB 2.0 or later.

MMS only supports backup for sharded clusters that run MongoDB 2.4 or later.

All backed up replica sets and config servers must be able to maintain oplog entries, by default, for at least 3 hours over the last 24 hour period. This window is configurable with the `brs.minimumReplicationOplogWindowHr` setting in the `conf-mms.properties` file for the MMS Application server.

Agent Architecture To avoid resource contention, run the agent on a host other than the hosts where the MongoDB instances are running. Be sure the agent can access the MongoDB hosts.

Running on Amazon EC2 If you run the Backup Agent on Amazon EC2, do not use the `t1.micro` instance type, which has a CPU scheduling policy that does not typically provide sufficient capacity to support a Backup Agent for a production deployment. Use a larger instance type instead.

Prerequisites

Install and configure the On Prem MMS Monitoring, as described in the *Getting Started with MMS Monitoring* documentation.

If your MongoDB instances operate within a firewall, configure your network infrastructure to allow outbound connections on port `443` (SSL) to `api-backup.mongodb.com`.

If you use On Prem MMS Backup with a MongoDB deployment that uses authentication, see the following before installing the On Prem MMS Backup Agent:

- *Backing up Clusters with Authentication*
- *MMS Agent Authentication Requirements*

Procedures

This section includes procedures for both installing and updating the Backup Agent on Linux or Mac OSX.

Install the Backup Agent from a `tar.gz` Archive Use this procedure to install the agent on Linux or Mac OSX.

Step 1: Download the latest version of the Backup Agent archive. On a system shell, issue a command that resembles the following. Replace `linux_x86_64` with your platform, as needed: depending on your operating system:

```
curl -OL <mmsUri>/download/agent/backup/mongodb-mms-backup-agent-latest.linux_x86_64.tar.gz
```

Step 2: Install the Backup Agent. To install the agent, extract the archive using a command that resembles the following. Replace `linux_x86_64` with your platform, as needed:

```
tar -xf mongodb-mms-backup-agent-latest.linux_x86_64.tar.gz
```

The Backup Agent is installed.

Step 3: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Backup Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 4: Edit the `local.config` file to include your MMS API key. In the directory where you installed the Backup Agent, locate and open the `local.config` file. Enter your API key as the value for the `mmsApiKey` setting.

Step 5: Optional. Configure the agent to use a proxy server. If the agent will connect to MMS via a proxy server, you must specify the server in the `http_proxy` environment variable. To specify the server, use the `export` command, as in the following example:

```
export http_proxy="http://proxy.example.com:9000"
```

To connect through a proxy, you must install the agent from a `.tar.gz` file, *not* from a `.deb` or `.rpm` file.

Step 6: Start the Backup Agent. Issue the following command:

```
nohup ./mongodb-mms-backup-agent >> backup-agent.log 2>&1 &
```

Update the Backup Agent from a `tar.gz` Archive Use this procedure to update the agent on Linux or Mac OSX.

Step 1: Stop any currently running Backup Agents. Issue the following command with the system shell:

```
pkill -f mongodb-mms-backup-agent
```

Step 2: Download the latest version of the Backup Agent archive. On a system shell, issue a command that resembles the following. Replace `linux_x86_64` with your platform, as needed: depending on your operating system:

```
curl -OL <mmsUri>/download/agent/backup/mongodb-mms-backup-agent-latest.linux_x86_64.tar.gz
```

Step 3: Install the Backup Agent. To install the agent, extract the archive using a command that resembles the following. Replace `linux_x86_64` with your platform, as needed:

```
tar -xf mongodb-mms-backup-agent-latest.linux_x86_64.tar.gz
```

The Backup Agent is installed.

Step 4: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Backup Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 5: Edit the `local.config` file to include your MMS API key. In the directory where you installed the Backup Agent, locate and open the `local.config` file. Enter your API key as the value for the `mmsApiKey` setting.

Step 6: Start the Backup Agent. Issue the following command:

```
nohup ./mongodb-mms-backup-agent >> backup-agent.log 2>&1 &
```

Additional Information

If you installed the Backup Agent from the `tar.gz` archives, see `http://mms.mongodb.com/helptutorial/rotate-agen` to configure log rotation.

The `README` included with the downloaded package also provides information about the Backup Agent.

For details about backup operations, see the *Frequently Asked Questions: Backup* page.

Install or Update the Backup Agent on OS X

Overview

The On Prem MMS Backup Agent polls the primary MongoDB instance of every backup-enabled replica set and transmits the operations to the On-Prem MongoDB Management Service service.

The Backup Agent relies on the MMS Monitoring Agent to populate the list of sharded clusters and replica sets eligible for backup. If the appropriate hosts are not added, or the Monitoring Agent is not being correctly run, the lists may be incomplete or out-of-date. If you have not already installed and configured On Prem MMS Monitoring, please refer to the *Getting Started with MMS Monitoring* documentation.

Considerations

MongoDB Requirements MMS only supports backing up replica sets and sharded cluster, and does *not* support backing up standalone instances.

MMS only supports backup for replica sets that run MongoDB 2.0 or later.

MMS only supports backup for sharded clusters that run MongoDB 2.4 or later.

All backed up replica sets and config servers must be able to maintain oplog entries, by default, for at least 3 hours over the last 24 hour period. This window is configurable with the `brs.minimumReplicationOplogWindowHr` setting in the `conf-mms.properties` file for the MMS Application server.

Agent Architecture To avoid resource contention, run the agent on a host other than the hosts where the MongoDB instances are running. Be sure the agent can access the MongoDB hosts.

Running on Amazon EC2 If you run the Backup Agent on Amazon EC2, do not use the `t1.micro` instance type, which has a CPU scheduling policy that does not typically provide sufficient capacity to support a Backup Agent for a production deployment. Use a larger instance type instead.

Prerequisites

Install and configure the On Prem MMS Monitoring, as described in the *Getting Started with MMS Monitoring* documentation.

If your MongoDB instances operate within a firewall, configure your network infrastructure to allow outbound connections on port `443` (SSL) to `api-backup.mongodb.com`.

If you use On Prem MMS Backup with a MongoDB deployment that uses authentication, see the following before installing the On Prem MMS Backup Agent:

- *Backing up Clusters with Authentication*
- *MMS Agent Authentication Requirements*

Procedures

Install the Backup Agent On OS X Use the following procedure to install the agent on OS X:

Step 1: Download the latest version of the Backup Agent archive. On a system shell, issue a command that resembles the following. Replace `linux_x86_64` with your platform, as needed: depending on your operating system:

```
curl -OL <mmsUri>/download/agent/backup/mongodb-mms-backup-agent-latest.osx_x86_64.tar.gz
```

Step 2: Install the Backup Agent. To install the agent, extract the archive using a command that resembles the following. Replace `linux_x86_64` with your platform, as needed:

```
tar -xf mongodb-mms-backup-agent-latest.osx_x86_64.tar.gz
```

The Backup Agent is installed.

Step 3: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Backup Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 4: Edit the `local.config` file to include your MMS API key. In the directory where you installed the Backup Agent, locate and open the `local.config` file. Enter your API key as the value for the `mmsApiKey` setting.

Step 5: Optional. Configure the agent to use a proxy server. If the agent will connect to MMS via a proxy server, you must specify the server in the `http_proxy` environment variable. To specify the server, use the `export` command, as in the following example:

```
export http_proxy="http://proxy.example.com:9000"
```

To connect through a proxy, you must install the agent from a `.tar.gz` file, *not* from a `.deb` or `.rpm` file.

Step 6: Start the Backup Agent. Issue the following command:

```
nohup ./mongodb-mms-backup-agent >> backup-agent.log 2>&1 &
```

Update the Backup Agent Use the following procedure to update the agent on OS X:

Step 1: Download the latest version of the Backup Agent archive. On a system shell, issue a command that resembles the following. Replace `linux_x86_64` with your platform, as needed: depending on your operating system:

```
curl -OL <mmsUri>/download/agent/backup/mongodb-mms-backup-agent-latest.osx_x86_64.tar.gz
```

Step 2: Install the Backup Agent. To install the agent, extract the archive using a command that resembles the following. Replace `linux_x86_64` with your platform, as needed:

```
tar -xf mongodb-mms-backup-agent-latest.osx_x86_64.tar.gz
```

The Backup Agent is installed.

Step 3: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Backup Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 4: Edit the `local.config` file to include your MMS API key. In the directory where you installed the Backup Agent, locate and open the `local.config` file. Enter your API key as the value for the `mmsApiKey` setting.

Step 5: Optional. Configure the agent to use a proxy server. If the agent will connect to MMS via a proxy server, you must specify the server in the `http_proxy` environment variable. To specify the server, use the `export` command, as in the following example:

```
export http_proxy="http://proxy.example.com:9000"
```

To connect through a proxy, you must install the agent from a `.tar.gz` file, *not* from a `.deb` or `.rpm` file.

Step 6: Start the Backup Agent. Issue the following command:

```
nohup ./mongodb-mms-backup-agent >> backup-agent.log 2>&1 &
```

Additional Information

The `README` included with the downloaded package also provides information about the Backup Agent.

For details about backup operations, see the *Frequently Asked Questions: Backup* page.

Install or Update the Backup Agent on Windows

Overview

The On Prem MMS Backup Agent polls the primary MongoDB instance of every backup-enabled replica set and transmits the operations to the On-Prem MongoDB Management Service service.

The Backup Agent relies on the MMS Monitoring Agent to populate the list of sharded clusters and replica sets eligible for backup. If the appropriate hosts are not added, or the Monitoring Agent is not being correctly run, the lists may be incomplete or out-of-date. If you have not already installed and configured On Prem MMS Monitoring, please refer to the *Getting Started with MMS Monitoring* documentation.

Considerations

MongoDB Requirements MMS only supports backing up replica sets and sharded cluster, and does *not* support backing up standalone instances.

MMS only supports backup for replica sets that run MongoDB 2.0 or later.

MMS only supports backup for sharded clusters that run MongoDB 2.4 or later.

All backed up replica sets and config servers must be able to maintain oplog entries, by default, for at least 3 hours over the last 24 hour period. This window is configurable with the `brs.minimumReplicationOplogWindowHr` setting in the `conf-mms.properties` file for the MMS Application server.

Agent Architecture To avoid resource contention, run the agent on a host other than the hosts where the MongoDB instances are running. Be sure the agent can access the MongoDB hosts.

Running on Amazon EC2 If you run the Backup Agent on Amazon EC2, do not use the `t1.micro` instance type, which has a CPU scheduling policy that does not typically provide sufficient capacity to support a Backup Agent for a production deployment. Use a larger instance type instead.

Prerequisites

Install and configure the On Prem MMS Monitoring, as described in the *Getting Started with MMS Monitoring* documentation.

If your MongoDB instances operate within a firewall, configure your network infrastructure to allow outbound connections on port `443` (SSL) to `api-backup.mongodb.com`.

If you use On Prem MMS Backup with a MongoDB deployment that uses authentication, see the following before installing the On Prem MMS Backup Agent:

- *Backing up Clusters with Authentication*
- *MMS Agent Authentication Requirements*

Procedures

Install the Backup Agent On Windows

Step 1: Download and run the latest version of the Backup Agent MSI file. To download the 64-bit MSI file, use the following URL, where `<mmsUri>` is the hostname of the Backup server:

```
<mmsUri>/download/agent/backup/mongodb-mms-backup-agent-latest.windows_x86_64.msi
```

To download the 32-bit MSI file, use the following URL, where `<mmsUri>` is the hostname of the Backup server:

```
<mmsUri>/download/agent/backup/mongodb-mms-backup-agent-latest.windows_i386.msi
```

During installation, the installer prompts you to specify the folder for storing configuration and log files. It is strongly advised that you encrypt or restrict access to this folder.

Step 2: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Backup Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 3: Edit the `local.config` file to include your MMS API key. In the directory where you installed the Backup Agent, locate and open the `local.config` file. Enter your API key as the value for the `mmsApiKey` setting.

Step 4: Edit the `local.config` file to include the hostname of the Backup server. Set the `mothership` property to hostname of the Backup server.

Step 5: Start the Backup Agent. In Windows `Control Panel`, open `Administrative Tools`, and then open `Services`.

In the list of services, select the `MMS Backup Agent` service. Select the `Action` menu and select `Start`.

Update the Backup Agent on Windows

Step 1: Stop all currently running Backup Agents. In Windows `Control Panel`, open `Administrative Tools` and then `Services`. In the list of services, select `MMS Backup Agent`. Select the `Action` menu and select `Stop`.

If you receive a message that your Backup Agent is out of date, make sure you are running an upgradeable version of the Backup Agent. If you are running the version of the Backup Agent named `MongoDBBackup`, you must remove it before upgrading. To check if you are running `MongoDBBackup`, issue the following command in an Administrative command prompt:

```
sc query MongoDBBackup
```

If the command returns a result, you must remove the `MongoDBBackup` agent. To remove it, issue the following:

```
sc delete MongoDBBackup
```

Step 2: Download and run the latest version of the Backup Agent MSI file. To download the 64-bit MSI file, use the following URL, where `<mmsUri>` is the hostname of the Backup server:

```
<mmsUri>/download/agent/backup/mongodb-mms-backup-agent-latest.windows_x86_64.msi
```

To download the 32-bit MSI file, use the following URL, where `<mmsUri>` is the hostname of the Backup server:

```
<mmsUri>/download/agent/backup/mongodb-mms-backup-agent-latest.windows_i386.msi
```

During installation, the installer prompts you to specify the folder for storing configuration and log files. It is strongly advised that you encrypt or restrict access to this folder.

Step 3: Retrieve the MMS API key for your MMS group. In the *Settings* tab on the *Backup Agent* page, click the box for your operating system. MMS will then display a procedure that includes a step to set your MMS API key. The step displays the actual MMS API key used by your MMS group. Copy the key.

Step 4: Edit the `local.config` file to include your MMS API key. In the directory where you installed the Backup Agent, locate and open the `local.config` file. Enter your API key as the value for the `mmsApiKey` setting.

Step 5: Start the Backup Agent. In Windows `Control Panel`, open `Administrative Tools`, and then open `Services`.

In the list of services, select the `MMS Backup Agent` service. Select the `Action` menu and select `Start`.

Additional Information

The `README` included with the downloaded package also provides information about the Backup Agent.

For details about Backup operations, see the *Frequently Asked Questions: Backup* page.

Activate Backup for a Replica Set

Overview

You can start, restart, stop, and terminate backups for replica sets. As appropriate, you also can specify which instance to use for the initial sync, exclude namespaces, authenticate, and change the snapshot schedule and retention.

Note: The replica set must be MongoDB version 2.0 or later.

Considerations

Excluded Namespaces Excluded namespaces are databases or collections that MMS will not back up. Exclude name spaces to prevent backing up collections that contain logging data, caches, or other ephemeral data. By excluding these kinds of databases and collections will allow you to reduce backup time and costs.

Snapshot Frequency and Retention Policy You can take snapshots every 6, 8, 12, or 24 hours and save them for 2-5 days. MMS can retain daily snapshots for up to 365 days, weekly snapshots for up to 52 weeks, and monthly snapshots for up to 36 months.

By default, MMS takes snapshots every 6 hours and stores these for 2 days, for use in point-in-time restores. Also by default, MMS retains daily snapshots for a week, weekly snapshots for a month, and monthly snapshots for a year.

You can set point-in-time restores going back 1, 2, 3, or 4 days.

Changes to the snapshot schedule will affect your snapshot storage costs. The longer your snapshot window, the longer it will take to build a point in time restore.

Prerequisites

Before enabling backup, ensure that the following is true for all replica sets that you back up:

- MMS Monitoring is actively collecting data.
- There is an active *primary*.
- If you explicitly select a sync target, ensure that the sync target is accessible on the network and keeping up with replication.

Procedure

Step 1: Go to the Backup page.

Step 2: Click on the Replica Set Status tab.

Step 3: Select Start in the Action column for your replica set. The Start Backups for Replica Set interface will appear with options to select the `mongod` to use as sync source, configure authentication, and manage any excluded namespaces.

Step 4: Select which instances to use for the initial sync source. To minimize the impact on the primaries, sync off of the secondaries.

Step 5: If needed, click to update authentication for the replica set. Configure replica set authentication credentials through this process to start a backup for a replica set. Or click the gear icon and *Edit Credentials* to the far right of any replica set name on the *Replica Set Status* tab.

Step 6: If needed, click to edit excluded namespaces. Configure excluded namespaces through this process to start a backup for a replica set. Or click the gear icon and *Edit Excluded Namespaces* to the far right of any replica set name on the *Replica Set Status* tab.

Step 7: If needed, click to edit backup snapshot schedule. Configure snapshots through this process to start a backup for a replica set. Or click the gear icon and *Edit Snapshot Schedule* to the far right of any replica set name on the *Replica Set Status* tab.

Step 8: Click the Authenticate, Start, or Restart button. The Start or Restart button will appear if you do not use two-factor authentication to login to MMS. Otherwise, click the Authenticate button to do two-factor authentication and activate backup for the selected replica set.

Your Backup Agent will start polling the specified instance for this replica set.

See the *Frequently Asked Questions: Backup* page for more details about backup configuration, namespaces, authentication, and snapshots.

Activate Backup for a Sharded Cluster

Overview

You can start, restart, stop, and terminate backups for a *sharded cluster*. As appropriate, you also can specify which instance to use for the initial sync, exclude namespaces, authenticate, and change the snapshot schedule and retention.

Note: The sharded-cluster must be MongoDB version 2.4.3 or later.

Considerations

Excluded Namespaces Excluded namespaces are databases or collections that MMS will not back up. Exclude name spaces to prevent backing up collections that contain logging data, caches, or other ephemeral data. By excluding these kinds of databases and collections will allow you to reduce backup time and costs.

Snapshot Frequency and Retention Policy You can take snapshots every 6, 8, 12, or 24 hours and save them for 2-5 days. MMS can retain daily snapshots for up to 365 days, weekly snapshots for up to 52 weeks, and monthly snapshots for up to 36 months.

By default, MMS takes snapshots every 6 hours and stores these for 2 days, for use in point-in-time restores. Also by default, MMS retains daily snapshots for a week, weekly snapshots for a month, and monthly snapshots for a year.

You can set point-in-time restores going back 1, 2, 3, or 4 days.

Changes to the snapshot schedule will affect your snapshot storage costs. The longer your snapshot window, the longer it will take to build a point in time restore.

Checkpoints Checkpoints provide additional restore points between snapshots. With checkpoints enabled, MMS Backup creates restoration points at configurable intervals of every 15, 30 or 60 minutes between snapshots.

To create a checkpoint, MMS Backup stops the *cluster balancer* and inserts a token into the *oplog* of each *shard* and *config server* in the cluster. These checkpoint tokens are lightweight and do not have a consequential impact on performance or disk use.

MMS backup does not require checkpoints: by default, MMS does not enable checkpoints.

Restoring from a checkpoint requires MMS Backup to apply the oplog of each shard and config server to the last snapshot captured before the checkpoint. Restoration from a checkpoint takes longer than restoration from a snapshot.

Prerequisites

Before enabling backup, ensure that the following is true for all replica sets that you back up:

- MMS Monitoring is actively collecting data from your cluster, including from at least one `mongos`.
- All the cluster's *config servers* are running, and the balancing round has completed within the last hour.
- There is an active *primary*.
- If you explicitly select a sync target, ensure that the sync target is accessible on the network and keeping up with replication.

Procedure

Step 1: Go to the Backup page.

Step 2: Click on the Sharded Cluster Status tab.

Step 3: Select Start in the Action column for your sharded cluster. The Start Backups for Sharded Clusters interface will appear with options to select the `mongod` to use as sync source, configure authentication, and manage any excluded namespaces.

Step 4: Select which instances to use for the initial sync source. To minimize the impact on the primaries, sync off of the secondaries.

Step 5: If needed, click to update authentication for the sharded cluster. Configure sharded cluster authentication credentials through this process to start a backup for a sharded cluster. Or click the gear icon and *Edit Credentials* to the far right of any sharded cluster name on the *Sharded Cluster Status* tab.

Step 6: If needed, click to edit excluded namespaces. Configure excluded namespaces through this process to start a backup for a sharded cluster. Or click the gear icon to the far right of any sharded cluster name on the *Sharded Cluster Status* tab.

To specify an excluded namespace, select *Manage excluded namespaces*, then add the names of the databases or collections that you do not want MMS Backup On Prem to include in any restores.

Step 7: If needed, click to edit backup snapshot schedule. Configure snapshots through this process to start a backup for a sharded cluster. Or click the gear icon and *Edit Snapshot Schedule* to the far right of any sharded cluster name on the *Sharded Cluster Status* tab.

Click the *Create a cluster checkpoint* checkbox to set a *checkpoint* for a selected period of time. A cluster checkpoint is a valid point in time when a sharded cluster can be restored. The agent pauses the balancer to create a checkpoint.

Step 8: Click the Authenticate, Start, or Restart button. The Start or Restart button will appear if you do not use two-factor authentication to login to MMS. Otherwise, click the Authenticate button to do two-factor authentication and activate backup for the selected sharded cluster.

Your Backup Agent will start polling the specified instance for this sharded cluster.

Backing up Clusters with Authentication

Overview

The Backup Agent supports MongoDB deployments that require authentication. The agent receives configurations from replica sets, sharded clusters, and hosts, as well as credentials from the MMS *Backup* tab for replica sets and sharded clusters.

If you use On Prem MMS Backup with a MongoDB instance with authentication enabled, there are additional steps you will need to take before you can use On Prem MMS Backup. The Monitoring Agent also will need to be updated to monitor databases using authentication.

Consideration

After you install the Backup Agent, **do not** use the agent's directory location for anything other than the agent itself. The Backup Agent periodically deletes the contents of its root directory.

Prerequisites

You will need to create a new user for the Backup Agent then install the Backup Agent.

Use the **mongo** shell to connect to the **mongod** or **mongos** instance you will be backing up. Create a new user for the On Prem MMS Backup Agent using the appropriate command shown in the *Authentication Requirements* page. Then restart your replica set and sharded cluster with authentication enabled, as appropriate.

You can now install the Backup Agent, as described in the procedures on the *Getting Started with MMS Backup* page. You will need to provide the username and password during the On Prem MMS Backup setup process.

Procedure

Update MMS with Replica Set Username and Password For each replica set using authentication:

Step 1: Go to *Backup* tab in the On-Prem MongoDB Management Service

Step 2: Click *Replica Set Status*

Step 3: Edit Credentials Click the gear icon to the far right of the replica set name and select Edit Credentials.

Step 4: Enter Username and Password for the replica set Click the *Use SSL* checkbox if the **mongod** runs with SSL enabled to backup data. See Connect to MongoDB with SSL for details about how to configure `mongod` to run with SSL enabled.

Update MMS with Sharded Cluster Username and Password For each sharded cluster using authentication:

Step 1: Go to *Backup* tab in the On-Prem MongoDB Management Service

Step 2: Click *Sharded Cluster Status*

Step 3: Edit Credentials Click the gear icon to the far right of the replica set name and select Edit Credentials.

Step 4: Enter Username and Password for the replica set Click the *Use SSL* checkbox if the **mongod** runs with SSL enabled to backup data. See Connect to MongoDB with SSL for details about how to configure `mongod` to run with SSL enabled.

Update Credentials Used by the Monitoring Agent Create a new user for the On Prem MMS Monitoring Agent using the appropriate command shown in the *Authentication Requirements* page.

Stop, Start, or Disable the MMS Backup Service

Overview

Stopping the MMS Backup Service for a replica set or sharded cluster suspends the service for that resource. MMS stops taking new snapshots but retains existing snapshots until their listed expiration date.

After stopping backups, you can restart the Backup Service at any time. Depending how much time has elapsed, the Backup Service may perform an *initial sync*.

Disabling the MMS Backup Service, by contrast, immediately deletes all snapshots. Later, if you want back up the cluster or replica set, when you enable backup, MMS behaves as if the resource has never been backed up. Enabling backups on a previously disabled resource always requires an initial sync.

Procedures

Stop the Backup Service for a Cluster or Replica Set

Step 1: Click the *Backup* tab and then click either *Sharded Cluster Status* or *Replica Set Status*, depending on the resource to stop backup for.

Step 2: Click *Stop* for the cluster or replica set. If prompted, enter the two-factor authentication code, click *Verify*, and click *Stop* again.

Restart the MMS Backup Service

Step 1: Click the *Backup* tab and then click either *Sharded Cluster Status* or *Replica Set Status*, depending on the resource to re-enable.

Step 2: Click *Restart* for the cluster or replica set.

Step 3: Select a *Sync source* and click *Restart.*

Disable the MMS Backup Service

Step 1: Stop and then terminate each sharded cluster enrolled in Backup. In MMS, click the *Backup* tab and select *Sharded Cluster Status*.

For each cluster enrolled in Backup, click *Stop*. If prompted, enter the two-factor authentication code, click *Verify*, and click *Stop* again.

When the *Terminate* button appears, click *Terminate*. Click *Terminate* again.

Step 2: Stop and then terminate each replica set enrolled in Backup. In MMS, click the *Backup* tab and select *Replica Set Status*.

For each replica set enrolled in Backup, click *Stop*. If prompted, enter the two-factor authentication code, click *Verify*, and click *Stop* again.

When the *Terminate* button appears, click *Terminate*. Click *Terminate* again.

Step 3: Stop all Backup Agents.

On Linux Issue the following command:

```
pkill -f mongodb-mms-backup-agent
```

On Windows In Windows `Control Panel`, open `Administrative Tools`, and then open `Services`.

In the list of services, select the `MMS Backup Agent` service. Select the `Action` menu and select `Stop`.

If you receive a message that your Backup Agent is out of date, see *Install or Update the Backup Agent on Windows*.

5.2 Restore MongoDB Instances with MMS Backup

Restore a Sharded Cluster from a Backup Restore a sharded cluster from a stored snapshot.

Restore a Replica Set from a Backup Restore a replica set from a stored snapshot or custom point-in-time snapshot.

Restore a Single Database Restore only a portion of a backup to a new `mongod` instance.

Restore from a Stored Snapshot Restore a replica set or sharded cluster from a stored snapshot.

Restore from a Point in the Last Day Restore a replica set from a custom snapshot from any point within a 24-hour period of time.

Seed a New Secondary with Backup Data Use MMS Backup to seed a new secondary in an existing replica set.

Configure Backup Data Delivery Select On Prem MMS Backup delivery method and file format.

Restore a Sharded Cluster from a Backup

Overview

You can restore a *sharded cluster* onto new hardware from the artifacts captured by On Prem MMS Backup.

You can restore from a snapshot or checkpoint. When you restore from a checkpoint, MMS takes the snapshot previous to the checkpoint and applies the oplog to create a custom snapshot. Checkpoint recovery takes longer than recovery from a stored snapshot.

MMS provides restore files as downloadable archive; MMS can also `scp` files directly to your system. The `scp` delivery method requires additional configuration, but provides faster delivery.

MMS provides a separate backup artifacts for each *shard* and one file for the *config servers*.

Sequence

The sequence to restore a snapshot is to:

- select and download the restore files,
- distribute the restore files to their new locations,
- start the `mongod` instances,
- configure each shard's replica set, and
- configure and start the cluster.

For a functional overview of MMS restore, see *Restores*.

Considerations

Procedures

Select and Download the Snapshot Files

Step 1: Click the *Backups* tab and then *Sharded Cluster Status*.

Step 2: Click the name of the sharded cluster to restore. MMS displays your selection's stored snapshots.

Step 3: Select the snapshot from which to restore. To select a **stored snapshot**, click the *Restore this snapshot* link next to the snapshot.

To select a **custom snapshot**, click the *Restore* button at the top of the page. In the resulting page, select a snapshot as the starting point. Then select the *Use Custom Point In Time* checkbox and enter the point in time in the *Date* and *Time* fields. MMS includes all operations up to but not including the point in time. For example, if you select 12:00, the last operation in the restore is 11:59:59 or earlier. Click *Next*.

Step 4: Select HTTP as the delivery method for the snapshot. In the *Delivery Method* field, select *Pull via Secure HTTP (HTTPS)*.

Step 5: Select `tar.gz` as the download format. In the *Format* drop-down list, select *Archive (tar.gz)*.

Step 6: Finalize the request. Click *Finalize Request* and confirm your identify via two-factor verification. Then click *Finalize Request* again.

Step 7: Retrieve the snapshot. MMS creates one-time links to tar files for the snapshot. The links are available for one download each, and each expires after an hour.

To download the tar files, select the MMS *Backup* tab and then *Restore Jobs*. When the restore job completes, the *download* link appears for every *config server* and *shard* in the cluster. Click each link to download the tar files and copy each tar file to its server. For a shard, copy the file to every member of the shard's *replica set*.

Restore Each Shard's Primary For *all* shards, restore the primary. You must have a copy of the snapshot on the server that provides the primary:

Step 1: Shut down the entire replica set. Connect to each member of the setand issue the following:

```
use admin
db.shutdownServer()
```

Step 2: Restore the snapshot data files to the primary. Extract the data files to the location where the `mongod` instance will access them. This is the location you will specify as the `dbpath` when running `mongod`.

For example, the following commands extract the data files and move them to a data directory:

```
tar -xvf <backup-restore-name>.tar.gz
mv <backup-restore-name> /data
```

Step 3: Restart the primary as a standalone, without the `--replSet` option. For example:

```
mongod --dbpath /data
```

Step 4: Run the `seedSecondary.sh` script on the primary. The seedSecondary.sh script re-creates the oplog collection and seeds it with correct timestamp. The file is included in the backup restore file, *except in certain circumstances*.

To run the script, issue the following command on at the system prompt:

```
./seedSecondary.sh <oplog-size-in-gigabytes>
```

Step 5: Restart the primary as part of a replica set. For example:

```
mongod --dbpath /data --replSet <replica-set-name>
```

Step 6: Connect to the primary and initiate the replica set. Use `rs.initiate()` to initiate the replica set. For example:

```
rs.initiate()
```

Restore All Secondaries After you have restored the primary for a shard you can restore all secondaries. You must have a copy of the snapshot on all servers that provide the secondaries:

Step 1: Connect to the server where you will create the new secondary.

Step 2: Restore the snapshot data files to the secondary. Extract the data files to the location where the `mongod` instance will access them. This is the location you will specify as the `dbpath` when running `mongod`.

For example, the following commands extract the data files and move them to a data directory:

```
tar -xvf <backup-restore-name>.tar.gz
mv <backup-restore-name> /data
```

Step 3: Restart the primary as a standalone, without the `--replSet` option. For example:

```
mongod --dbpath /data
```

Step 4: Run the `seedSecondary.sh` script on the secondary. The seedSecondary.sh script re-creates the oplog collection and seeds it with correct timestamp. The file is included in the backup restore file, *except in certain circumstances*.

To run the script, issue the following command on at the system prompt:

```
./seedSecondary.sh <oplog-size-in-gigabytes>
```

Step 5: Restart the secondary as part of the replica set. For example:

```
mongod --dbpath /data --replSet <replica-set-name>
```

Step 6: Connect to the primary and add the secondary to the replica set. Connect to the primary and use `rs.add()` to add the secondary to the replica set.

```
rs.add("<host>:<port>")
```

Repeat this operation for each member of the set.

Restore Each Config Server Perform this procedure separately for each *config server*. Each config server must have a copy of the tar file with the config server data.

Step 1: Restore the snapshot to the config server. Extract the data files to the location where the config server's `mongod` instance will access them. This is the location you will specify as the `dbPath` when running `mongod` for the config server.

```
tar -xvf <backup-restore-name>.tar.gz
mv <backup-restore-name> /data
```

Step 2: Start the config server. The following example starts the config server using the new data:

```
mongod --configsvr --dbpath /data
```

Step 3: Update the sharded cluster metadata. If the new shards do not have the same hostnames and ports as the original cluster, you must update the shard metadata. To do this, connect to each config server and update the data.

First connect to the config server with the `mongo` shell. For example:

```
mongo
```

Then access the `shards` collection in the config database. For example:

```
use config
db.shards.find().pretty()
```

The `find()` method returns the documents in the `shards` collection. The collection contains a document for each shard in the cluster. The `host` field for a shard displays the name of the shard's replica set and then the hostname and port of the shard. For example:

```
{ "_id" : "shard0000", "host" : "shard1/localhost:30000" }
```

To change a shard's hostname and port, use the MongoDB `update()` command to modify the documents in the `shards` collection.

Start the `mongos` Start the cluster's `mongos` bound to your new config servers.

Restore a Replica Set from a Backup

Overview

You can restore a replica set onto new hardware from the artifacts captured by On Prem MMS Backup.

You can restore from a stored snapshot or from a point in time in the last 24 hours. For point-in-time recovery, MMS applies the oplog to the snapshot that is previous to the selected point in order to create a custom snapshot. Point-in-time recovery takes longer than recovery from a stored snapshot.

MMS provides restore files as downloadable archive; MMS can also `scp` files directly to your system. The `scp` delivery method requires additional configuration, but provides faster delivery.

Sequence

The sequence to restore a replica set is to:

- download the restore file,
- distribute the restore file to to each server,
- start the `mongod` instances, and
- initiate and configure the replica set.

For more information on restores, see *Restores*. For additional approaches to restoring replica sets, see the procedure from the MongoDB Manual to Restore a Replica Set from a Backup.

Considerations

Procedures

Select and Download the Snapshot When you select a snapshot to restore, MMS creates a link to download the snapshot as a tar file. The link is available for one download only and times out after an hour. Once you download the tar file, copy it to each server to restore.

Step 1: Click the *Backups* tab and then *Replica Set Status*.

Step 2: Click the name of the replica set to restore. MMS displays your selection's stored snapshots.

Step 3: Select the snapshot from which to restore. To select a **stored snapshot**, click the *Restore this snapshot* link next to the snapshot.

To select a **custom snapshot**, click the *Restore* button at the top of the page. In the resulting page, select a snapshot as the starting point. Then select the *Use Custom Point In Time* checkbox and enter the point in time in the *Date* and *Time* fields. MMS includes all operations up to but not including the point in time. For example, if you select 12:00, the last operation in the restore is 11:59:59 or earlier. Click *Next*.

Step 4: Select HTTP as the delivery method for the snapshot. In the *Delivery Method* field, select *Pull via Secure HTTP (HTTPS)*.

Step 5: Finalize the request. Click *Finalize Request* and confirm your identify via two-factor verification. Then click *Finalize Request* again.

Step 6: Retrieve the snapshot. MMS creates a one-time link to a tar file of the snapshot. The link is available for one download and times out after an hour.

To download the snapshot, select the MMS *Backup* tab and then select *Restore Jobs*. When the restore job completes, select the *download* link next to the snapshot.

Restore the Primary You must have a copy of the snapshot on the server that provides the primary:

Step 1: Shut down the entire replica set. Connect to each member of the setand issue the following:

```
use admin
db.shutdownServer()
```

Step 2: Restore the snapshot data files to the primary. Extract the data files to the location where the `mongod` instance will access them. This is the location you will specify as the `dbpath` when running `mongod`.

For example, the following commands extract the data files and move them to a data directory:

```
tar -xvf <backup-restore-name>.tar.gz
mv <backup-restore-name> /data
```

Step 3: Restart the primary as a standalone, without the `--replSet` option. For example:

```
mongod --dbpath /data
```

Step 4: Run the `seedSecondary.sh` script on the primary. The seedSecondary.sh script re-creates the oplog collection and seeds it with correct timestamp. The file is included in the backup restore file, *except in certain circumstances*.

To run the script, issue the following command on at the system prompt:

```
./seedSecondary.sh <oplog-size-in-gigabytes>
```

Step 5: Restart the primary as part of a replica set. For example:

```
mongod --dbpath /data --replSet <replica-set-name>
```

Step 6: Connect to the primary and initiate the replica set. Use `rs.initiate()` to initiate the replica set. For example:

```
rs.initiate()
```

Restore Each Secondary After you have restored the primary you can restore all secondaries. You must have a copy of the snapshot on all servers that provide the secondaries:

Step 1: Connect to the server where you will create the new secondary.

Step 2: Restore the snapshot data files to the secondary. Extract the data files to the location where the `mongod` instance will access them. This is the location you will specify as the `dbpath` when running `mongod`.

For example, the following commands extract the data files and move them to a data directory:

```
tar -xvf <backup-restore-name>.tar.gz
mv <backup-restore-name> /data
```

Step 3: Restart the primary as a standalone, without the `--replSet` option. For example:

```
mongod --dbpath /data
```

Step 4: Run the `seedSecondary.sh` script on the secondary. The seedSecondary.sh script re-creates the oplog collection and seeds it with correct timestamp. The file is included in the backup restore file, *except in certain circumstances*.

To run the script, issue the following command on at the system prompt:

```
./seedSecondary.sh <oplog-size-in-gigabytes>
```

Step 5: Restart the secondary as part of the replica set. For example:

```
mongod --dbpath /data --replSet <replica-set-name>
```

Step 6: Connect to the primary and add the secondary to the replica set. Connect to the primary and use `rs.add()` to add the secondary to the replica set.

```
rs.add("<host>:<port>")
```

Repeat this operation for each member of the set.

Restore a Single Database

Overview

A backup snapshot contains a complete copy of the contents of your `mongod` `dbpath`. To restore a single collection or database or partial data, retrieve a backup snapshot and expand the snapshot data on a volume. Then use the `mongodump` and `mongorestore` commands to build and restore your data.

Procedure

Select and Download a Snapshot

Step 1: Click the *Backups* tab and then *Replica Set Status*.

Step 2: Click the name of the replica set that contains the database to restore. MMS displays your selection's stored snapshots.

Step 3: Select the snapshot from which to restore. To select a **stored snapshot**, click the *Restore this snapshot* link next to the snapshot.

To select a **custom snapshot**, click the *Restore* button at the top of the page. In the resulting page, select a snapshot as the starting point. Then select the *Use Custom Point In Time* checkbox and enter the point in time in the *Date* and *Time* fields. MMS includes all operations up to but not including the point in time. For example, if you select 12:00, the last operation in the restore is 11:59:59 or earlier. Click *Next*.

Step 4: Select HTTP as the delivery method for the snapshot. In the *Delivery Method* field, select *Pull via Secure HTTP (HTTPS)*.

Step 5: Finalize the request. Click *Finalize Request* and confirm your identify via two-factor verification. Then click *Finalize Request* again.

Step 6: Retrieve the snapshot. MMS creates a one-time link to a tar file of the snapshot. The link is available for one download and times out after an hour.

To download the snapshot, select the MMS *Backup* tab and then select *Restore Jobs*. When the restore job completes, select the *download* link next to the snapshot.

Restore the Database

Step 1: Use the mongodump command to dump a single database. Use the unpacked snapshot restore directory as the `dpath` switch and the single database name as the `--db` switch in the `mongodump` command:

```
mongodump --dbpath <path> --db <database-name>
```

Step 2: Use the mongorestore command to import the single database dump. Enter this `mongorestore` command:

```
mongorestore --db <database-name> --drop
```

You also may specify the `--drop` switch to drop all collections from the target database before you restore them from the `bson` file created with `mongodump`.

Restore from a Stored Snapshot

Overview

With On Prem MMS Backup, you can restore from a stored snapshot or build a *custom snapshot* reflecting a different point in the last 24 hours. For all backups, restoring from a stored snapshot is faster than restoring from a custom snapshot in the last 24 hours.

On Prem MMS Backup automatically takes and stores a snapshot every 6 hours. These snapshots are available for restores following the snapshot retention policy. For more information about the snapshot retention policy, as well as setting your own retention policy, see *the MMS Backup FAQ page*.

For replica sets, you will receive one `.tar.gz` file containing your data; for sharded clusters, you will receive a series of `.tar.gz` files.

Procedure

Step 1: Select the *Backups* tab, and then select either *Sharded Cluster Status* or *Replica Set Status*.

Step 2: Click the name of the sharded cluster or replica set to restore. MMS displays your selection's stored snapshots.

Step 3: Select the snapshot from which to restore. To select a **stored snapshot**, click the *Restore this snapshot* link next to the snapshot.

To select a **custom snapshot**, click the *Restore* button at the top of the page. In the resulting page, select a snapshot as the starting point. Then select the *Use Custom Point In Time* checkbox and enter the point in time in the *Date* and *Time* fields. MMS includes all operations up to but not including the point in time. For example, if you select 12:00, the last operation in the restore is 11:59:59 or earlier. Click *Next*.

Step 4: Select HTTP as the delivery method for the snapshot. In the *Delivery Method* field, select *Pull via Secure HTTP (HTTPS)*.

Step 5: Finalize the request. Click *Finalize Request* and confirm your identify via two-factor verification. Then click *Finalize Request* again.

Step 6: Retrieve the snapshot. To download the snapshot, select the MMS *Backup* tab and then select *Restore Jobs*. When the restore job completes, select the *download* link next to the snapshot.

For a sharded clusters, MMS provides several *download* links for the several `.tar.gz` files.

Step 7: Extract the data files from the .tar.gz archive created by the backup service.

```
tar -zxvf <tarball-name>.tar.gz
```

Step 8: Select the location where the mongod will access the data files. The directory you choose will become the mongod's data directory. You can either create a new directory or use the existing location of the extracted data files.

If you create a new directory, move the files to that directory.

If you use the existing location of the extracted data files, you can optionally create a symbolic link to the location using the following command, where `<hash>-<rsname>-<time>` is the name of the snapshot and `<dbpath>` is the data directory:

```
ln -s <hash>-<rsname>-<time>/ <dbpath>
```

Step 9: Start the mongod with the new data directory as the dbpath. In the mongod configuration, set the `dbpath` option to the path of the data directory that holds the data files from the On Prem MMS Backup snapshot.

```
mongod --dbpath /data/db
```

Additional Information

Restore from a Point in the Last 24 Hours

Restore from a Point in the Last 24 Hours

On Prem MMS Backup lets you restore data from a point within the last 24-hour period. MMS creates a backup that includes all operations up to the point in time you select. The point in time is an upper *exclusive* bound: if you select a timestamp of 12:00, then the last operation in the restore will be no later than 11:59:59.

To restore from a point in time, see the procedure for the resource you are restoring:

- *Restore a Sharded Cluster from a Backup*
- *Restore a Replica Set from a Backup*

Seed a New Secondary from Backup Restore

Overview

When a natural synchronization of a new secondary host costs too much time or resources, seeding a secondary from a backup restore is a faster better alternative. Seeding also does not hit a live mongo instance to retrieve data.

Prerequisites

To seed a secondary from a backup restore file, you must have:

- A backup restore file.
- The `seedSecondary.sh` file included in the backup restore file.

Considerations

The seedSecondary.sh file will not be in the backup restore if you have blacklisted dbs or collections or have resynced your backup after the snapshot (or for config servers). In these cases, including the script would cause an inconsistent secondary. In the case of a blacklist, your secondary would not include some collections which would cause problems for your deployment.

Seeding a new secondary from a backup restore requires an oplog window on the current primary that spans back to the snapshot's timestamp.

Procedure

Step 1: Remove the broken secondary from your replica set.

```
rs.remove("SECONDARYHOST:SECONDARYPORT")
```

Step 2: Login to the server on which to create the new secondary.

Step 3: Bring up new node as a standalone.

```
tar -xvf backup-restore-name.tar.gz
mv backup-restore-name data
mongod --port <alternate-port> --dbpath /data
```

Where `ALTERNATEPORT` is not the usual port your secondary runs on.

Step 4: Run seedSecondary.sh script to create oplog collection and seed with correct timestamp.

Step 5: Shut down the new secondary on the alternate port.

Step 6: Start up the new secondary.

```
mongod --port <secondary-port> --dbpath /data  --replSet REPLICASETNAME
```

Step 7: Add the new secondary to the replica set on the primary host.

```
rs.add("<secondary-host>:<secondary-port>")
```

Select Backup File Delivery Method and Format

Overview

With On Prem MMS Backup, you can restore from a *stored snapshot* or, if you are restoring a replica set, you may build a *custom snapshot* reflecting a different point in the last 24 hours. For all backups, restoring from a stored snapshot is faster than restoring from a custom snapshot in the last 24 hours.

Once you select a backup-enabled sharded cluster or replica set to restore, the next step is to select the delivery method and file format.

Procedures

Select Backup File via Secure HTTP (HTTPS) In the *Select Restore Destination* window, select *Pull via Secure HTTP (HTTPS)* to create a one-time direct download link.

Select Backup File via Secure Copy (SCP) In the *Select Restore Destination* window, select *Push via Secure Copy (SCP)*. You can grant access by supplying MMS with a username and password to your server, or you can provide a username and grant access via SSH public key.

To grant access via SSH public key:

Step 1: Select the *Settings* tab, and then select *Restore Settings*.

Step 2: Enter a passphrase and click the *Generate a New Public Key* button. On Prem MMS Backup generates and displays a public key.

Step 3: Log in to your server using the same username you will supply in your restore request.

Step 4: Add your public key to the authorized hosts file for that user. For security reasons, you should remove the public key from the authorized hosts file once you have obtained your backup file. The authorized hosts file is often located at `~/.ssh/authorized_keys`.

Note: For security reasons, you should remove this public key from the authorized hosts file once you have obtained your backup file.

Select Backup File Format In the *Select Restore Destination* window, select *Individual DB Files* or *Archive (tar.gz)* as the *Format*:

- Select *Individual DB Files* to transmit MongoDB data files produced by MMS Backup directly to the target directory. The individual database files are faster for On Prem MMS Backup to construct, but require additional file space on the destination server. The data *is* compressed during transmission.
- Select *Archive (tar.gz)* to deliver database files in a single `tar.gz` file you must extract before reconstructing databases.

Next Steps

Restore from a Stored Snapshot

Restore from a Point in the Last 24 Hours

5.3 Backup Use and Operation

Delete Backup Snapshots Manually remove unneeded stored snapshots from MMS.

SSL Configure Backup Agent to support SSL.

Delete Snapshots for Replica Sets and Sharded Clusters

Overview

To delete snapshots for replica sets and sharded clusters, use the MMS console to find then select a backup snapshot to delete.

Constraints

You can delete any replica set or sharded cluster snapshot if it is not needed for replica set point-in-time restores.

Procedure

Step 1: Click the MMS Backup tab.

Step 2: Select type of backup file to delete. Select either *Replica Set Status* or *Sharded Cluster Status*.

Step 3: Click the name of the replica set or sharded cluster. Displays replica set or sharded cluster details with a list of possible backup files to delete.

Step 4: Select backup file to delete. On the list of snapshots, click the *Delete* link to the right of a snapshot.

Step 5: Confirm deletion. Click the OK button on the *Delete Snapshot* interface to confirm deletion of the backup.

SSL

Using SSL with On Prem MMS Backup

On Prem MMS Backup can backup MongoDB instances running with SSL. To use SSL with `mongod` and `mongos`, you must enable it at compile time, or use MongoDB Enterprise.

The Backup Agent must be configured with the trusted CA certificates used to sign the certificates used by any MongoDB instances running with SSL. Edit the `local.config` file in your agent installation to set `sslTrustedServerCertificates` to the path of a file containing one or more certificates in PEM format.

By default, the Backup Agent only connects to MongoDB instances with a trusted certificate. For testing purposes, set the `sslRequireValidServerCertificates` setting in the `local.config` file to `False` to bypass this check. This configuration is NOT recommended for production use as it makes the connection insecure.

Activate SSL

Step 1: Go to the Backups page.

Step 2: Click the Replica Set Status tab.

Step 3: Click on the appropriate Start button next to your replica set. Configure SSL through this process or click the gear icon to the far right of any replica set name on the *Replica Set Status* tab.

Step 4: Click the Use SSL checkbox.

Step 5: Click the Save button. If you have questions about On Prem MMS Backup, consider *Frequently Asked Questions: Backup* or open a support request via the On Prem MMS Backup interface or open a support request.

6 Frequently Asked Questions

MMS Administration

Monitoring

Backup

6.1 Frequently Asked Questions: Management

See also:

Frequently Asked Questions: Monitoring and *Frequently Asked Questions: Backup*.

User and Group Management

How do I reset my password?

You can reset your password using the password reset form.

How do I change my password?

You can change your password by resetting your password.

What are the password requirements?

Passwords must be at least 8 characters long and contain at least one letter, one digit, and one special character.

Passwords for the MongoDB Jira instance and MMS are the same, although the length and character requirements are different for Jira and MMS.

How do I add a user to my company/group?

If the user already has a MongoDB Jira or MMS account, you can add their username to your group on the admin page.

If the user does not have a Jira account then they can create a new account. After they have created an account, you can add their username to the company/group on the admin page.

How do I remove my company/group?

Please contact your MMS administrator to remove a company or group from your MMS account.

How can I configure multiple Google Authenticator apps to use the same account?

By selecting the *Can't scan the barcode?* option during the procedure to *Configure Two-Factor Authentication with Google Authenticator*. The option provides a common key that multiple Google Authenticator apps can use.

Activity

My alert email says my host(s) are exposed to the public Internet. What does that mean?

This alert indicates only that the MMS server can connect to the `mongod` that it is monitoring. It does not diagnose whether your host is exposed to the public, despite the alert message. This alert occurs if you configured a setting called *Exposed DB Host Check*, which is a setting used with the Cloud version of MMS.

See *Manage Alerts* to disable or modify the exposed host alerts.

How do I modify my alert settings?

You can enable, disable, or modify alerts on the settings tab of the *Activity* page.

How frequently can alerts be set?

MMS processes alerts on a 5-minute interval. Therefore, the minimum frequency for an alert is 5 minutes. The default frequency for new alert configurations is 60 minutes.

Operations

Do Web or Database Hosting Companies Integrate with MMS ?

Web hosting companies can offer the ability to use MMS with their hosted MongoDB databases, for example, to set up software agents to monitor and backup databases hosted on their servers. MongoDB has confirmed compatability with MongoHQ, MongoLab, and Heroku. Implementation details depend on each hosting company.

MongoHQ offers MMS upon request as part of their Database as a Service (DaaS) business.

MongoLab offers MMS as part of their Database as a Service (DaaS) business. MongoLab offers the service on their dedicated plans and shared replica set plan. They also provide instructions to tune MongoDB performance with MMS on their servers.

MongoHQ and MongoLab are MongoDB Advanced Partners.

Heroku offers web hosting with a MongoHQ add-on and MongoLab add-on to use MongoDB databases from these database hosting companies. Heroku also offers MMS monitoring of those databases with detailed setup instructions.

About On-Prem MongoDB Management Service

What open source projects does MMS use?

- Database: MongoDB
- App framework: Google Guice
- Http server: Jetty
- Web framework: Jersey
- Misc server libs: Apache Commons
- UI lib: jQuery , Bootstrap
- Charts: dygraphs
- Graphics: Font-Awesome

6.2 Frequently Asked Questions: Monitoring

See also:

Frequently Asked Questions: Management

Host Configuration

How do I add a new host or server?

Click on the plus icon on the top of the hosts page.

Can I monitor Kerberos-enabled nodes?

Yes. On Prem MMS Monitoring does support monitoring for Kerberos-enabled MongoDB instances. See *Connect to Hosts with Kerberos Authentication* for more information.

How does MMS gather database statistics?

In most instances, On Prem MMS Monitoring will scale its request cycle to limit more expensive statistics gathering. The information on the DB Stats tab updates every 10 minutes, and the agent will throttle the frequency to reduce the impact on the database. [1] Even so, the "DB stats" operation impacts the performance of your database, as is possible when installations have a large number of databases and collections.

If the collection of database statistics impacts database performance, disable database stats collection. See the "DB Stats" section on the "Settings" page in MMS before starting your agent.

On Prem MMS Monitoring Agent

Do I need a Monitoring Agent for every MongoDB?

No. A single Monitoring Agent can be used to connect to all MongoDB databases in your deployment. We strongly recommend you complete your initial Monitoring Agent setup with a single Monitoring Agent.

For redundancy, you may wish to run a second Monitoring Agent. See the *Monitoring Agent Redundancy* for more information.

Can I use two Monitoring Agents to connect MongoDBs in different data centers?

No. The Monitoring Agent must connect to every server in your MongoDB deployment. Configure firewalls to allow the Monitoring Agent to connect across data centers and servers.

Use multiple Monitoring Agents *only* to provide redundancy. Each agent must be able to connect to every monitored MongoDB. We strongly recommend you complete your initial Monitoring Agent setup with a single agent.

What happens if a Monitoring Agent becomes unavailable? How can I ensure my MongoDBs are always monitored?

You can run multiple Monitoring Agents. If one Monitoring Agent fails, another starts monitoring. As long as at least one Monitoring Agent is available, MMS will not trigger an *Monitoring Agent Down* alert. To run multiple Monitoring Agents, see *Monitoring Agent Redundancy*.

You also can create an alert to notify you when an agent is down. In MMS, click *Activity* then *Agent Settings*. Click the *Add Alert* button then set the alert through the fields in the *Create a New Alert* window.

[1] The DB Stats tab will not appear until 30 minutes after you add the host to On Prem MMS Monitoring

Where should I run the Monitoring Agent?

The amount of resources the Monitoring Agent requires varies depending on infrastructure size, the number of nodes and the databases it's monitoring. Run the agent on an existing machine with additional capacity that *does not* run a **mongod** instance. You may also run the Monitoring Agent on a smaller dedicated instance.

The Monitoring Agent load scales with the number of monitored `mongod` plus `mongos` processes and the number of databases in your MongoDB environment.

Never install the Monitoring Agent on the same server as a data bearing **mongod** instance. This will allow you to perform maintenance on a the **mongod** and its host without affecting the monitoring for your deployment. Additionally a Monitoring Agent may contend for resources with the **mongod**

You can install the Monitoring Agent on the same system as an *arbiter*, a **mongos**, or an application server depending on the requirements of these services and available resources.

Can I run the Monitoring Agent on an AWS micro instances?

If you monitor five or fewer `mongod` instances, you can use a AWS micro instance.

Why can't the Monitoring Agent connect to my host?

The most common problem is that the agent is unable to resolve the hostname of the server. Check DNS and the `/etc/hosts` file.

The second most common problem is that there are firewall rules in place that prohibit access to the server from the agent.

To test the connection, login to the server running the agent and run: `mongo hostname:port/test` If you are unable to connect, the agent will not be able to connect.

In addition, On Prem MMS Monitoring supports monitoring for Kerberos-enabled nodes.

Why does the Monitoring Agent connect with hostnames instead of IP addresses?

By default, the Monitoring Agent tries to connect by resolving hostnames. If the agent cannot connect by resolving a hostname, you can force the Monitoring Agent to prefer an IP address over its corresponding hostname for a specific IP address.

To create a preferred hostname, on the *Settings* page, click the *Group Settings* tab then click the *Add* button to the right of the Preferred Hostnames heading. If your IP addresses have a common prefix, create a preferred hostname with the *ends-with* button or click the *regexp* button to use a regular expression.

Preferred hostnames also allow you to specify the hostname to use for servers with multiple aliases. This prevents servers from appearing multiple times under different names in the MMS interface.

How do I download the Monitoring Agent?

You can download the Monitoring Agent from the "Monitoring Agent" section on the MMS settings page.

How do I setup and configure the agent?

See the `README` file included in the agent download.

How do I delete a Monitoring Agent from MMS?

Monitoring Agents report their status to the On-Prem MongoDB Management Service. When an agent does not report for more than 24 hours, the agent no longer appears in MMS.

For more details, see *Delete Monitoring Agents*.

Can I run the MMS Monitoring Agent with MMS Backup On Prem?

Yes. Both the MMS Monitoring Service and MMS Backup On Prem can operate in the same environment. You will need to install and configure two separate Monitoring Agents: configure one agent for the On Prem environment and the other for the MMS Service.

Data Presentation

What are all those vertical bars in my charts?

A *red bar* indicates a server restart.

A *purple bar* indicates the server is now a primary.

A *yellow bar* indicates the server is now a secondary.

How do I magnify dashboard chart data?

Click and drag on a dashboard chart, horizontally or vertically, to zoom and isolate a specific data region. Other charts will automatically zoom to the same region. Double click on a chart to reset zoom level.

Why is my Monitoring Agent highlighted in red on the Agents tab?

Your agent is out of date.

You can update the Monitoring Agent from the "Monitoring Agent" section on the MMS settings page.

Data Retention

What is the data retention policy for On-Prem MongoDB Management Service?

Data retention policies, as defined in the Terms of Service are always subject to change. Currently, On-Prem MongoDB Management Service preserves 3 days of minute-level data is retained, 94 days of hour data, and unlimited day-level data.

6.3 Frequently Asked Questions: Backup

MMS Backup creates backups of MongoDB replica sets and sharded clusters. After an initial sync to MongoDB's datacenters, MMS Backup tails the operation log (oplog) to provide a continuous backup with point-in-time recovery of replica sets and consistent snapshots of sharded clusters. For more information, please review these frequently asked questions or create an MMS Backup account.

Requirements

What version of MongoDB does On Prem MMS Backup require?

To back up a sharded cluster, On Prem MMS Backup requires version 2.4.3 or later.

To back up a replica set, On Prem MMS Backup requires version 2.2 or later.

What MongoDB permissions does the Backup Agent require?

If you are backing up a MongoDB instance that has authentication enabled, the Backup Agent requires elevated privileges, as described in *MMS Backup*.

See also:

User Privilege Roles in MongoDB.

Are there any limits to the types of deployments On Prem MMS Backup supports?

Yes. On Prem MMS Backup does not currently support standalone deployments. On Prem MMS Backup has full support for replica sets and sharded clusters.

Why doesn't On Prem MMS Backup support standalone deployments?

After an initial sync of your data to MMS, On Prem MMS Backup copies data from the *oplog* to provide a continuous backup with point-in-time recovery. On Prem MMS Backup does not support standalone servers, which do not have an oplog. To support backup with a single `mongod` instance, you can run a one-node replica set.

See also:

Convert a Standalone to a Replica Set.

How Does MMS Measure Data Size?

MMS uses the following conversions to measure snapshot size and to measure how much oplog data has been processed:

- 1 MB = 1024^2 bytes
- 1 GB = 1024^3 bytes
- 1 TB = 1024^4 bytes

Interface

How can I verify that I'm running the latest version of the Backup Agent?

If your Backup Agent is out of date, it will be highlighted in red on the *Agents* tab of the "Hosts" page in the MMS interface.

Why is my Backup Agent highlighted in red?

If your agent is highlighted in red on the *Agents* tab of the "Hosts" page, your agent is out of date. For instructions on updating the agent, see the *installation instructions*.

Operations

How does On Prem MMS Backup work?

You install the Backup Agent on a server in the same deployment with your MongoDB infrastructure. The agent conducts an initial sync of your data to MMS. After the initial sync, the agent tails the oplog to provide a continuous backup of your deployment.

Where should I run the Backup Agent?

The Backup Agent can run anywhere in your infrastructure that has access to your `mongod` instances. To avoid contention for network and CPU resources, *do not* run the Backup Agent on the same hosts that provide your `mongod` instances.

The Backup Agent has the same performance profile impact as a secondary. For the initial backup, the load scales with the size of your data set. Once an initial backup exists, the load scales with oplog gigabytes used per hour.

Can I run the Backup and Monitoring Agents on a Single System?

There is no technical restriction that prevents the Backup Agent and the Monitoring Agent from running on a single system or host. However, both agents have resource requirements, and running both on a single system can affect the ability of these agents support your deployment in MMS.

The resources required by the Backup Agent depend on rate and size of new oplog entries (i.e. total oplog gigabyte/per-hour produced.) The resources required by the Monitoring Agent depend on the number of monitored `mongod` instances and the total number of *databases* provided by the `mongod` instances.

Can I run multiple Backup Agents to achieve high availability?

You can run multiple Backup Agents for high availability. If you do, the Backup Agents must run on different hosts.

When you run multiple Backup Agents, only one agent per group or environment is the **primary agent**. The primary agent performs the backups. The remaining agents are completely idle, except to log their status as standbys and to periodically ask MMS whether they should become the primary.

Will the MongoDB Backup Service impact my production databases?

On Prem MMS Backup will typically have minimal impact on production MongoDB deployments. This impact will be similar to that of adding a new *secondary* to a *replica set*.

By default, the Backup Agent will perform its initial sync, the most resource intensive operation for On Prem MMS Backup, against a secondary member of the replica set to limit its impact. You may optionally configure the Backup Agent to perform the initial sync against the replica set's *primary*, although this will increase the impact of the initial sync operation.

Does the Backup Agent modify my database?

The Backup Agent writes a small token into the oplog of the source database every hour. These tokens provide a heartbeat for On Prem MMS Backup and have no effect on the source deployment. Each token is less than 100 bytes.

What is the load on the database during the initial Backup sync?

The impact of the initial backup synchronization should be similar to syncing a new secondary replica set member. The Backup Agent does not throttle its activity, and attempts to perform the sync as quickly as possible.

Can I backup my standalone deployment?

No. On Prem MMS Backup does not currently support standalone deployments. To convert to a replica set, consult MongoDB's replication documentation.

How do I perform maintenance on a Replica Set with Backup enabled?

Most operations in a replica set are replicated via the oplog and are thus captured by the backup process. Some operations, however, make changes that are *not* replicated: for these operations you *must* have the Backup service resync from your set to include the changes.

The following operations are not replicated and therefore require resync:

- Renaming or deleting a database by deleting the data files in the data directory. As an alternative, remove databases using an operation that MongoDB will replicate, such as `db.dropDatabase()` from the `mongo` shell.
- Changing any data while the instance is running as a *standalone*.
- Using `compact` or `repairDatabase` to reclaim a significant amount of space. This is not strictly necessary but will ensure that the MMS copy of the data is resized, which means quicker restores and lower costs.

See also:

Maintenance Operations for Replica Set Members.

Does the Backup Agent Support SSL?

The Backup Agent can connect to replica sets and shared clusters configured with SSL. See the *Backup Agent Configuration* documentation for more information.

How Do I Delete a Backup Snapshot?

You can delete replica set backup snapshots and snapshots for replica sets in a sharded cluster set if the snapshots are not needed for point-in-time restores. See *Delete Snapshots for Replica Sets and Sharded Clusters* for details.

Configuration

What are "excluded namespaces"?

Excluded namespaces are databases and collections that MMS will not back up. This is useful for large databases or collections that contain data that you will not need to restore: caches and logs, for example.

How can I prevent On Prem MMS Backup from backing up a collection?

On Prem MMS Backup allows you to specify "excluded namespaces", which are collections or databases that you do not want MMS to back up.

You can specify the namespaces to exclude when you initially enable backup on a replica set or sharded cluster, or can edit the list at any time by selecting the "*gear icon*" in the *Sharded Cluster Status* or *Replcia Set Status* tables in MMS.

How can I change which namespaces are on the "excluded namespaces" list?

Click on the "*gear icon*" next to the name of the replica set or sharded cluster whose excluded namespaces you want to modify in the *Sharded Cluster Status* or *Replcia Set Status* tables in MMS. A modal window will open, where you can add databases or collections to the list, or remove list items by clicking on the red *x* icon.

Removing a namespace from the excluded namespaces list necessitates a re-sync. On Prem MMS Backup handles this re-sync.

How can I use backup if Backup Jobs Fail to Bind?

The most common reason that jobs fail to bind to a Backup daemon is when no daemon has space for local copy of the backed up replica set.

To increase capacity so that the backup job can bind, you can:

- add an additional backup daemon.
- increase the size of the file system that holds the `rootDirectory` directory.
- move the `rootDirectory` data to a new volume with more space, and create a symlink or configure the file system mount points so that the daemon can access the data using the original path.

How do I resolve `applyOps` Errors During Backups?

If you notice *consistent* errors in `applyOps` commands in your Backup logs, it *may* indicate that the daemon has run out of space.

To increase space on a daemon to support continued operations, you can:

- increase the size of the file system that holds the `rootDirectory` directory.
- move the `rootDirectory` data to a new volume with more space, and create a symlink or configure the file system mount points so that the daemon can access the data using the original path.

Restoration

MMS Backup produces a copy of your data files that you can use to seed a new deployment. For an overview of restore, see *Restores*.

How can MMS provide point-in-time restores for any point in time?

Although it is faster to provide a restore for the time at which a snapshot was actually stored, this might not be ideal when restoring a replica set or sharded cluster. In consequence, the Backup service can build a restore to any point in time within a 24-hour period by replaying the oplog to the desired time.

For details, see the *procedures for restoring replica sets and sharded clusters*.

Can I take snapshots more or less often than every 6 hours?

No, MMS does not support a snapshot schedule with more frequent snapshots. See *Activate MMS Backup for a Replica Set* and *Activate MMS Backup for a Sharded Cluster* for more information on configuring backup snapshot schedules.

Can I set my own snapshot retention policy?

Yes. Backup snapshot retention is configurable.

The default snapshot retention policy is to maintain:

- 6-hour interval snapshots for 2 days,
- Daily snapshots stored for 1 week,
- Weekly snapshots stored for 1 month, and
- Monthly snapshots stored for 1 year.

You can customize both the frequency and schedule of snapshots that MMS captures. This allows you to tune your backup strategy based on your requirements.

For example you may choose to capture more frequent snapshots for the most mission critical data, and capture snapshots less frequently for less critical data.

See *Activate MMS Backup for a Replica Set* and *Activate MMS Backup for a Sharded Cluster* for details about configuring snapshot retention frequency and excluding namespaces for non-critical databases and collections.

How long does it take to create a restore?

On-Prem MongoDB Management Service transmits all backups in a compressed form from the On-Prem MongoDB Management Service server to your infrastructure.

In addition, point-in-time restores that require creating a new snapshot take additional time, which depends on the size of the scheduled snapshot and the amount the oplog entries that On Prem MMS Backup must apply to the preceding snapshot to roll forward to the requested point-in-time of the backup.

Does On Prem MMS Backup perform any data validation?

On Prem MMS Backup conducts basic corruption checks and provides an alert if any component (e.g., the agent) is down or broken, but does not perform explicit data validation. When it detects corruption, On Prem MMS Backup errs on the side of caution and invalidates the current backup and sends an alert.

How do I restore? What do I get when I restore?

You can request a restore via MMS, where you can then choose which snapshot to restore and how you want On Prem MMS Backup to deliver the restore. All restores require 2-factor authentication. MMS will send an authorization code via SMS code to your administrator. You must enter the authorization code into the backup interface to begin the restore process.

Note: From India, use Google Authenticator for two-factor authentication. Google Authenticator is more reliable than authentication with SMS text messages to Indian mobile phone numbers (i.e. country code 91).

On Prem MMS Backup delivers restores as `tar.gz` archives of MongoDB data files.

Restore delivery options are:

- **SCP to your Infrastructure**: On Prem MMS Backup will transmit the backup to your infrastructure over a secure channel. You must provide connection information for a host in your deployment.
- **Download**: On Prem MMS Backup will make your restore data available using a custom, one-time-use URL.

What is the SCP public key for On-Prem MongoDB Management Service?

On-Prem MongoDB Management Service generates an SSH public key on a per user basis to use when for delivering backups via SCP. To generate a public key, go to the "Settings" page and choose "Backup and Restore Public Key," then type in a passphrase and click on "Generate a New Public Key".

The public key will generate an SSH key and display it. Add this key to your authorized hosts file.

See the *Restore via Secure Copy* documentation for more information about granting access via SSH public key.

How does Backup handle Rollbacks?

If your MongoDB deployment experiences a rollback, then MMS Backup also rolls back.

Backup detects the rollback when a *tailing cursor* finds a mismatch in timestamps or hashes of write operations. Backup enters a rollback state and tests three points in the oplog of your replica set's *primary* to locate a common point in history. MMS rollback differs from MongoDB *secondary* rollback in that the common point does not necessarily have to be the most *recent* common point.

When Backup finds a common point, the service invalidates oplog entries and snapshots beyond that point and rolls back to the most recent snapshot before the common point. Backup then resumes normal operations.

If MMS cannot find a common point, a resync is required.

7 Reference

Configuration An overview of the hardware and software requirements for On Prem MMS.

User Roles Describes the user roles available within the MMS.

Alert Conditions Identifies all available alert triggers and conditions.

MMS Agent Authentication Requirements Details the permissions required to use MMS with MongoDB instances that enforce access control.

On Prem MMS Reference Reference for the On Prem Application, including all ports used by MMS components.

Monitoring Reference A reference sheet for the monitoring service.

Supported Browsers A list of browsers supported by the MMS console.

MMS Public API Complete documentation of the HTTP API for MMS.

Monitoring Agent Configuration Documentation of the settings available in the Monitoring Agent configuration file.

Backup Agent Configuration Documentation of the settings available in the Backup Agent configuration file.

7.1 Configuration

Overview

The MMS Application Package and the MMS Backup Daemon Package include a `conf-mms.properties` file. Located in the `<install_dir>/conf/` directory, the `conf-mms.properties` files contain configuration settings for their respective services.

To start either service, you must configure the *Application URL Settings* and *Email Address Settings* in the respective `conf-mms.properties` file.

Since the configuration file may contain user credentials in plain text, follow standard practice and reduce the permissions on the configuration file:

```
sudo chmod 600 <install_dir>/conf/conf-mms.properties
```

Settings

Application URL Settings

The following two settings are mandatory.

`mms.`**`centralUrl`**
Type: string

Required. Fully qualified URL, including the port number, of the MMS Monitoring server. For example,

```
mms.centralUrl=http://mms.example.com:8080
```

`mms.`**`backupCentralUrl`**
Type: string

Required. The hostname and port of MMS Backup server. For example,

```
mms.backupCentralUrl=http://mms.example.com:8081
```

You must set `mms.backupCentralUrl`, even if you are only using MMS Monitoring and not MMS Backup.

Email Settings

Email Address Settings The following email address settings are mandatory. You **must** define them before the On Prem MMS Monitoring instance will start.

`mms.`**`fromEmailAddr`**
Type: string

Required. The email address used for sending the general emails, such as MMS alerts. You can include an alias with the email address. For example:

```
mms.fromEmailAddr=MMS Alerts <mms-alerts@example.com>
```

mms.**replyToEmailAddr**
Type: string

Required. The email address to send replies to general emails. For example:

```
mms.replyToEmailAddr=mms-no-reply@example.com
```

mms.**adminFromEmailAddr**
Type: string

Required. The email address to send messages from the MMS admin. You can include an alias with the email address. For example:

```
mms.adminFromEmailAddr=MMS Admin <mms-admin@example.com>
```

mms.**adminEmailAddr**
Type: string

Required. The email address to send messages or replies to the MMS admin. You can include an alias with the email address. For example:

```
mms.adminEmailAddr=mms-admin@example.com
```

mms.**bounceEmailAddr**
Type: string

Required. The email address to send bounce messages, i.e. messages of non-delivery of alerts or messages from MMS admin. For example:

```
mms.bounceEmailAddr=bounce@example.com
```

Email Service Settings

mms.**emailDaoClass**
Type: string

The email interface to use. For AWS Simple Email Service, specify `com.xgen.svc.core.dao.email.AwsEmailDao`, as in:

```
mms.emailDaoClass=com.xgen.svc.core.dao.email.AwsEmailDao
```

For AWS Simple Email Service, see also `aws.accesskey` and `aws.secretkey`.

For JavaEmailDao, specify `com.xgen.svc.core.dao.email.JavaEmailDao`, as in:

```
mms.emailDaoClass=com.xgen.svc.core.dao.email.JavaEmailDao
```

mms.mail.**transport**
Type: string *Default*: smtp

Transfer protocol `smtp` or `smtps` as specified by your email provider. For example:

```
mms.mail.transport=smtp
```

mms.mail.**hostname**
Type: string *Default*: localhost

Email hostname as specified by your email provider. For example:

```
mms.mail.hostname=mail.example.com
```

mms.mail.**port**
Type: number *Default*: 25

Port number for the transfer protocol as specified by your email provider. For example:

```
mms.mail.port=25
```

mms.mail.**tls**
Type: boolean *Default*: false

Indicator of whether the transfer protocol runs on top of TLS. For example:

```
mms.mail.tls=false
```

mms.mail.**username**
Type: string

User name of the email account. If unset, defaults to disabled SMTP authentication.

```
mms.mail.username=
```

mms.mail.**password**
Type: string

Password for the email account. If unset, defaults to disabled SMTP authentication.

```
mms.mail.password=emailPassword
```

aws.**accesskey**
Required if using AWS Simple Email Service. The access key ID for AWS.

```
aws.accesskey=EXAMPLEAccessKeyID
```

aws.**secretkey**
Required if using AWS Simple Email Service. The secret access key for AWS.

```
aws.secretkey=eXampLe/aCcESs/KEY
```

Twilio SMS Alert Settings

To receive alert notifications via SMS, you must have a Twilio account and specify your Twilio account information in the configuration file.

twilio.account.**sid**
Type: string

Twilio account ID.

twilio.auth.**token**
Type: string

Twilio API token.

twilio.from.**num**
Type: string

Twilio phone number.

MongoDB Settings

mongo.**mongoUri**
Type: string

Required. The connection string to the MongoDB server for MMS, i.e. the MMS Application Database. For example, the following specifies the URI for a *replica set*:

```
mongo.mongoUri=mongodb://db1.example.net:40000,db2.example.net:40000,db3.example.net:40000
```

For a MongoDB server with access control, prefix the hostname with the MongoDB username and password in the form `<username>:<password>@`, and append after the port the `http://mms.mongodb.com/helpadmin` database. For example:

```
mongo.mongoUri=mongodb://mongodbuser1:password@mydb1.example.net:40000/admin
```

For additional considerations when specifying user credentials, such as encrypting user credentials, see *MongoDB Access Control Considerations*.

See Connection String URI Format for more information on the connection string.

mongo.**replicaSet**
Type: string

Required if using a replica set for `mongo.mongoUri`. The name of the replica set. For example:

```
mongo.replicaSet=mmsreplset
```

mongo.backupdb.**mongoUri**
Type: string

Required for MMS Backup. The connection string to the MMS Backup Blockstore Database. This must be a separate MongoDB Server than the MMS Application Database. For example:

```
mongo.backupdb.mongoUri=mongodb://db5.example.net:50000,db6.example.net:50000,db7.example.net:50
```

mongo.backupdb.**replicaSet**
Type: string

Required for MMS Backup if using a replica set for `mongo.backupdb.mongoUri`. The name of the replica set. For example:

```
mongo.backupdb.replicaSet=mmsbackupreplset
```

mongo.**encryptedCredentials**
Type: boolean

Optional. Set to `true` if `mongo.mongoUri` contains the encrypted username and password.

```
mongo.encryptedCredentials=true
```

The username and password must have been encrypted using the On Prem MMS Monitoring `credentialstool`. See *MongoDB Access Control Considerations* for more information on encrypting username and password.

Important: The `conf-mms.properties` file can contain multiple `mongo.MongoURI` settings. If `mongo.encryptedCredentials` is `true`, you must encrypt all user credentials found in the various `mongo.MongoURI` settings.

MMS Backup Daemon Settings

These settings in the `conf-daemon.properties` file, are necessary only if you are using MMS Backup.

`rootDirectory`
Type: string

The disk partition used by the Backup Daemon to dynamically create and maintain the *replica set* HEAD directories. For more information on HEADs, see the *MMS Backup functional overview*.

This directory must be writable by the mongodb-mms user and must end in a trailing slash. It is critical that this partition is sized appropriately.

Important: Data in this directory is dynamically created, maintained and destroyed by the MMS Backup Daemon. This partition should not be used for any other purpose. This partition should *not* overlap with the partition used for the Backup Blockstore Database.

`mongodb.release.`**`directory`**
Type: string

Specifies the full path to the directory that contains every MongoDB release needed by the Backup Daemon. When backing up a replica set, The Backup Daemon must use a `mongod` that matches the version of the replica set being backed up.

If you update versions manually, name the folders within this full directory path using the following form:

```
mongodb-<platform>-<architecture>-<version>
```

For example:

```
mongodb-linux-x86_64-2.4.8
mongodb-linux-x86_64-2.4.9
mongodb-linux-x86_64-2.4.10
mongodb-linux-x86_64-2.6.0
```

The Backup Daemon includes the `mongodb-fetch` utility that will download the latest releases directly from mongodb.org/downloads. The `mongodb.release.autoDownload` setting automatically runs this utility every hour once the service starts. For details, including the option to download manually, see `mongodb.release.autoDownload`.

`mongodb.release.`**`autoDownload`**
Type: boolean

Specify `true` to enable automatic downloads; `false` to disable.

If this setting is enabled, Backup automatically downloads the latest release of MongoDB from mongodb..org/downloads and stores it in the directory specified by the `mongodb.release.directory` setting. The Backup Daemon includes the `mongodb-fetch` utility, located in the `http://mms.mongodb.com/helpopt/mongodb/backup-daemon/bin` directory, which runs once an hour to perform the downloads.

If you set `mongodb.release.autoDownload` `false`, then you must manually download and install the needed MongoDB releases in the `mongodb.release.directory`. If you backup deployments that use different MongoDB versions, you must download and install each version.

Download MongoDB from mongodb..org/downloads and extract them. Alternately, you can use the `mongodb-fetch` utility manually, included in the distribution the backup component ensures that the Backup Daemon has the correct version of `mongod` for every backed up replica set.

Session Management Setting

`mms.session.`**`maxHours`**
Type: number

The number of hours before a session on the MMS website expires.

Password Policy Settings

You can configure the password policy for MMS user accounts with the following settings:

`mms.password.`**`minChangesBeforeReuse`**
Type: number

The number of previous passwords to remember. You cannot reuse a remembered password as a new password.

`mms.password.`**`maxFailedAttemptsBeforeAccountLock`**
Type: number

The number of failed login attempts before an account becomes locked. Only an an MMS Administrator can unlock a locked account.

`mms.password.`**`maxDaysInactiveBeforeAccountLock`**
Type: number

The maximum number of days with no visits to the MMS website before an account should be locked.

`mms.password.`**`maxDaysBeforeChangeRequired`**
Type: number

The number of days a password is valid before the password expires.

SNMP Heartbeat Settings

You can configure the On Prem MMS Server to send a periodic heartbeat trap notification (v2c) that contain an internal health assessment of the MMS Server. The MMS Server can send traps to one or more endpoints on the standard SNMP UDP port 162.

To configure the On Prem MMS Server to send trap notifications, download the Management Information Base (MIB) file at http://downloads.mongodb.com/on-prem-monitoring/MMS-MONGODB-MIB.txt and configure the following settings:

`snmp.default.`**`hosts`**
Type: string *Default*: blank

Comma-separated list of hosts where 'heartbeat' traps will be sent on the standard UDP port 162. You must set `snmp.default.hosts` to enable the SNMP heartbeat functionality; otherwise, leaving the setting blank disables the SNMP heartbeat functionality.

`snmp.listen.`**`port`**
Type: number *Default*: 11611

Listening UDP port for SNMP. Setting to a number less than 1024 will require running MMS server with root privileges.

`snmp.default.heartbeat.`**`interval`**
Type: number *Default*: 300

Number of seconds between heartbeat notifications.

reCaptcha Settings

To enable reCaptcha anti-spam test on new user registration, you must have a reCaptcha account and specify the API information in the configuration file.

`reCaptcha.public.key`
Type: string

The reCaptcha public key associated with your account.

`reCaptcha.private.key`
Type: string

The reCaptcha private key associated with your account.

LDAP Settings

LDAP Server Setting
`mms.userSvcClass`
Type: string

The LDAP service class `com.xgen.svc.mms.svc.user.UserSvcLdap`; i.e.

```
mms.userSvcClass=com.xgen.svc.mms.svc.user.UserSvcLdap
```

LDAP User Settings Specify the LDAP directory schema properties in the following settings:

`mms.ldap.url`
Type: string

The URI for the LDAP server. For example:

```
mms.ldap.url=ldap://174.129.71.167:3890
```

`mms.ldap.bindDn`
Type: string

The LDAP user used to execute searches for other users. For example:

```
mms.ldap.url=_search_
```

`mms.ldap.bindPassword`
Type: string

The credentials for the search user. For example:

```
mms.ldap.bindPassword=dISDFFFnj7WMmc
```

`mms.ldap.user.baseDn`
Type: string

The base Directory Name (DN) used for searching for users. Escape the = sign with \. For example:

```
mms.ldap.user.baseDn=c\=users,d\=identity
```

`mms.ldap.user.searchAttribute`
Type: string

The LDAP user record attribute that MMS uses to search and authenticate users when a user types their username into the MMS login form. For example:

```
mms.ldap.user.searchAttribute=uid
```

mms.ldap.user.**firstName**
Type: string

The LDAP user attribute that contains the user's first name. For example:

```
mms.ldap.user.firstName=givenName
```

mms.ldap.user.**lastName**
Type: string

The LDAP user attribute that contains the user's last name. For example:

```
mms.ldap.user.lastName=sn
```

mms.ldap.user.**email**
Type: string

The LDAP user attribute that contains the user's email address. For example:

```
mms.ldap.user.email=mail
```

mms.ldap.user.**group**
Type: string

The LDAP user attribute that contains the list of groups that the user belongs to.

```
mms.ldap.user.group=groups
```

These can be either Common Names (CN's) or Distinguished Names (DN's) as long as they are consistent with those provided in the MMS *LDAP Global Role Settings*. For example:

LDAP Global Role Settings

Global parameters can be in any format for an LDAP group. They can be a Common Name (i.e. `cn`) or a Distinguished Name (i.e. `dn`). The format must match the property specified by the `mms.ldap.user.group` setting.

mms.ldap.global.role.**read_only**
Type: string

The LDAP group attribute name for users assigned the global read-only role in MMS. This role can only view data in MMS. For example:

```
mms.ldap.global.role.read_only=AcmeDbas
```

mms.ldap.global.role.**monitoring_admin**
Type: string

The LDAP group attribute name for users assigned the global monitoring administrative role in MMS. This role can view hosts, charts, and other data, as well as monitor hosts, manage monitoring settings, download the Monitoring Agent, and other tasks. For example:

```
mms.ldap.global.role.monitoring_admin=AcmeDbas
```

mms.ldap.global.role.**backup_admin**
Type: string

The LDAP group attribute name for users assigned the global backup administrative role in MMS. This role can view backup status, snapshot lists, and modify backup settings, as well as start/stop/terminate backups, request restores, view/edit host passwords, and other tasks.

```
mms.ldap.global.role.backup_admin=AcmeDbas
```

mms.ldap.global.role.**owner**
Type: string

The LDAP group attribute name for users assigned the global owner role in MMS. This role can perform all administrative tasks in MMS.

```
mms.ldap.global.role.backup_admin=AcmeDbas
```

See also:

User Roles

Kerberos Settings

To enable Kerberos authentication between the MMS application and its *backing database*, configure the following settings. You must configure all required Kerberos settings to enable Kerberos authentication.

jvm.java.security.krb5.**kdc**
Required. The IP/FQDN (Fully Qualified Domain Name) of the KDC server. The value will be set to JVM's java.security.krb5.kdc.

```
jvm.java.security.krb5.kdc=kdc.example.com
```

jvm.java.security.krb5.**realm**
Required. This is the default REALM for Kerberos. It is being used for JVM's java.security.krb5.realm.

```
jvm.java.security.krb5.realm=EXAMPLE.COM
```

mms.kerberos.**principal**
Required. The principal we used to authenticate with MongoDB. This should be the exact same user on the `mongo.mongoUri` above.

```
mms.kerberos.principal=mms/mmsweb.example.com@EXAMPLE.COM
```

mms.kerberos.**keyTab**
Required. The absolute path to the keytab file for the principal.

```
mms.kerberos.keyTab=/path/to/mms.keytab
```

mms.kerberos.**debug**
Optional. The debug flag to output more information on Kerberos authentication process.

```
mms.kerberos.debug=false
```

MongoDB Access Control Considerations

For a MongoDB server with access control, the `mongo.mongoUri` includes the MongoDB user credentials. For example:

```
mongo.mongoUri=mongodb://mongodbuser1:password@mydb1.example.net:40000/admin
```

Encrypt MongoDB User Credentials

If you do not want to store credentials in plain text, On Prem MMS Monitoring provides a tool to encrypt the MongoDB credentials. To encrypt authentication credentials:

1. Issue the following command to create an encrypted credential pair, replacing `<username>` with your username:

   ```
   sudo <install_dir>/bin/credentialstool --username <username> --password
   ```

   This will prompt you to enter the password and will output the encrypted credential pair.

   `credentialstool` requires root privileges, (i.e. `sudo`) when installed with `rpm` or `deb` packages, because it modifies the `/etc/mongodb-mms/gen.key` file.

2. Use the encrypted credential pair in the `mongo.MongoURI` settings where needed, and add the `mongo.encryptedCredentials = true` setting. For example:

   ```
   mongo.mongoUri=mongodb://da83ex3s:a4fbcf3a1@mydb1.example.net:40000/admin
   mongo.encryptedCredentials=true
   ```

Important: The `conf-mms.properties` file can contain multiple `mongo.MongoURI` settings. If `mongo.encryptedCredentials` is `true`, you must encrypt all user credentials found in the various `mongo.MongoURI` settings.

MongoDB User Access

The MongoDB user *must* have the following roles: `readWriteAnyDatabase`, `clusterAdmin`, and `dbAdminAnyDatabase`.

7.2 User Roles

Overview

User roles allow you to grant a user the set of privileges needed to perform tasks but no more.

If you use LDAP authentication for MMS, you must create LDAP groups for each available role described below then assign users to LDAP groups. There is no round trip synchronization between your LDAP server and MMS.

Read Only

The **Read Only** role has the lowest level of privileges. The user can generally see everything in a group, including all monitoring, backup, and automation data, all activity, and all users and user roles. The user, however, cannot modify or delete anything.

User Admin

The **User Admin** role grants access to do the following:

- Add an existing user to a group.
- Invite a new user to a group.
- Remove an existing group invitation.

- Remove a user's request to join a group, which denies the user access to the group.
- Remove a user from a group.
- Modify a user's roles within a group.
- Update the billing email address.

Monitoring Admin

The **Monitoring Admin** role grants all the privileges of the **Read Only** role and grants additional access to do the following:

- Manage alerts (create, modify, delete, enable/disable, acknowledge/unacknowledge).
- Manage hosts (add, edit, delete, enable deactivated).
- Manage dashboards (create, edit, delete).
- Manage group-wide settings.
- Download Monitoring Agent.

Backup Admin

The **Backup Admin** role grants all the privileges of the **Read Only** role and grants access to manage *backups*, including the following:

- Start, stop, and terminate backups.
- Request restores.
- View and edit excluded namespaces.
- View and edit host passwords.
- Modify backup settings.
- Generate SSH keys.
- Download the Backup Agent.

Automation Admin

The **Automation Admin** role grants all the privileges of the **Read Only** role and grants access to manage `automation`, including the following:

- View deployments.
- Provision machines.
- Edit configuration files.
- Modify settings.
- Download the Automation Agent.

Group Owner

The **Group Owner** role has the privileges of all the other roles combined, as well as additional privileges available only to the owner. In addition to the privileges of other roles, a Group Owner can:

- Set up the *Backup* service.
- Update billing information.
- Enable the public API.

Group Roles

The following roles grant privileges within a group:

Global Roles

Global roles have all the same privileges as the equivalent Group roles, except that they have these privileges for all groups. They also have some additional privileges as noted below.

Global Read Only

The **Global Read Only** role grants *read only* access to all groups. The role additionally grants access to do the following:

- View *backups* and other statistics through the *admin* UI.
- Global user search.

Global User Admin

The **Global User Admin** role grants *user admin* access to all groups. The role additionally grants access to do the following:

- Add new groups.
- Manage UI messages.
- Send test emails, SMS messages, and voice calls.
- Edit user accounts.
- Manage LDAP group mappings.

Global Monitoring Admin

The **Global Monitoring Admin** role grants *monitoring admin* access to all groups. The role additionally grants access to do the following:

- View system statistics through the *admin* UI.

Global Backup Admin

The **Global Backup Admin** role grants *backup admin* access to all groups. The role additionally grants access to do the following:

- View system statistics through the *admin* UI.
- Manage blockstore, daemon, and oplog store configurations.
- Move jobs between daemons.
- Approve backups in awaiting provisioning state.

Global Automation Admin

The **Global Automation Admin** role grants *automation admin* access to all groups. The role additionally grants access to view system statistics through the *admin* UI.

Global Owner

The **Global Owner** role for an MMS account has the privileges of all the other roles combined.

7.3 Alert Conditions

Overview

On-Prem MongoDB Management Service provides configurable alert conditions that you can apply to MMS components, such as hosts, clusters, or agents. This document groups the conditions according to the target components to which they apply.

Select alert conditions when configuring alerts, for more information on configuring alerts, see the *Create an Alert Configuration* and *Manage Alerts* documents.

Host Alerts

The Host Alerts are applicable to MongoDB hosts (i.e. `mongos` and `mongod` instances). and are grouped here according to the category monitored.

Host Status

`is down`
: Sends an alert when MMS does not receive a ping from a host for more than 9 minutes. Under normal operation the Monitoring Agent connects to each monitored host about once per minute. MMS will not alert immediately, however, but waits nine minutes in order to minimize false positives, as would occur, for example, during a host restart.

`is recovering`
: Sends an alert when a *secondary* member of a *replica set* enters the `RECOVERING` state. For information on the `RECOVERING` state, see Replica Set Member States.

does not have latest version
This does not apply to On Prem MongoDB Management Service.

Sends an alert when the version of MongoDB running on a host is more than two releases behind. For example if the current production version of MongoDB is 2.6.0 and the previous release is 2.4.9 then a host running version 2.4.8 will trigger this alert but a host running 2.4.9 (previous) 2.6.0 (current) or 2.6.1-rc2 (nightly) will not.

is exposed to the public internet
Sends an alert when the host is exposed to the public internet. When configured, MMS periodically attempts to make a socket connection to your hosts. If MMS is able to connect, MMS triggers the alert. MMS runs this check the 1st and 15th of the month *only*.

Asserts

These alert conditions refer to the metrics found on the host's `asserts` chart. To view the chart, see *Accessing the Host Statistics*.

Asserts: Regular is
Sends an alert if the rate of regular asserts meets the specified threshold.

Asserts: Warning is
Sends an alert if the rate of warnings meets the specified threshold.

Asserts: Msg is
Sends an alert if the rate of message asserts meets the specified threshold. Message asserts are internal server errors. Stack traces are logged for these.

Asserts: User is
Sends an alert if the rate of errors generated by users meets the specified threshold.

Opcounter

These alert conditions refer to the metrics found on the host's `opcounters` chart. To view the chart, see *Accessing the Host Statistics*.

Opcounter: Cmd is
Sends an alert if the rate of commands performed meets the specified threshold.

Opcounter: Query is
Sends an alert if the rate of queries meets the specified threshold.

Opcounter: Update is
Sends an alert if the rate of updates meets the specified threshold.

Opcounter: Delete is
Sends an alert if the rate of deletes meets the specified threshold.

Opcounter: Insert is
Sends an alert if the rate of inserts meets the specified threshold.

Opcounter - Repl

These alert conditions apply to hosts that are *secondary* members of *replica sets*. The alerts use the metrics found on the host's `opcounters - repl` chart. To view the chart, see *Accessing the Host Statistics*.

Opcounter: Repl Update is
Sends an alert if the rate of replicated updates meets the specified threshold.

Opcounter: Repl Delete is
Sends an alert if the rate of replicated deletes meets the specified threshold.

Opcounter: Repl Insert is
Sends an alert if the rate of replicated inserts meets the specified threshold.

Memory

These alert conditions refer to the metrics found on the host's `memory` and `non-mapped virtual memory` charts. To view the charts, see *Accessing the Host Statistics*. For additional information about these metrics, click the *i* icon for each chart.

Memory: Resident is
Sends an alert if the size of the resident memory meets the specified threshold. It is typical over time, on a dedicated database server, for the size of the resident memory to approach the amount of physical RAM on the box.

Memory: Virtual is
Sends an alert if the size of virtual memory for the mongod process meets the specified threshold. You can use this alert to flag excessive memory outside of memory mapping. For more information, click the `memory` chart's *i* icon.

Memory: Mapped is
Sends an alert if the size of mapped memory, which maps the data files, meets the specified threshold. As MongoDB memory-maps all the data files, the size of mapped memory is likely to approach total database size.

Memory: Computed is
Sends an alert if the size of virtual memory that is not accounted for by memory-mapping meets the specified threshold. If this number is very high (multiple gigabytes), it indicates that excessive memory is being used outside of memory mapping. For more information on how to use this metric, view the `non-mapped virtual memory` chart and click the chart's *i* icon.

B-tree

These alert conditions refer to the metrics found on the host's `btree` chart. To view the chart, see *Accessing the Host Statistics*.

B-tree: accesses is
Sends an alert if the number of accesses to B-tree indexes meets the specified average.

B-tree: hits is
Sends an alert if the number of times a B-tree page was in memory meets the specified average.

B-tree: misses is
Sends an alert if the number of times a B-tree page was *not* in memory meets the specified average.

B-tree: miss ratio is
Sends an alert if the ratio of misses to hits meets the specified threshold.

Lock %

This alert condition refers to metric found on the host's `lock %` chart. To view the chart, see *Accessing the Host Statistics*.

Lock % is
: Sends an alert if the amount of time the host is *write locked* meets the specified threshold. For details on this metric, view the `lock %` chart and click the chart's *i* icon.

Background

This alert condition refers to metric found on the host's `background flush avg` chart. To view the chart, see *Accessing the Host Statistics*.

Background Flush Average is
: Sends an alert if the average time for background flushes meets the specified threshold. For details on this metric, view the `background flush avg` chart and click the chart's *i* icon.

Connections

The following alert conditions refer to the metrics found on the host's `connections` chart. To view the chart, see *Accessing the Host Statistics*.

Connections is
: Sends an alert if the number of active connections to the host meets the specified average.

Connections Max is
: Sends an alert if the sum of the total number of active connections plus the total number of remaining connections available on the server meets the specified threshold.

Queues

These alert conditions refer to the metrics found on the host's `queues` chart. To view the chart, see *Accessing the Host Statistics*.

Queues: Total is
: Sends an alert if the number of operations waiting on a *lock* of any type meets the specified average.

Queues: Readers is
: Sends an alert if the number of operations waiting on a *read lock* meets the specified average.

Queues: Writers is
: Sends an alert if the number of operations waiting on a *write lock* meets the specified average.

Cursors

These alert conditions refer to the metrics found on the host's `cursors` chart. To view the chart, see *Accessing the Host Statistics*.

Cursors: Open is
: Sends an alert if the number of cursors the server is maintaining for clients meets the specified average.

Cursors: Timed Out is
: Sends an alert if the number of timed-out cursors the server is maintaining for clients meets the specified average.

Cursors: Client Cursors Size is
: Sends an alert if the cumulative size of the cursors the server is maintaining for clients meets the specified average.

Network

These alert conditions refer to the metrics found on the host's `network` chart. To view the chart, see *Accessing the Host Statistics*.

`Network: Bytes In is`
Sends an alert if the number of bytes sent *to* the database server meets the specified threshold.

`Network: Bytes Out is`
Sends an alert if the number of bytes sent *from* the database server meets the specified threshold.

`Network: Num Requests is`
Sends an alert if the number of requests sent to the database server meets the specified average.

Replication

These alert conditions refer to the metrics found on a *primary's* `replication oplog window` chart or a *secondary's* `replication lag` chart. To view the charts, see *Accessing the Host Statistics*.

`Replication Oplog Window is`
Sends an alert if the approximate amount of time available in the primary's replication *oplog* meets the specified threshold.

`Replication Lag is`
Sends an alert if the approximate amount of time that the secondary is behind the primary meets the specified threshold.

`Replication Headroom is`
Sends an alert when the difference between the primary oplog window and the replication lag time on a secondary meets the specified threshold.

`Oplog Data per Hour is`
Sends an alert when the amount of data per hour being written to a primary's oplog meets the specified threshold.

Page Faults

This alert condition refers to the metric displayed on the host's `page faults` chart. To view the chart, see *Accessing the Host Statistics*.

`Page Faults is`
Sends an alert if the rate of page faults meets the specified threshold.

DB Storage

This alert condition refers to the metric displayed on the host's `db storage` chart. To view the chart, see *Accessing the Host Statistics*.

`DB Storage is`
Sends an alert if the amount of on-disk storage space used by extents meets the specified threshold. Extents are contiguously allocated chunks of datafile space. For more information on extents, see the collStats command.

Journaling

These alert conditions refer to the metrics found on the host's `journal - commits in write lock` chart and `journal stats` chart. To view the charts, see *Accessing the Host Statistics*.

Journaling Commits in Write Lock is
Sends an alert if the rate of commits that occurred while the database was in *write lock* meets the specified average.

Journaling MB is
Sends an alert if the average amount of data written to the recovery log meets the specified threshold.

Journaling Write Data Files MB is
Sends an alert if the average amount of data written to the data files meets the specified threshold.

Replica Set Alerts

These alert conditions are applicable to *replica sets*.

Primary Elected
Sends an alert when a set elects a new *primary*. Each time MMS receives a ping, it inspects the output of the replica set's rs.status() method for the status of each replica set member. From this output, MMS determines which replica set member is the primary. If the primary found in the ping data is different than the current primary known to MMS, this alert triggers.

`Primary Elected` does not always mean that the set elected a *new* primary. `Primary Elected` may also trigger when the same primary is re-elected. This can happen when MMS processes a ping in the midst of an election.

No Primary
Sends an alert when a replica set does not have a *primary*. Specifically, when none of the members of a replica set have a status of `PRIMARY`, the alert triggers. For example, this condition may arise when a set has an even number of voting members resulting in a tie.

If the Monitoring Agent collects data during an election for primary, this alert might send a false positive. To prevent such false positives, set the alert configuration's *after waiting* interval (in the configuration's *Send to* section).

Number of Healthy Members is below
Sends an alert when a replica set has fewer than the specified number of healthy members. If the replica set has the specified number of healthy members or more, MMS triggers no alert.

A replica set member is healthy if its state, as reported in the rs.status() output, is either `PRIMARY` or `SECONDARY`. Hidden secondaries are not counted.

As an example, if you have a replica set with one member in the `PRIMARY` state, two members in the `SECONDARY` state, one hidden member in the `SECONDARY`, one `ARBITER`, and one member in the `RECOVERING` state, then the healthy count is 3.

Number of Unhealthy Members is above
Sends an alert when a replica set has more than the specified number of unhealthy members. If the replica set has the specified number or fewer, MMS sends no alert.

Replica set members are unhealthy when the agent cannot connect to them, or the member is in a rollback or recovering state.

Hidden secondaries are not counted.

Agent Alerts

These alert conditions are applicable to Monitoring Agents and Backup Agents.

Monitoring Agent is down
Sends an alert if the Monitoring Agent has been down for at least 7 minutes. Under normal operation, the

Monitoring Agent sends a ping to MMS roughly once per minute. If MMS does not receive a ping for at least 7 minutes, this alert triggers. However, this alert will never trigger for a group that has no hosts configured.

Important: When the Monitoring Agent is down, MMS will trigger no other alerts. For example, if a host is down there is no Monitoring Agent to send data to MMS that could trigger new alerts.

`Backup Agent is down`
Sends an alert if the Backup Agent has been down for at least 15 minutes. Under normal operation, the Backup Agent periodically sends data to MMS. This alert is never triggered for a group that has no running backups.

`Monitoring Agent is out of date`
Sends an alert when the Monitoring Agent is not running the latest version of the software.

Backup Alerts

These alert conditions are applicable to the MMS Backup service.

`Oplog Behind`
Sends an alert if the most recent *oplog* data received by MMS is more than 75 minutes old.

`Resync Required`
Sends an alert if the replication process for a backup falls too far behind the *oplog* to catch up. This occurs when the host overwrites oplog entries that backup has not yet replicated. When this happens, backup must be fully resynced.

User Alerts

These alert conditions are applicable to the MMS Users.

`Added to Group`
Sends an alert when a new user joins the group.

`Removed from Group`
Sends an alert when a user leaves the group.

`Changed Roles`
Sends an alert when a user's roles have been changed.

7.4 MMS Agent Authentication Requirements

If your MongoDB deployment requires authentication, On-Prem MongoDB Management Service requires additional privileges to collect complete data from MongoDB 2.4 and MongoDB 2.6 instances. If authentication is not enabled on your MongoDB instance, this requirement does not apply.

The MMS Monitoring and MMS Backup Agents authenticate to your MongoDB instances using the username and password that you provide in the UI. This document describes the requirements for those users.

Define all user accounts on the `admin` database.

If your MongoDB deployment uses Kerberos authentication, the On-Prem MongoDB Management Service agents can connect to MongoDB hosts using Kerberos authentication. See *Connect to Hosts with Kerberos Authentication* for more information.

MMS Backup

To support backup for sharded clusters create both shard-local users on each shard, as well cluster-wide users. Create cluster users while connected to the `mongos`: these credentials persist to the config servers. Create shard-local users by connecting directly to the replica set for each shard.

MongoDB 2.6

To backup MongoDB 2.6 instances, the Backup Agent must be able to authenticate to with the followsing roles:

- `clusterAdmin`
- `readAnyDatabase`
- `userAdminAnyDatabase`

And the following `otherDBRoles`:

- `readWrite` role on the `local` database
- `readWrite` role on the `admin` database

Create a user with this roles on the `admin` database with an operation that resembles the following:

```
use admin
db.createUser( { user: "<username>",
                 pwd: "<password>",
                 roles: [ "clusterAdmin", "readAnyDatabase",
                          "userAdminAnyDatabase",
                          { role: "readWrite", db: "admin" },
                          { role: "readWrite", db: "local" },
                        ] } )
```

MongoDB 2.4

To backup MongoDB 2.4 instances, the Backup Agent must be able to authenticate to the database with a user that has following roles:

- `clusterAdmin`
- `readAnyDatabase`
- `userAdminAnyDatabase`

And the following `otherDBRoles`:

- `readWrite` role on the `local` database
- `readWrite` role on the `admin` database

Create a user on the `admin` database with an operation that resembles the following:

```
use admin
db.addUser( { user: "<username>",
              pwd: "<password>",
              roles: [ "clusterAdmin",
                       "readAnyDatabase",
                       "userAdminAnyDatabase"
                     ],
               otherDBRoles: { local: ['readWrite'],
                               admin: ['readWrite'] } } )
```

MMS Monitoring

MongoDB 2.6

To monitor MongoDB 2.6 instances, the agent must be able to authenticate to the database with a user that has the `clusterMonitor` role.

Create a user with this role on the `admin` database with an operation that resembles the following:

```
use admin
db.createUser( { user: "<username>",
                 pwd: "<password>",
                 roles: [
                   { role: "clusterMonitor", db: "admin" }
                 ] } )
```

Additionally, the agent may attempt to query the local database in order to provide compatibility with 2.4 and mixed deployments. With only the `clusterMonitor` role, this will produce an authentication error that will appear in the `mongod` logs. The agent can recover from this error and you can safely ignore these messages in the `mongod` log.

MongoDB 2.4

To monitor MongoDB 2.4 instances, the agent must be able to authenticate to the database with a user that have the following roles:

- `clusterAdmin`
- `readAnyDatabase`

Create a user with these roles on the `admin` database with an operation that resembles the following:

```
use admin
db.addUser( { user: "<username>",
              pwd: "<password>",
              roles: [ "clusterAdmin",
                       "readAnyDatabase" ] } )
```

Because user accounts created for basic monitoring do not require the `dbAdminAnyDatabase` role, the `mongod` log file may report the following messages at the default logging level:

```
command denied: { profile: -1 }
```

You can ignore this message if you do not want MMS to collect profile data. If you want to collect profile data, configure MMS monitoring with database profiling as described below.

MMS Monitoring with Database Profiling

Profiling captures in-progress read and write operations, cursor operations, and database command information about the database.

MongoDB 2.6

MMS monitoring of MongoDB 2.6 databases with database profiling requires authenticated users to have the `clusterMonitor` role.

```
use admin
db.createUser( { user: "<username>",
                 pwd: "<password>",
                 roles: [ { role: "clusterMonitor", db: "admin" } ] } )
```

MongoDB 2.4

MMS monitoring of MongoDB 2.4 databases with database profiling requires authenticated users to have these roles:

- `clusterAdmin`
- `readAnyDatabase`
- `dbAdminAnyDatabase`

Create a user with these roles on the `admin` database with an operation that resembles the following:

```
use admin
db.addUser( { user: "<username>",
              pwd: "<password>",
              roles: [ "clusterAdmin",
                       "readAnyDatabase",
                       "dbAdminAnyDatabase"
                     ] } )
```

MMS Monitoring *without* `dbStats`

Monitoring without `dbStats` will generate monitoring statistics without database storage, records, indexes, and other statistics.

MongoDB 2.6

The `clusterMonitor` includes access to the `dbStats` operations. Create a user on the `admin` database with `clusterMonitor` as in the following example.

```
use admin
db.createUser( { user: "<username>",
                 pwd: "<password>",
                 roles: [ { role: "clusterMonitor", db: "admin" } ] } )
```

MongoDB 2.4

To monitor MongoDB 2.4 instance *with* database profiling that requires authenticated users have the `clusterAdmin` role.

To provision this access, create a user with this role on the `admin` database with an operation that resembles the following:

```
use admin
db.addUser( { user: "<username>",
              pwd: "<password>",
              roles: [ "clusterAdmin" ] } )
```

7.5 On Prem MMS Reference

MMS On Prem components use the default ports and health-check endpoints described here.

Ports

Although the following components are logically distinct, the *MMS Application and Monitoring Server*, *MMS Backup Ingestion Server*, and *MMS Backup Alerts Service* are all part of the MMS Application package and typically run on a single system. The MMS Backup Daemon typically runs on a distinct system.

MMS Application and Monitoring Server

8080
: Web server, available from browsers. The system running the Monitoring Agent must be able to access this port as well.

MMS Backup Ingestion Server

8081
: Available from browsers for HTTP restores, and from systems running the Backup Agent.

8091
: *Optional*

  Used for internal diagnostics. Only available on the localhost interface.

MMS Backup Alerts Service

8650
: A "kill-port" that the control script uses to signal a shutdown.

  Only available on the localhost interface.

8092
: *Optional*

  Used for internal diagnostics. Only available on the localhost interface.

MMS Backup Daemon

8640
: A "kill-port" that the control script uses to signal a shutdown.

  Only available on the localhost interface.

27500–27503
: The backup daemon uses this port range to on the localhost interface to run `mongod` instances to apply oplog entries to maintain the local copies of the backed up database.

8090
: *Optional*

  Used for internal diagnostics. Only available on the localhost interface.

Monitoring HTTP Endpoints

MMS On Prem provides health-check endpoints for the monitoring of the On Prem components via a standard monitoring service, such as Zabbix or Nagios. These endpoints are only accessible on the `localhost` interface.

Backup HTTP Service Endpoint

The *Backup HTTP Service* on the *MMS Application and Monitoring Server* exposes the following endpoint:

```
http://localhost:8091/health
```

The endpoint checks the connections from the service to the *MMS Application Database* and the *MMS Backup Blockstore Database*.

A successful return from the endpoint returns the following:

```
{
  "mms_db": "OK",
  "backup_db": "OK"
}
```

Backup Alert Service Endpoint

The *Backup Alert Service* on the *MMS Application and Monitoring Server* exposes the following health-check endpoint:

```
http://localhost:8092/health
```

Backup Daemon Endpoint

The *Backup Daemon* on the Backup Daemon server exposes a health-check endpoint at:

```
http://localhost:8090/health
```

7.6 Monitoring Reference

This document contains references of the different types of hosts, databases, and other statuses that may occur in On Prem MMS Monitoring.

Host Types

The possible values for the "Type" column in the Hosts page are:

- primary
- secondary
- standalone
- master
- slave
- unknown

- recovering

The "Host Type" selector on the advanced dashboard creator also includes:

- conf
- mongos

Note: The host type column may also have the value "no data," which means that On Prem MMS Monitoring has not received any data from the Monitoring Agent for this host. Possible causes for this state:

- If the Monitoring Agent can't connect to the server because of networking restrictions or issues (i.e. firewalls, proxies, routing.)
- If your database is running with SSL. You must enable SSL either globally or on a per-host basis. See *Using SSL with On Prem MMS Monitoring* for more information.
- If your database is running with authentication. You must supply On Prem MMS Monitoring with the authentication credentials either when you're adding a host *or* by clicking on the edit (i.e. "Pencil" button) on the right of the entry on the "Hosts" page.

Host Process Types

On Prem MMS Monitoring can monitor the process types:

- `mongod` database processes
- `mongod` arbiter processes
- `mongos`
- Monitoring Agents

Event Types

Types of events in the Events section of the MMS console:

- new host
- restart
- upgrade

Alert Types

The available alert types are:

- Old Host Version
- Host Down
- Agent Down
- Now Secondary
- Now Primary

Chart Colors

- A *red bar* indicates a server restart.
- A *purple bar* indicates the server is now a primary.
- A *yellow bar* indicates the server is now a secondary.

Status Page

- cpu time
- db storage
- page faults
- repl lag
- replica
- network
- cursors
- queues
- connections
- background flush avg
- lock % [2]
- btree
- non-mapped virtual memory
- memory
- asserts
- opcounters-repl
- opcounters

DB Stats Page

- collections
- objects
- average object size
- data size
- storage size
- num extents
- indexes
- index size

[2] For versions of MongoDB after 2.1.1, this chart has a drop-down menu next to the tile that lists available databases, including "global" to represent the global lock for this host. Select a database to see its lock utilization. See the documentation of lock reporting in serverStatus for more information.

- file size

Database Commands Used by the Monitoring Agent

- serverStatus
- buildinfo
- getCmdLineOpts
- connPoolStats
- _isSelf
- getParameter
- ismaster
- getShardVersion
- netstat
- replSetGetStatus
- shards.find
- mongos.find
- config.chunks.group
- oplog.find
- collstats - oplog.rs
- sources.find (slave)
- config.settings.find
- dbstats
- db.locks

7.7 Supported Browsers

On-Prem MongoDB Management Service supports clients using the following browsers:

- Chrome 8 and greater.
- Firefox 12 and greater.
- IE 9 and greater.
- Safari 6 and greater.

The MMS application will display a warning on non-supported browsers.

7.8 MMS Public API

The MMS Public API follows the principles of the REST architectural style to expose a number of internal resources which enable programmatic access to MMS's features. Some highlights of API include:

- **JSON throughout** - All entities are expressed in JSON.

- **Digest authentication** - To ensure that your API key is never sent over the network, API requests are authenticated using HTTP Digest Authentication.
- **Browsable interface** - Using a consistent linking mechanism, you can browse the entire API by starting at the root resource and following links to related resources.

Resources

Root

This is the starting point (or the homepage, if you will) for the MMS API. From here, you can traverse the `links` to reach all other API resources.

Sample Entity

```
{
  "throttling": false,
  "links": [ ... ]
}
```

Entity Fields

Name	Type	Description
throttling	boolean	Tells whether or not MMS is throttling data. This can be used as a simple indicator of the current health of MMS, since throttling is generally enabled when MMS is in an unhealthy state.

Links

Relation	Description
self	Me
groups	Groups accessible to the current API user.
user	The current API user.

Example Retrieve the root resource:

```
curl -u "<username>:<apiKey>" "https://mms.mongodb.com/api/public/v1.0" --digest -i

HTTP/1.1 200 OK

{
  "throttling" : false,
  "links" : [ ... ]
}
```

Hosts

Sample Entity

```
{
  "id": "680ab316473d6b28f966364b947134fc",
  "groupId": "2847387cd717dabc348a",
  "hostname": "localhost",
  "port": 27017,
  "typeName": "SHARD_SECONDARY",
  "lastPing": "2014-02-15T16:03:47Z",
  "ipAddress": "127.0.0.1",
  "version": "2.4.3",
```

```
  "deactivated": false,
  "hasStartupWarnings": true,
  "sslEnabled": false,
  "logsEnabled": false,
  "lastReactivated": "2013-12-15T09:17:23Z",
  "uptimeMsec": 48918394,
  "lastRestart": "2014-01-16T12:34:01Z",
  "shardName": "sh1",
  "replicaSetName": "rs1",
  "replicaStateName": "RECOVERING",
  "created": "2013-11-05T03:04:05Z",
  "hostEnabled": true,
  "journalingEnabled": false,
  "alertsEnabled": true,
  "hidden": false,
  "muninEnabled": false,
  "profilerEnabled": false,
  "lowUlimit": false,
  "muninPort": 4949,
  "authMechanismName": "MONGODB_CR",
  "username": "mongo",
  "links": [ ... ]
}
```

Name	Type	Description
id	string	Unique identifier.
groupId	string	ID of the group that owns this host.
hostname	string	Primary hostname. A host typically has several aliases, so the primary is the best available name a
port	integer	Port that MongoDB process (`mongod` or `mongos`) listens on.
typeName	enum	Type for this host. Possible values are: `STANDALONE REPLICA_PRIMARY REPLICA_SECO`
lastPing	date	When the last ping for this host was received.
ipAddress	string	IP address of this host.
version	string	Version of MongoDB running on this host.
deactivated	boolean	Has this host been deactivated by MMS? A host will be marked as deactivated when MMS hasn't
hasStartupWarnings	boolean	Are there startup warnings for this host?
sslEnabled	boolean	Is SSL enabled for this host?
logsEnabled	boolean	Is MMS collecting logs for this host?
lastReactivated	date	The last time this has was manually reactivated.
uptimeMsec	long	Number of milliseconds since this host's last restart.
lastRestart	date	Date this host was last restarted.
shardName	string	Name of the shard this host belongs to. Only present if the host is part of a sharded cluster.
replicaSetName	string	Name of the replica set this host belongs to. Only present if this host is part of a replica set.
replicaStateName	enum	Current state of this host within a replica set. Only present if this host is part of a replica set. See I
created	date	Date this host was created or first discovered by MMS.
hostEnabled	boolean	Is this host currently enabled? Hosts can be manually disabled in the MMS UI.
journalingEnabled	boolean	Is journaling enabled for this host?
alertsEnabled	boolean	Are alerts enabled for this host?
muninEnabled	boolean	Are Munin stats being collected for this host?
hidden	boolean	Is this host currently hidden? When MMS deactivates a host, it will also mark it as hidden.
profilerEnabled	boolean	Is MMS collecting profile information from this host?
lowUlimit	boolean	Does this host have a low `ulimit` setting?
muninPort	integer	What port should be used to collect Munin stats from this host?
authMechanismName	enum	The authentication mechanism used to connect to this host. Possible values are: `MONGODB_CR G`

Name	Type	Description
username	string	Username for connecting to this host. Only present when the `authMechanismName` is `MONGOD`
password	string	Password for connecting to this host. If a host's `authMechanismName` is `MONGODB_CR`, then

Entity Fields

Links

Relation	Description
self	Me
cluster	The cluster this host belongs to. Only present if the host is part of a replica set or master/slave.
parentCluster	The parent cluster. Only present if the host is part of a sharded cluster.
group	The group that this host belongs to.

Operations

- `GET /api/public/v1.0/groups/GROUP-ID/hosts` - Get all hosts in a group. Use the `clusterId` query parameter to only get the hosts that belong to the specified cluster. The resulting list is sorted alphabetically by `hostname:port`.
- `GET /api/public/v1.0/groups/GROUP-ID/hosts/HOST-ID` - Get a single host by ID.
- `POST /api/public/v1.0/groups/GROUP-ID/hosts` - Create a new host in the group. Note that after a new host is created, MMS will not know much about it except what is provided. Thus, the document returned in the response will be missing many values until they are discovered, which could take several minutes. Only these fields may be specified when creating a host:
  - `hostname` - Required.
  - `port` - Required.
  - `username` - If `authMechanismName` is `MONGODB_CR`, this field is required. Otherwise it's illegal.
  - `password` - If `authMechanismName` is `MONGODB_CR`, this field is required. Otherwise it's illegal.
  - `sslEnabled` - Default is `false` if omitted.
  - `logsEnabled` - Default is `false` if omitted.
  - `alertsEnabled` - Default is `true` if omitted.
  - `profilerEnabled` - Default is `false` if omitted.
  - `muninPort` - Default is 0 and Munin stats are not collected if omitted.
  - `authMechanismName` - Default is `NONE` if omitted. If set to `MONGODB_CR` then you must provide the `username` and `password`.
- `PATCH /api/public/v1.0/groups/GROUP-ID/hosts/HOST-ID` - Update an existing host using the fields provided. Unspecified fields will preserve their current values.
  - Only these fields may be specified: `username password sslEnabled logsEnabled alertsEnabled profilerEnabled muninPort authMechanismName`
  - If `authMechanismName` is `NONE` then any existing value for `username` and `password` will be cleared out. For `MONGODB_CR` you must provide both `username` and `password`.
- `DELETE /api/public/v1.0/groups/GROUP-ID/hosts/HOST-ID` - Remove a host.

Examples Create a new host:

```
curl -u "username:apiKey" -H "Content-Type: application/json" "https://mms.mongodb.com/api/public/v1
{
  "hostname": "localhost",
  "port": 27017
}

HTTP/1.1 201 Created
Location: https://mms.mongodb.com/api/public/v1.0/groups/5196d3628d022db4cbc26d9e/hosts/680ab316473d6

{
  "id" : "4059580c20c4581872ef24d0b8f5dca0",
  "groupId" : "5196d3628d022db4cbc26d9e",
  "hostname" : "localhost",
  "port" : 27017,
  "deactivated" : false,
  "hasStartupWarnings" : false,
  "sslEnabled" : false,
  "logsEnabled" : false,
  "created" : "2014-04-22T19:56:50Z",
  "hostEnabled" : true,
  "journalingEnabled" : false,
  "alertsEnabled" : true,
  "hidden" : false,
  "muninEnabled" : false,
  "profilerEnabled" : false,
  "lowUlimit" : false,
  "authMechanismName" : "NONE",
  "links" : [ ... ]
}
```

Update a host:

```
curl -u "username:apiKey" -H "Content-Type: application/json" "https://mms.mongodb.com/api/public/v1
{
  "sslEnabled": true,
  "username": "mongo",
  "password": "M0ng0DB!:)"
}

HTTP/1.1 200 OK

{
  "id" : "680ab316473d6b28f966364b947134fc",
  "groupId" : "533c5895b91030606f21033a",
  "hostname" : "localhost",
  "port" : 26000,
  "deactivated" : false,
  "hasStartupWarnings" : false,
  "sslEnabled" : true,
  "logsEnabled" : false,
  "created" : "2014-04-22T19:56:50Z",
  "hostEnabled" : true,
  "journalingEnabled" : false,
  "alertsEnabled" : true,
  "hidden" : false,
  "muninEnabled" : false,
  "profilerEnabled" : false,
```

```
  "lowUlimit" : false,
  "authMechanismName" : "MONGODB_CR",
  "username" : "mongo",
  "links" : [ ... ]
}
```

Get one host:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/533c5895b91030606f21033a/ho

HTTP/1.1 200 OK

{
  "id" : "56e9378f601dc49360a40949c8a6df6c",
  "groupId" : "533c5895b91030606f21033a",
  "hostname" : "localhost",
  "port" : 26000,
  "deactivated" : false,
  "hasStartupWarnings" : false,
  "sslEnabled" : true,
  "logsEnabled" : false,
  "created" : "2014-04-22T19:56:50Z",
  "hostEnabled" : true,
  "journalingEnabled" : false,
  "alertsEnabled" : true,
  "hidden" : false,
  "muninEnabled" : false,
  "profilerEnabled" : false,
  "lowUlimit" : false,
  "authMechanismName" : "MONGODB_CR",
  "username" : "mongo",
  "links" : [ ... ]
}
```

Get all hosts:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/533c5895b91030606f21033a/ho

HTTP/1.1 200 OK

{
  "totalCount" : 2,
  "results" : [ {
    "id" : "56e9378f601dc49360a40949c8a6df6c",
    "groupId" : "533c5895b91030606f21033a",
    "hostname" : "localhost",
    "port" : 26000,
    "deactivated" : false,
    "hasStartupWarnings" : false,
    "sslEnabled" : true,
    "logsEnabled" : false,
    "created" : "2014-04-22T19:56:50Z",
    "hostEnabled" : true,
    "journalingEnabled" : false,
    "alertsEnabled" : true,
    "hidden" : false,
    "muninEnabled" : false,
    "profilerEnabled" : false,
    "lowUlimit" : false,
```

```
    "authMechanismName" : "MONGODB_CR",
    "username" : "mongo",
    "links" : [ ... ]
  }, {
    ...
  } ]
}
```

Delete a host:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/533c5895b91030606f21033a/ho

HTTP/1.1 200 OK
```

Clusters

Sample Entity

```
{
  "id": "yyy",
  "groupId": "xxx",
  "typeName": "REPLICA_SET",
  "clusterName": "Cluster 0",
  "shardName": "shard001",
  "replicaSetName": "rs1",
  "lastHeartbeat": "2014-02-26T17:32:45Z",
  "links": [ ... ]
}
```

Entity Fields

Name	Type	Description
id	string	Unique identifier.
groupId	string	ID of the group that owns this cluster.
typeName	enum	Specifies what kind of cluster this is. Possible values are: `MASTER_SLAVE` `REPLICA_SET` `SHARDED` `SHARDED_REPLICA_SET`
clusterName	string	Display name of the cluster. Only applies to sharded clusters. Note that `mongod` itself doesn't allow you to name a cluster; this name is supplied by (and editable within) MMS. For a replica set within a sharded cluster, the cluster name is the name of its parent cluster.
shardName	string	Name of the shard. Only present for a cluster of type `SHARDED` or `REPLICA_SET` that is part of a sharded cluster.
replicaSetName	string	Name of the replica set. Only present for a cluster of type `REPLICA_SET`.
lastHeartbeat	date	The approximate last time MMS processed a ping from this cluster.

Links

Relation	Description
self	Me
parentCluster	The parent cluster. Only present if the type is `SHARDED` or `REPLICA_SET` within a sharded cluster.
group	The group that this cluster belongs to.
clusters	The member shards that belong to this cluster. Only present if the type is `SHARDED_REPLICA_SET`.
hosts	The member hosts that belong to this cluster. Present for all types except `SHARDED_REPLICA_SET`. Note: to get the hosts of a sharded cluster, follow the `clusters` link and get the `hosts` for each shard.

Operations

- `GET /api/public/v1.0/groups/GROUP-ID/clusters/CLUSTER-ID` - Get a single cluster by ID.
- `GET /api/public/v1.0/groups/GROUP-ID/clusters` - Get all clusters in a group. Note that if MMS hasn't received a ping from a cluster in several days, it will be considered inactive and will be filtered from this list. Use the `parentClusterId` query parameter to get all clusters with the specified parent cluster ID. The list of entities is sorted in ascending order by the date that MMS discovered the cluster.
- `PATCH /api/public/v1.0/groups/GROUP-ID/clusters/CLUSTER-ID` - Update a cluster by ID. The only property that you may modify is the `clusterName`, since all other properties of a cluster are discovered by MMS. Additionally, this operation is only permitted on clusters of type `SHARDED` and `SHARDED_REPLICA_SET`.

Examples Get a cluster:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/533c5895b91030606f21033a/cl

HTTP/1.1 200 OK

{
  "id" : "533d7d4730040be257defe88",
  "typeName" : "SHARDED_REPLICA_SET",
  "clusterName" : "Animals",
  "lastHeartbeat" : "2014-04-03T15:26:58Z",
  "links" : [ ... ]
}
```

Get all clusters:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/533c5895b91030606f21033a/cl

HTTP/1.1 200 OK

{
  "totalCount" : 3,
  "results" : [ {
    "id" : "533d7d4730040be257defe88",
    "typeName" : "SHARDED_REPLICA_SET",
    "clusterName" : "Animals",
    "lastHeartbeat" : "2014-04-03T15:26:58Z",
    "links" : [ ... ]
  }, {
    "id" : "533d7d4630040be257defe85",
    "typeName" : "REPLICA_SET",
```

```
    "clusterName" : "Animals",
    "shardName" : "cats",
    "replicaSetName" : "cats",
    "lastHeartbeat" : "2014-04-03T15:24:54Z",
    "links" : [ ... ]
  }, {
    "id" : "533d7d4630040be257defe83",
    "typeName" : "REPLICA_SET",
    "clusterName" : "Animals",
    "shardName" : "dogs",
    "replicaSetName" : "dogs",
    "lastHeartbeat" : "2014-04-03T15:26:30Z",
    "links" : [ ... ]
  } ],
 "links" : [ ... ]
}
```

Update a cluster:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/533c5895b91030606f21033a/cl
{
  "clusterName": "Zoo"
}

HTTP/1.1 200 OK

{
  "id" : "533d7d4730040be257defe88",
  "typeName" : "SHARDED_REPLICA_SET",
  "clusterName" : "Zoo",
  "lastHeartbeat" : "2014-04-03T15:26:58Z",
  "links" : [ ... ]
}
```

Groups

Sample Entity

```
{
  "id": "xxx",
  "name": "My Group",
  "hostCounts": {
    "arbiter": 2,
    "config": 1,
    "primary": 4,
    "secondary": 8,
    "mongos": 2.
    "master": 0,
    "slave": 0
  },
  "lastActiveAgent": ISODate("2014-02-05T07:23:34Z"),
  "activeAgentCount": 1,
  "replicaSetCount": 3,
  "shardCount": 2,
  "publicApiEnabled": true,
  "links": [ ... ]
}
```

Entity Fields

Name	Type	Description
id	string	Unique identifier.
name	string	Display name for the group.
hostCounts	object	The total number of hosts by type. The embedded fields should be self-explanatory.
lastActiveAgent	date	Date that a ping was last received from one of the group's Monitoring Agents.
activeAgentCount	integer	Number of Monitoring Agents sending regular pings to MMS.
replicaSetCount	integer	Total number of replica sets for this group.
shardCount	integer	Total number of shards for this group.
publicApiEnabled	boolean	Is the Public API enabled for this group? This is a read-only field that will always be true for groups created with the API. Note that for groups created in the MMS UI, the only way to set this flag to `true` is by enabling the Public API for the group in the **Settings** tab.

Links

Relation	Description
self	Me
hosts	All hosts in the group.
users	All users in the group.
clusters	All clusters in the group.
alerts	All open alerts for the group.
alertConfigs	All alert configurations for the group.
backupConfigs	All backup configurations for the group.

Operations

- `GET /api/public/v1.0/groups/GROUP-ID` - Get a single group by ID.
- `GET /api/public/v1.0/groups` - Get all groups for the current user.
- `POST /api/public/v1.0/groups` - Create a new group. Only the `name` field may be specified. The `publicApiEnabled` field will be set to `true` for groups created with the API.
- `GET /api/public/v1.0/groups/GROUP-ID/users` - Get all users in a group.
- `DELETE /api/public/v1.0/groups/GROUP-ID/users/USER-ID` - Remove a user from a group.
- `POST /api/public/v1.0/groups/GROUP-ID/users` - Add existing user(s) to a group.
  - You must send an array of entities, even if you're only adding a single user.
  - For each user being added, specify the user ID and role(s) to be assigned.
  - If a user is specified that is already part of the group, then their existing role(s) will be overwritten.
- `DELETE /api/public/v1.0/groups/GROUP-ID` - Delete a group. Once a group is deleted, its name cannot be reclaimed. Thus, if you create a group named **My Group** and then delete it, you will not be able to create another group named **My Group**.

Examples Get a group:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/5196d3628d022db4cbc26d9e" -

HTTP/1.1 200 OK

{
  "id" : "5196d3628d022db4cbc26d9e",
  "name" : "API Example",
  "hostCounts" : {
    "arbiter" : 0,
    "config" : 1,
    "primary" : 3,
    "secondary" : 4,
    "mongos" : 2,
    "master" : 0,
    "slave" : 0
  },
  "lastActiveAgent" : "2014-04-03T18:18:12Z",
  "activeAgentCount" : 1,
  "replicaSetCount" : 3,
  "shardCount" : 2,
  "links" : [ ... ]
}
```

Get all groups for current user:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups" --digest -i

HTTP/1.1 200 OK

{
  "totalCount" : 6,
  "results" : [ {
    "id" : "5196d3628d022db4cbc26d9e",
    "name" : "API Example",
    "hostCounts" : {
      "arbiter" : 0,
      "config" : 1,
      "primary" : 3,
      "secondary" : 4,
      "mongos" : 2,
      "master" : 0,
      "slave" : 0
    },
    "lastActiveAgent" : "2014-04-03T18:18:12Z",
    "activeAgentCount" : 1,
    "replicaSetCount" : 3,
    "shardCount" : 2,
    "links" : [ ... ]
  }, {
    // etc.
  } ],
  "links" : [ ... ]
}
```

Create a group:

```
curl -u "username:apiKey" -H "Content-Type: application/json" "https://mms.mongodb.com/api/public/v1
{
  "name": "API Example 2"
```

```
}

HTTP/1.1 201 Created
Location: https://mms.mongodb.com/api/public/v1.0/groups/533daa30879bb2da07807696

{
  "id" : "533daa30879bb2da07807696",
  "name" : "API Example 2",
  "activeAgentCount" : 0,
  "replicaSetCount" : 0,
  "shardCount" : 0,
  "hostCounts" : {
    "arbiter" : 0,
    "config" : 0,
    "primary" : 0,
    "secondary" : 0,
    "mongos" : 0,
    "master" : 0,
    "slave" : 0
  },
  "links" : [ ... ]
}
```

Add users to a group:

```
$ curl -u "username:apiKey" -H "Content-Type: application/json" "https://mms.mongodb.com/api/public/v
[
  {
    "id": "5329c8dfe4b0b07a83d67e7d",
    "roles": [{
      "roleName": "GROUP_READ_ONLY"
    }]
  },
  {
    "id": "5329c906e4b0b07a83d691ba",
    "roles": [{
      "roleName": "GROUP_MONITORING_ADMIN"
    }, {
      "roleName": "GROUP_BACKUP_ADMIN"
    }]
  }
]

HTTP/1.1 200 OK
```

Delete a group:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/533daa30879bb2da07807696" -

HTTP/1.1 200 OK
```

Get users in a group:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/5356823bc0edc2788a835ed0/us

HTTP/1.1 200 OK

{
  "totalCount" : 2,
```

```
  "results" : [ {
    "id" : "5357e25a300490374243f425",
    "username" : "user1@foo.com",
    "emailAddress" : "user1@foo.com",
    "firstName" : "User",
    "lastName" : "One",
    "roles" : [ {
      "groupId" : "5356823bc0edc2788a835ed0",
      "roleName" : "GROUP_USER_ADMIN"
    } ],
    "links" : [ ... ]
  }, {
    "id" : "5356823b3004dee37132bb7b",
    "username" : "user2@foo.com",
    "emailAddress" : "user2@foo.com",
    "firstName" : "User",
    "lastName" : "Deux",
    "roles" : [ {
      "groupId" : "5356823bc0edc2788a835ed0",
      "roleName" : "GROUP_OWNER"
    }, {
      "groupId" : "5356823bc0edc2788a835ecd",
      "roleName" : "GROUP_OWNER"
    } ],
    "links" : [ ... ]
  } ],
  "links" : [ ... ]
}
```

Delete a user from a group:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/5356823bc0edc2788a835ed0/us

HTTP/1.1 200 OK
```

Users

Sample Entity

```
{
  "id": "xxx",
  "username": "somebody@somewhere.com",
  "password": "abc123",
  "emailAddress": "somebody@somewhere-else.com",
  "mobileNumber": "2125551234",
  "firstName": "John",
  "lastName": "Doe",
  "roles": [
      {
        "groupId": "8491812938cbda83918c",
        "roleName": "GROUP_OWNER"
      },
      {
        "groupId": "4829cbda839cbdac3819",
        "roleName": "GROUP_READ_ONLY"
      }
  ],
  "links": [ ... ]
```

```
}
```

Entity Fields

Name	Type	Description
id	string	Unique identifier.
username	string	MMS username.
password	string	Password. This field is NOT included in the entity returned from the server. It can only be sent in the entity body when creating a new user.
emailAddress	string	Email address.
mobileNumber	string	Mobile number. This field can only be set or edited using the MMS UI because it is tied to two factor authentication.
firstName	string	First name.
lastName	string	Last name.
roles	object array	Role assignments.
roles.groupId	string	The groupId in which the user has the specified role. Note that for the "global" roles (those whose name starts with `GLOBAL_`) there is no `groupId` since these roles are not tied to a group.
roles.roleName	enum	The name of the role. Possible values are: `GLOBAL_AUTOMATION_ADMIN` `GLOBAL_BACKUP_ADMIN` `GLOBAL_MONITORING_ADMIN` `GLOBAL_OWNER` `GLOBAL_READ_ONLY` `GLOBAL_USER_ADMIN` `GROUP_AUTOMATION_ADMIN` `GROUP_BACKUP_ADMIN` `GROUP_MONITORING_ADMIN` `GROUP_OWNER` `GROUP_READ_ONLY` `GROUP_USER_ADMIN`

Links

Relation	Description
self	Me
whitelist	The user's whitelist.

Operations

- `GET /api/public/v1.0/users/USER-ID/xxx` - Get a single user by ID. You can only retrieve a user if you have at least one group in common.
- `GET /api/public/v1.0/groups/GROUP-ID/users` - Get all users in a group.
- `POST /api/public/v1.0/users` - Create a new user. All fields are required.
- `PATCH /api/public/v1.0/users/USER-ID` - Update an existing user using the fields provided. Unspecified fields will preserve their current values. You cannot specify the password for security reasons.

Examples Get a user:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/users/533dc19ce4b00835ff81e2eb" --

HTTP/1.1 200 OK

{
  "id" : "533dc19ce4b00835ff81e2eb",
  "username" : "jane.doe@mongodb.com",
```

```
  "emailAddress" : "doh.jane@gmail.com",
  "firstName" : "Jane",
  "lastName" : "D'oh",
  "roles" : [ {
    "groupId" : "533daa30879bb2da07807696",
    "roleName" : "GROUP_USER_ADMIN"
  } ],
  "links": [ ... ]
}
```

Get all users in a group:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/533daa30879bb2da07807696/us

HTTP/1.1 200 OK

{
  "totalCount": 3,
  "results": [ {
    "id" : "5329c8dfe4b0b07a83d67e7d",
    "username" : "john.doe@somewhere.com",
    "emailAddress" : "john.doe@somewhere.com",
    "firstName" : "John",
    "lastName" : "Doe",
    "roles" : [ {
      "groupId" : "5329cb6e879bb2da07806511",
      "roleName" : "GROUP_OWNER"
    }, {
      "groupId" : "5196d3628d022db4cbc26d9e",
      "roleName" : "GROUP_READ_ONLY"
    }, {
      "groupId" : "533daa30879bb2da07807696",
      "roleName" : "GROUP_READ_ONLY"
    } ],
    "links": [ ... ]
  }, {
    // etc.
  } ],
  "links": [ ... ]
}
```

Create a user:

```
curl -u "username:apiKey" -H "Content-Type: application/json" "https://mms.mongodb.com/api/public/v1
{
  "username": "jane.doe@mongodb.com",
  "emailAddress": "jane.doe@mongodb.com",
  "firstName": "Jane",
  "lastName": "Doe",
  "password": "M0ng0D8!:)",
  "roles": [{
    "groupId": "533daa30879bb2da07807696",
    "roleName": "GROUP_USER_ADMIN"
  }]
}

HTTP/1.1 201 Created
Location: https://mms.mongodb.com/api/public/v1.0/users/533dc19ce4b00835ff81e2eb
```

```
{
  "id" : "533dc19ce4b00835ff81e2eb",
  "username" : "jane.doe@mongodb.com",
  "emailAddress" : "jane.doe@mongodb.com",
  "firstName" : "Jane",
  "lastName" : "Doe",
  "roles" : [ {
    "groupId" : "533daa30879bb2da07807696",
    "roleName" : "GROUP_USER_ADMIN"
  } ],
  "links" : [ ... ]
}
```

Update a user:

```
curl -u "username:apiKey" -H "Content-Type: application/json" "https://mms.mongodb.com/api/public/v1
{
  "emailAddress": "doh.jane@gmail.com",
  "lastName": "D'oh"
}

HTTP/1.1 200 OK

{
  "id" : "533dc19ce4b00835ff81e2eb",
  "username" : "jane.doe@mongodb.com",
  "emailAddress" : "doh.jane@gmail.com",
  "firstName" : "Jane",
  "lastName" : "D'oh",
  "roles" : [ {
    "groupId" : "533daa30879bb2da07807696",
    "roleName" : "GROUP_USER_ADMIN"
  } ],
  "links" : [ ... ]
}
```

Alerts

Sample Entity

```
{
  "id": "yyy",
  "groupId": "xxx",
  "typeName": "HOST_METRIC",
  "eventTypeName": "OUTSIDE_METRIC_THRESHOLD",
  "status": "OPEN",
  "acknowledgedUntil": "2014-03-01T12:00:00Z",
  "created": "2014-02-01T12:34:12Z",
  "updated": "2014-02-02T01:23:45Z",
  "resolved": null,
  "lastNotified": "2014-02-04T02:43:13Z",
  "currentValue": {
    "number": 123.45,
    "units": "MEGABYTES"
  },
  "links": [ ... ]
}
```

Name	Type	Description
id	string	Unique identifier.
groupId	string	ID of the group that this alert was opened for.
typeName	enum	The type of alert. Possible values are: `AGENT BACKUP HOST HOST_METRIC REPLICA_SET`
eventTypeName	enum	The name of the event that triggered the alert. The possible values here depend on the `typeName`: • **AGENT** - Possible values: `MONITORING_AGENT_DOWN BACKUP_AGENT_DOWN` • **HOST** - Possible values: `HOST_DOWN HOST_RECOVERING VERSION_BEHIND HOST_EXPOSED` • **HOST_METRIC** - Possible values: `OUTSIDE_METRIC_THRESH` • **BACKUP** - Possible values: `OPLOG_BEHIND RESYNC_REQUIRED`
status	enum	The current state of the alert. Possible values are: `OPEN CLOSED`
acknowledgedUntil	date	The date through which the alert has been acknowledged. Will not be present if the alert has never been acknowledged.
acknowledgementComment	string	The comment left by the user who acknowledged the alert. Will not be present if the alert has never been acknowledged.
acknowledgingUsername	string	The username of the user who acknowledged the alert. Will not be present if the alert has never been acknowledged.
created	date	When the alert was opened.
updated	date	When the alert was last updated.
resolved	date	When the alert was closed. Only present if the status is `CLOSED`.
lastNotified	date	When the last notification was sent for this alert. Only present if notifications have been sent.
metricName	enum	The name of the metric whose value went outside the threshold. Only present for alerts of type `HOST_METRIC`. Possible values are: `ASSERT_REGULAR ASSERT_WARNING ASSERT_MSG ASSERT_USER OPCOUNTER_CMD OPCOUNTER_QUERY OPCOUNTER_UPDATE OPCOUNTER_DELETE OPCOUNTER_INSERT OPCOUNTER_REPL_UPDATE OPCOUNTER_REPL_DELETE`

Entity Fields

Links

Relation	Description
self	Me
group	The group that this alert was triggered for.
alertConfigs	The alert configuration(s) that triggered this alert.
host	The host that triggered this alert. Only present for alerts of type `HOST`.

Operations

- `GET /api/public/v1.0/groups/GROUP-ID/alerts` - Gets all alerts with the specified status. Use the `status` query parameter with one of these possible values: `OPEN CLOSED`
- `GET /api/public/v1.0/groups/GROUP-ID/alerts/ALERT-ID` - Get a single alert by ID.
- `GET /api/public/v1.0/groups/GROUP-ID/alerts/ALERT-ID/alertConfigs` - Get the alert configuration(s) that triggered this alert.
- `PATCH /api/public/v1.0/groups/GROUP-ID/alerts/ALERT-ID` - Update an existing alert. The only field you may modify is the `acknowledgedUntil` field.
  - To acknowledge an alert "forever" set the date to 100 years in the future.
  - To unacknowledge a previously acknowledged alert, set the date in the past.

Examples Get an alert:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/5196d3628d022db4cbc26d9e/al

HTTP/1.1 200 OK

{
  "id" : "533cb4b8e4b0f1820cdabc7f",
  "groupId" : "5196d3628d022db4cbc26d9e",
  "typeName" : "BACKUP",
  "eventTypeName" : "OPLOG_BEHIND",
  "status" : "CLOSED",
  "created" : "2014-04-03T01:09:12Z",
  "updated" : "2014-04-03T01:14:12Z",
  "resolved" : "2014-04-03T01:14:12Z",
  "links" : [ ... ]
}
```

Get open alerts:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/5196d3628d022db4cbc26d9e/al

HTTP/1.1 200 OK

{
  "totalCount": 1,
  "results": [ {
    "id" : "533dc45ee4b00835ff81ec2a",
    "groupId" : "5196d3628d022db4cbc26d9e",
    "typeName" : "HOST_METRIC",
    "eventTypeName" : "OUTSIDE_METRIC_THRESHOLD",
    "status" : "OPEN",
    "created" : "2014-04-03T20:28:14Z",
    "updated" : "2014-04-03T20:28:14Z",
    "lastNotified" : "2014-04-03T20:28:23Z",
    "metricName": "ASSERTS_REGULAR",
```

```
    "currentValue" : {
      "number" : 0.0,
      "units" : "RAW"
    },
    "links" : [ ... ]
  } ],
  "links" : [ ... ]
}
```

Get alert configurations that triggered an alert:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/5196d3628d022db4cbc26d9e/al

HTTP/1.1 200 OK

{
  "totalCount": 3,
  "results": [ {
    "id" : "5271259ee4b00ece6b4754ef",
    "groupId" : "5196d3628d022db4cbc26d9e",
    "typeName" : "BACKUP",
    "eventTypeName" : "RESYNC_REQUIRED",
    "created" : "2013-10-30T15:28:30Z",
    "updated" : "2014-02-12T16:11:05Z",
    "enabled" : true,
    "matchers" : [ ],
    "notifications" : [ {
      "typeName" : "EMAIL",
      "intervalMin" : 60,
      "delayMin" : 0,
      "emailAddress" : "somebody@somewhere.com"
    } ],
    "links" : [ ... ]
  } ],
  "links" : [ ... ]
}
```

Acknowledge an alert:

```
curl -u "username:apiKey" -H "Content-Type: application/json" "https://mms.mongodb.com/api/public/v1
{
  "acknowledgedUntil": "2014-04-15T00:00:00-0400",
  "acknowledgementComment": "This is normal. Please ignore."
}

HTTP/1.1 200 OK

{
  "id" : "533dc45ee4b00835ff81ec2a",
  "groupId" : "5196d3628d022db4cbc26d9e",
  "typeName" : "HOST_METRIC",
  "eventTypeName" : "OUTSIDE_METRIC_THRESHOLD",
  "status" : "OPEN",
  "acknowledgedUntil" : "2014-04-15T04:00:00Z",
  "acknowledgementComment" : "This is normal. Please ignore.",
  "acknowledgingUsername" : "someuser@yourcompany.com",
  "created" : "2014-04-03T20:28:14Z",
  "updated" : "2014-04-03T20:33:14Z",
  "lastNotified" : "2014-04-03T20:33:23Z",
```

```
  "metricName": "ASSERTS_REGULAR",
  "currentValue" : {
    "number" : 0.0,
    "units" : "RAW"
  },
  "links" : [ ... ]
}
```

Alert Configurations

Sample Entity

```
{
  "id": "yyy",
  "groupId": "xxx",
  "typeName": "HOST_METRIC",
  "eventTypeName": "OUTSIDE_METRIC_THRESHOLD",
  "created": "2014-02-01T12:34:12Z",
  "updated": "2014-02-02T01:23:45Z",
  "enabled": true,
  "matchers": [{
    "fieldName": "HOSTNAME",
    "operator": "STARTS_WITH",
    "value": "my-host-prefix"
  }, {
    "fieldName": "PORT",
    "operator": "EQUALS",
    "value": "27017"
  }],
  "notifications": [{
    "typeName": "EMAIL",
    "intervalMin": 5,
    "delayMin": 0,
    "emailAddress": "somebody@somewhere.com"
  }, {
    "typeName": "HIP_CHAT",
    "intervalMin": 5,
    "delayMin": 0,
    "notificationToken": "123456abcdef",
    "roomName": "MMS Test Chat Room"
  }, {
    "typeName": "GROUP",
    "intervalMin": 5,
    "delayMin": 0,
    "groupId": "2847387cd717dabc348a",
    "groupName": "test1",
    "emailEnabled": true,
    "smsEnabled": true
  }, {
    "typeName": "USER",
    "intervalMin": 5,
    "delayMin": 0,
    "username": "john.doe",
    "emailEnabled": true,
    "smsEnabled": true
  }, {
    "typeName": "SMS",
```

```
    "intervalMin": 5,
    "delayMin": 0,
    "mobileNumber": "(212) 212-1212"
  }, {
    "typeName": "SNMP",
    "intervalMin": 5,
    "delayMin": 0,
    "snmpAddress": "somedomain.com:161"
  }, {
    "typeName": "PAGER_DUTY",
    "intervalMin": 5,
    "delayMin": 0,
    "serviceKey": "123456abcdef"
  }],
  "metricThreshold": {
    "metricName": "MEMORY_RESIDENT",
    "operator": "GREATER_THAN",
    "threshold": 7,
    "units": "GIGABYTES",
    "mode": "TOTAL"
  },
  "links": [ ... ]
}
```

Name	Type	Description
id	string	Unique identifier.
groupId	string	ID of the group that owns this alert configuration.
typeName	enum	The type of this alert configuration. Supports the same values as the `typeName` field of the **alerts** resource.
eventTypeName	enum	The type of event that will trigger an alert. Supports the same values as the `eventTypeName` field of the **alerts** resource.
created	date	When this alert configuration was created.
updated	date	When this alert configuration was last updated.
enabled	boolean	Is this alert configuration enabled?
matchers	object array	Rules to apply when matching an object against this alert configuration. Only entities that match *all* these rules will be checked for an alert condition.

Continued on next page

Table 2 – continued from previous page

Name	Type	Description
matchers.fieldName	string	The name of the field in the target object to match on. The available fields depend on the `typeName`: • **AGENT** - Not applicable. • **BACKUP** - Not applicable. • **HOST** and **HOST_METRIC** - Possible values are: `HOSTNAME PORT HOSTNAME_AND_PORT REPLICA_SET_NAME TYPE_NAME`. • **REPLICA_SET** - Possible values are: `REPLICA_SET_NAME SHARD_NAME CLUSTER_NAME`
matchers.operator	enum	The operator to test the field's value. Possible values are: `EQUALS NOT_EQUALS CONTAINS NOT_CONTAINS STARTS_WITH ENDS_WITH REGEX`
matchers.value	string	The value to test with the specified operator. When matching on the `TYPE_NAME` field for a `HOST` or `HOST_METRIC` alert, the possible typeName values are: `PRIMARY SECONDARY STANDALONE CONFIG MONGOS`
notifications	object array	Notifications to send when an alert condition is detected.
notifications.typeName	enum	The type of alert notification. Possible values are: `GROUP USER SMS EMAIL PAGER_DUTY HIPCHAT SNMP`. Note that `SNMP` notifications are not available in the cloud version of MMS. This feature is only available to On Premise installations.
notifications.delayMin	integer	The number of minutes to wait after an alert condition is detected before sending out the first notification.
notifications.intervalMin	integer	The number of minutes to wait between succcessive notifications for unacknowledged alerts that are not resolved.
notifications.emailAddress	string	The email address to which to send notification. Only present for notifications of type `EMAIL`.
notifications.notificationToken	string	A HipChat API token. Only present for notifications of type `HIP_CHAT`.

Continued on next page

Table 2 – continued from previous page

Name	Type	Description
notifications.roomName	string	HipChat room name. Only present for notifications of type `HIP_CHAT`.
notifications.emailEnabled	boolean	Should email notifications be sent? Only present for notifications of type `GROUP` and `USER`.
notifications.smsEnabled	boolean	Should SMS notifications be sent? Only present for notifications of type `GROUP` and `USER`.
notifications.username	string	The name of an MMS user to which to send notifications. Only a user in the group that owns the alert configuration is allowed here.
notifications.mobileNumber	string	Mobile number to send SMS messages to. Only present for notifications of type `SMS`.
notifications.snmpAddress	string	Hostname and port to send SNMP traps to. Note that SNMP is only supported for On Premise MMS; also, at this time MMS is only able to send SNMP traps to the standard SNMP port (161).
notifications.serviceKey	string	PagerDuty service key.
metricThreshold	object	The threshold that will cause an alert to be triggered. Only present for alerts of the `HOST_METRIC`.
metricThreshold.metricName	enum	The name of the metric to check. Supports the same values as the `metricName` field of the **alerts** resource.
metricThreshold.operator	enum	The operator to apply when checking the current metric value against the threshold value. Possible values are: `GREATER_THAN LESS_THAN`
metricThreshold.threshold	integer	The threshold value outside of which an alert will be triggered.
metricThreshold.units	enum	The units for the threshold value. Supports the same values as the `currentValue.units` field of the **alerts** resource.
metricThreshold.mode	enum	The mode to use when computing the current metric value. Possible values are: `AVERAGE TOTAL`

Entity Fields

Links

Relation	Description
self	Me
group	The group that owns this alert configuration.
alerts	Open alerts triggered by this alert configuration.

Operations

- `GET /api/public/v1.0/groups/GROUP-ID/xxx/alertConfigs/ALERT-CONFIG-ID` - Get a single alert configuration by ID.
- `GET /api/public/v1.0/groups/GROUP-ID/xxx/alertConfigs` - Get all alert configurations for a group.
- `GET /api/public/v1.0/groups/GROUP-ID/xxx/alertConfigs/ALERT-CONFIG-ID/alerts` - Get all open alerts that were triggered by an alert configuration.
- `POST /api/public/v1.0/groups/GROUP-ID/xxx/alertConfigs` - Create a new alert configuration. All fields are required except `created` and `updated`.
- `PUT /api/public/v1.0/groups/GROUP-ID/xxx/alertConfigs` - Update an existing alert configuration. Partial updates are not supported except for one field (see `PATCH` below), so you must send the entire entity.
- `PATCH /api/public/v1.0/groups/GROUP-ID/xxx/alertConfigs/ALERT-CONFIG-ID` - Use to enable/disable an alert configuration by setting the `enabled` field.
- `DELETE /api/public/v1.0/groups/GROUP-ID/xxx/alertConfigs/ALERT-CONFIG-ID` - Remove an alert configuration.

Examples Get all alert configurations in a group:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/5196d3628d022db4cbc26d9e/a

HTTP/1.1 200 OK

{
  "totalCount": 3,
  "results": [ {
    "id" : "5271259ee4b00ece6b4754ef",
    "groupId" : "5196d3628d022db4cbc26d9e",
    "typeName" : "BACKUP",
    "eventTypeName" : "RESYNC_REQUIRED",
    "created" : "2013-10-30T15:28:30Z",
    "updated" : "2014-02-12T16:11:05Z",
    "enabled" : true,
    "matchers" : [ ],
    "notifications" : [ {
      "typeName" : "EMAIL",
      "intervalMin" : 60,
      "delayMin" : 0,
      "emailAddress" : "somebody@somewhere.com"
    } ],
    "links" : [ ... ]
  }, {
    "id" : "5329c8dfe4b0b07a83d67e7e",
    "groupId" : "5196d3628d022db4cbc26d9e",
    "typeName" : "AGENT",
    "eventTypeName" : "MONITORING_AGENT_DOWN",
    "created" : "2014-03-19T16:42:07Z",
    "updated" : "2014-03-19T16:42:07Z",
    "enabled" : true,
    "matchers" : [ ],
    "notifications" : [ {
      "typeName" : "GROUP",
      "intervalMin" : 5,
```

```
      "delayMin" : 0,
      "emailEnabled" : true,
      "smsEnabled" : false
    } ],
    "links" : [ ... ]
  }, {
    "id" : "533dc40ae4b00835ff81eaee",
    "groupId" : "5196d3628d022db4cbc26d9e",
    "typeName" : "HOST_METRIC",
    "eventTypeName" : "OUTSIDE_METRIC_THRESHOLD",
    "created" : "2014-04-03T20:26:50Z",
    "updated" : "2014-04-03T20:26:50Z",
    "enabled" : true,
    "matchers" : [ {
      "field" : "hostnameAndPort",
      "operator" : "EQUALS",
      "value" : "mongo.babypearfoo.com:27017"
    } ],
    "notifications" : [ {
      "typeName" : "SMS",
      "intervalMin" : 5,
      "delayMin" : 0,
      "mobileNumber" : "2343454567"
    } ],
    "metricThreshold" : {
      "metricName" : "ASSERT_REGULAR",
      "operator" : "LESS_THAN",
      "threshold" : 99.0,
      "units" : "RAW",
      "mode" : "AVERAGE"
    },
    "links" : [ ... ]
  } ]
}
```

Get an alert configuration:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/5196d3628d022db4cbc26d9e/al

HTTP/1.1 200 OK

{
  "id" : "533dc40ae4b00835ff81eaee",
  "groupId" : "5196d3628d022db4cbc26d9e",
  "typeName" : "HOST_METRIC",
  "eventTypeName" : "OUTSIDE_METRIC_THRESHOLD",
  "created" : "2014-04-03T20:26:50Z",
  "updated" : "2014-04-03T20:26:50Z",
  "enabled" : true,
  "matchers" : [ {
    "field" : "hostnameAndPort",
    "operator" : "EQUALS",
    "value" : "mongo.babypearfoo.com:27017"
  } ],
  "notifications" : [ {
    "typeName" : "SMS",
    "intervalMin" : 5,
    "delayMin" : 0,
    "mobileNumber" : "2343454567"
```

```
  } ],
  "metricThreshold" : {
    "metricName" : "ASSERT_REGULAR",
    "operator" : "LESS_THAN",
    "threshold" : 99.0,
    "units" : "RAW",
    "mode" : "AVERAGE"
  },
  "links" : [ ... ]
}
```

Get all open alerts triggered by an alert configuration:

```
curl -u "username:apiKey" "https://mms.mongodb.com/api/public/v1.0/groups/5196d3628d022db4cbc26d9e/al

HTTP/1.1 200 OK

{
  "totalCount" : 2,
  "results" : [ {
    "id" : "53569159300495c7702ee3a3",
    "groupId" : "4d1b6314e528c81a1f200e03",
    "typeName" : "HOST_METRIC",
    "eventTypeName" : "OUTSIDE_METRIC_THRESHOLD",
    "status" : "OPEN",
    "acknowledgedUntil" : "2014-05-01T14:00:00Z",
    "created" : "2014-04-22T15:57:13.562Z",
    "updated" : "2014-04-22T20:14:11.388Z",
    "lastNotified" : "2014-04-22T15:57:24.126Z",
    "metricName" : "ASSERT_REGULAR",
    "currentValue" : {
      "number" : 0.0,
      "units" : "RAW"
    },
    "links" : [ ... ]
  }, {
    "id" : "5356ca0e300495c770333340",
    "groupId" : "4d1b6314e528c81a1f200e03",
    "typeName" : "HOST_METRIC",
    "eventTypeName" : "OUTSIDE_METRIC_THRESHOLD",
    "status" : "OPEN",
    "created" : "2014-04-22T19:59:10.657Z",
    "updated" : "2014-04-22T20:14:11.388Z",
    "lastNotified" : "2014-04-22T20:14:19.313Z",
    "metricName" : "ASSERT_REGULAR",
    "currentValue" : {
      "number" : 0.0,
      "units" : "RAW"
    },
    "links" : [ ... ]
  } ],
  "links" : [ ... ]
}
```

Create a new alert configuration:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/4d1b6314e528c81
{
  "groupId" : "4d1b6314e528c81a1f200e03",
```

```
  "typeName" : "REPLICA_SET",
  "eventTypeName" : "RESYNC_REQUIRED",
  "enabled" : true,
  "notifications" : [ {
    "typeName" : "GROUP",
    "intervalMin" : 5,
    "delayMin" : 0,
    "smsEnabled" : false,
    "emailEnabled" : true
  } ]
}

HTTP/1.1 201 Created
Location: https://mms.mongodb.com/api/public/v1.0/groups/4d1b6314e528c81a1f200e03/alertConfigs/5357ce

{
  "id" : "5357ce3e3004d83bd9c7864c",
  "groupId" : "4d1b6314e528c81a1f200e03",
  "typeName" : "REPLICA_SET",
  "created" : "2014-04-23T14:29:18Z",
  "updated" : "2014-04-23T14:29:18Z",
  "enabled" : true,
  "matchers" : [ ],
  "notifications" : [ {
    "typeName" : "GROUP",
    "intervalMin" : 5,
    "delayMin" : 0,
    "emailEnabled" : true,
    "smsEnabled" : false
  } ],
  "links" : [ ... ]
}
```

Update an existing alert configuration:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/4d1b6314e528c81
{
  "groupId" : "4d1b6314e528c81a1f200e03",
  "typeName" : "REPLICA_SET",
  "eventTypeName" : "RESYNC_REQUIRED",
  "enabled" : true,
  "matchers" : [ {
    "fieldName" : "REPLICA_SET_NAME",
    "operator" : "EQUALS",
    "value" : "rs1"
  } ],
  "notifications" : [ {
    "typeName" : "EMAIL",
    "emailAddress" : "sos@foo.com",
    "intervalMin" : 60,
    "delayMin" : 5
  }, {
    "typeName" : "GROUP",
    "intervalMin" : 120,
    "delayMin" : 60,
    "smsEnabled" : true,
    "emailEnabled" : false
  } ]
}
```

```
HTTP/1.1 200 OK

{
  "id" : "5357ce3e3004d83bd9c7864c",
  "groupId" : "4d1b6314e528c81a1f200e03",
  "typeName" : "REPLICA_SET",
  "created" : "2014-04-23T14:52:29Z",
  "updated" : "2014-04-23T14:52:29Z",
  "enabled" : true,
  "matchers" : [ {
    "fieldName" : "REPLICA_SET_NAME",
    "operator" : "EQUALS",
    "value" : "rs1"
  } ],
  "notifications" : [ {
    "typeName" : "EMAIL",
    "intervalMin" : 60,
    "delayMin" : 5,
    "emailAddress" : "sos@foo.com"
  }, {
    "typeName" : "GROUP",
    "intervalMin" : 120,
    "delayMin" : 60,
    "emailEnabled" : false,
    "smsEnabled" : true
  } ],
  "links" : [ ... ]
}
```

Disable an alert configuration:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/4d1b6314e528c81
{
  "enabled" : false
}

HTTP/1.1 200 OK

{
  "id" : "5357ce3e3004d83bd9c7864c",
  "groupId" : "4d1b6314e528c81a1f200e03",
  "typeName" : "REPLICA_SET",
  "created" : "2014-04-23T14:52:29Z",
  "updated" : "2014-04-23T14:56:25Z",
  "enabled" : false,
  "matchers" : [ {
    "fieldName" : "REPLICA_SET_NAME",
    "operator" : "EQUALS",
    "value" : "rs1"
  } ],
  "notifications" : [ {
    "typeName" : "EMAIL",
    "intervalMin" : 60,
    "delayMin" : 5,
    "emailAddress" : "sos@foo.com"
  }, {
    "typeName" : "GROUP",
    "intervalMin" : 120,
```

```
    "delayMin" : 60,
    "emailEnabled" : false,
    "smsEnabled" : true
  } ],
  "links" : [ ... ]
}
```

Delete an alert configuration:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/4d1b6314e528c81

HTTP/1.1 200 OK
```

Backup Configurations

The resource modification operation `PATCH` only accepts requests from whitelisted IP addressses. You can only modify a backup configuration if the request originates from an IP address on the API user's whitelist.

Sample Entity

```
{
  "groupId": "xxx",
  "clusterId": "yyy",
  "statusName": "STARTED",
  "authMechanismName": "MONGODB_CR",
  "username": "johnny5",
  "password": "guess!",
  "sslEnabled": false,
  "syncSource": "PRIMARY",
  "provisioned": true,
  "excludedNamespaces:" [ "a", "b", "c.d" ]
}
```

Entity Fields

Name	Type	Description
groupId	string	ID of the group that owns this backup configuration.
clusterId	string	ID of the cluster that this backup configuration is for.
statusName	enum	The current (or desired) status of the backup configuration. Possible values are: `INACTIVE PROVISIONING STARTED STOPPED TERMINATING`
authMechanism-Name	enum	The name of the authentication mechanism to use when connecting to the sync cource database. Only present when using authentication. Possible values are: `MONGODB_CR GSSAPI`
username	string	The username to use to connect to the sync source database. Only present when backing up `mongod` instances that require clients to authenticate.
password	string	The password to use to connect to the sync source database. Only present when backup the `mongod` instances that require clients to authenticate. You may only send this field to MMS when updating backup configuration. `GET` request do *not* include this field.
sslEnabled	boolean	Is SSL enabled for the sync source database?
syncSource	string	The `mongod` instance to get backup data from. Possible values are either a specific hostname or one of: `PRIMARY` and `SECONDARY`. This field is only used when updating a backup configuration. It is not returned by a `GET` request.
excluded-Names-paces	string array	A list of database names and/or collection names that to omit from the back up. If a string has a dot (e.g. `.`), then it is a fully qualified namespace in the form of `<database>.<collection>`, otherwise strings are database names.

Additionally, On Prem versions of MMS return the following additional field:

Name	Type	Description
provisioned	boolean	Reports if MMS has provisioned the resources needed to store a backup This field is only present when the amount of data to be backed up exceeds a certain threshold.

Links

Relation	Description
self	Me
cluster	The cluster that this backup configuration is for.
group	The group that owns this backup configuration.
snapshotSchedule	The snapshot schedule for this backup configuration.

Operations

- `GET /api/public/v1.0/groups/GROUP-ID/backupConfigs/CLUSTER-ID` - Get a single backup configuration by cluster ID. `CLUSTER-ID` must be the ID of either a replica set or a sharded cluster.
- `GET /api/public/v1.0/groups/GROUP-ID/backupConfigs` - Get all backup configurations for a group.
- `PATCH /api/public/v1.0/groups/GROUP-ID/backupConfigs/CLUSTER-ID` - Request a state change to an existing backup configuration. Note that these changes are generally asynchronous and will result in a response status code of 202 (Accepted). Additionally, if you issue a `GET` request for a backup configuration after a successful `PATCH`, the returned entity may not immediately reflect the update given the asynchronous nature of these state transitions.

  When modifying the `statusName` property, these are the acceptable transitions:

  - `STARTED` - Only valid if the current status is `STOPPED` or `INACTIVE`.
  - `STOPPED` - Only valid if the current status is `STARTED`.
  - `TERMINATING` - Only valid if the current status is `STOPPED`.
  - You cannot change the `statusName` to these values: `INACTIVE PROVISIONING`.

Examples Get a single backup configuration:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/5196d3628d022d

HTTP/1.1 200 OK

{
  "groupId" : "5196d3628d022db4cbc26d9e",
  "clusterId" : "5196e5b0e4b0fca9cc88334a",
  "statusName" : "STARTED",
  "sslEnabled" : false,
  "excludedNamespaces" : [ ],
  "links" : [ ... ]
}
```

Get all backup configurations for a group:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/5196d3628d022d

HTTP/1.1 200 OK

{
  "totalCount" : 3,
```

```
  "results" : [ {
    "groupId" : "5196d3628d022db4cbc26d9e",
    "clusterId" : "5196e5b0e4b0fca9cc88334a",
    "statusName" : "STARTED",
    "sslEnabled" : false,
    "excludedNamespaces" : [ ],
    "links" : [ ... ]
  }, {
    "groupId" : "5196d3628d022db4cbc26d9e",
    "clusterId" : "51a2ac88e4b0371c2dbf46ea",
    "statusName" : "STARTED",
    "sslEnabled" : false,
    "excludedNamespaces" : [ ],
    "links" : [ ... ]
  }, {
    "groupId" : "5196d3628d022db4cbc26d9e",
    "clusterId" : "52d33abee4b0ca49bc6acd6c",
    "statusName" : "STOPPED",
    "sslEnabled" : false,
    "excludedNamespaces" : [ ],
    "links" : [ ... ]
  } ],
  "links" : [ ... ]
}
```

Update a backup configuration

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/5196d3628d022db
{
  "statusName": "STOPPED"
}

HTTP/1.1 202 Accepted

{
  "groupId" : "5196d3628d022db4cbc26d9e",
  "clusterId" : "5196e5b0e4b0fca9cc88334a",
  "statusName" : "STOPPED",
  "sslEnabled" : false,
  "excludedNamespaces" : [ ],
  "links" : [ ... ]
}
```

Snapshot Schedule

This resource allows you to view and configure various properties of snapshot creation and retention for a replica set or cluster. In order to modify this resource, the request must originate from an IP address on the API user's whitelist.

Sample Entity

```
{
  "groupId": "xxx",
  "clusterId": "yyy",
  "snapshotIntervalHours": 6,
  "snapshotRetentionDays": 3,
  "clusterCheckpointIntervalMin": 15,
  "dailySnapshotRetentionDays": 14,
```

```
  "weeklySnapshotRetentionWeeks": 6,
  "monthlySnapshotRetentionMonths": 12
}
```

Entity Fields

Name	Type	Description
groupId	string	ID of the group that owns the backup configuration.
clusterId	string	ID of the cluster to which this backup configuration applies.
snapshotIntervalHours	integer	Number of hours between snapshots. Supported values are 6, 8, 12, and 24.
snapshotRetentionDays	integer	Number of days to keep recent snapshots. Supported values are 1 - 5.
clusterCheckpointIntervalMin	integer	Number of minutes between successive cluster checkpoints. This only applies to sharded clusters. This number determines the granularity of point-in-time restores for sharded clusters.
dailySnapshotRetentionDays	integer	Number of days to retain daily snapshots. Supported values are 1 - 365.
weeklySnapshotRetentionWeeks	integer	Number of weeks to retain weekly snapshots. Supported values are 1 - 52.
monthlySnapshotRetentionMonths	integer	Number of months to retain monthly snapshots. Supported values are 1 - 36.

Links

Relation	Description
self	Me
cluster	The cluster that this backup configuration is for.
group	The group that owns this backup configuration.
backupConfig	The backup configuration that this schedule belongs to.

Operations

- `GET /api/public/v1.0/groups/GROUP-ID/backupConfigs/CLUSTER-ID/snapshotSchedule` - Get the snapshot schedule for a cluster. `CLUSTER-ID` must be the ID of either a replica set or a sharded cluster.
- `PATCH /api/public/v1.0/groups/GROUP-ID/backupConfigs/CLUSTER-ID/snapshotSchedule` - Change the parameters of snapshot creation and retention. Any combination of the snapshot schedule's attributes can be modified.

Examples Get a snapshot schedule:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/525ec8394f5e625

HTTP/1.1 200 OK

{
  "groupId" : "525ec8394f5e625c80c7404a",
  "clusterId" : "53bc556ce4b049c88baec825",
  "snapshotIntervalHours" : 6,
  "snapshotRetentionDays" : 2,
```

```
  "dailySnapshotRetentionDays" : 7,
  "weeklySnapshotRetentionWeeks" : 4,
  "monthlySnapshotRetentionMonths" : 13,
  "links": [ ... ]
}
```

Update a snapshot schedule:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/525ec8394f5e625
{
  "snapshotIntervalHours": 8,
  "dailySnapshotRetentionDays": 14,
  "monthlySnapshotRetentionMonths": 6
}

HTTP/1.1 200 OK

{
  "groupId" : "525ec8394f5e625c80c7404a",
  "clusterId" : "53bc556ce4b049c88baec825",
  "snapshotIntervalHours" : 8,
  "snapshotRetentionDays" : 2,
  "dailySnapshotRetentionDays" : 14,
  "weeklySnapshotRetentionWeeks" : 4,
  "monthlySnapshotRetentionMonths" : 6,
  "links": [ ... ]
}
```

Snapshots

This resource allows you to view snapshot metadata and remove existing snapshots. A snapshot is a complete copy of the data in a `mongod` instance at a point in time. In order to delete a resource, the request must originate from an IP address on the API user's whitelist.

Note that this resource is only meant to provide snapshot metadata. In order to retrieve the snapshot data (in order to perform a restore, for example), you must create a Restore Job.

Sample Entity

```
{
  "id": "5875f665da588548965b",
  "groupId": "2847387cd717dabc348a",
  "clusterId": "348938fbdbca74718cba",
  "created": {
    "date": "2014-02-01T12:34:12Z",
    "increment": 54
  },
  "expires": "2014-08-01T12:34:12Z",
  "complete": true,
  "parts": [
    {
      "typeName": "REPLICA_SET",
      "clusterId": "2383294dbcafa82928ad",
      "replicaSetName": "rs0",
      "mongodVersion": "2.4.8",
      "dataSizeBytes": 489283492,
      "storageSizeBytes": 489746352,
```

```
      "fileSizeBytes": 518263456
    }, {
      "typeName": "REPLICA_SET",
      "clusterId": "2383294dbcafa82928b3",
      "replicaSetName": "rs1",
      "mongodVersion": "2.4.8",
      "dataSizeBytes": 489283492,
      "storageSizeBytes": 489746352,
      "fileSizeBytes": 518263456
    }, {
      "typeName": "CONFIG_SERVER",
      "mongodVersion": "2.4.6",
      "dataSizeBytes": 48928,
      "storageSizeBytes": 48974,
      "fileSizeBytes": 51826
    }
  ]
}
```


Entity Fields

Name	Type	Description
groupId	string	ID of the group that owns the snapshot.
clusterId	string	ID of the cluster represented by the snapshot.
created	BSON timestamp	The exact point-in-time at which the snapshot was taken.
expires	timestamp	The date after which this snapshot is eligible for deletion.
complete	boolean	Is this snapshot complete? This will be false if the snapshot creation job is still in progress.
parts	array of parts	The individual parts that comprise the complete snapshot. For a replica set, this array will contain a single element. For a sharded cluster, there will be one element for each shard plus one element for the config server.
parts.typeName	enum	The type of server represented by the part. Possible values are: `REPLICA_SET` `CONFIG_SERVER`
parts.clusterId	string	ID of the replica set. Not present for a config server.
parts.replicaSetName	string	Name of the replica set. Not present for a config server.
parts.mongodVersion	string	The version of `mongod` that was running when the snapshot was created.
parts.dataSizeBytes	integer	The total size of the data in the snapshot.
parts.storageSizeBytes	integer	The total size of space allocated for document storage.
parts.fileSizeBytes	integer	The total size of the data files.

Links

Relation	Description
self	Me
cluster	The cluster that this snapshot belongs to.
group	The group that owns this snapshot.

Operations

- `GET /api/public/v1.0/groups/GROUP-ID/clusters/CLUSTER-ID/snapshots` - Get all snapshots for a cluster. `CLUSTER-ID` must be the ID of either a replica set or a sharded cluster.
- `GET /api/public/v1.0/groups/GROUP-ID/clusters/CLUSTER-ID/snapshots/SNAPSHOT-ID` - Get a single snapshot.

- `DELETE /api/public/v1.0/groups/GROUP-ID/clusters/CLUSTER-ID/snapshots/SNAPSHOT-ID` - Remove a single snapshot. Note that while the two above methods return metadata about the snapshot, this will actually remove the underlying backed-up data.

Examples Get all snapshots:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/525ec8394f5e625

HTTP/1.1 200 OK

{
  "totalCount" : 3,
  "results" : [ {
    "id" : "53bd5fb5e4b0774946a16fad",
    "groupId" : "525ec8394f5e625c80c7404a",
    "clusterId" : "53bc556ce4b049c88baec825",
    "created" : {
      "date" : "2014-07-09T15:24:37Z",
      "increment" : 1
    },
    "expires" : "2014-07-11T15:24:37Z",
    "complete" : true,
    "parts" : [ {
      "typeName" : "REPLICA_SET",
      "clusterId" : "53bc556ce4b049c88baec825",
      "replicaSetName" : "rs0",
      "mongodVersion" : "2.6.3",
      "dataSizeBytes" : 17344,
      "storageSizeBytes" : 10502144,
      "fileSizeBytes" : 67108864
    } ],
    "links" : [ ... ]
  }, {
    ...
  } ],
  "links": [ ... ]
}
```

Get one snapshot:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/525ec8394f5e625

HTTP/1.1 200 OK

{
  "id" : "53bd5fb5e4b0774946a16fad",
  "groupId" : "525ec8394f5e625c80c7404a",
  "clusterId" : "53bc556ce4b049c88baec825",
  "created" : {
    "date" : "2014-07-09T15:24:37Z",
    "increment" : 1
  },
  "expires" : "2014-07-11T15:24:37Z",
  "complete" : true,
  "parts" : [ {
    "typeName" : "REPLICA_SET",
    "clusterId" : "53bc556ce4b049c88baec825",
    "replicaSetName" : "rs0",
```

```
    "mongodVersion" : "2.6.3",
    "dataSizeBytes" : 17344,
    "storageSizeBytes" : 10502144,
    "fileSizeBytes" : 67108864
  } ],
  "links" : [ ... ]
}
```

Remove a snapshot:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/525ec8394f5e625

HTTP/1.1 200 OK
```

Restore Jobs

This resource allows you to manage restore jobs. A restore job is essentially a request to retrieve one of your existing snapshots, or a snapshot for a recent specific point-in-time, in order to restore a `mongod` to a previous state. In order to initiate a restore job, the request must originate from an IP address on the API user's whitelist.

Sample Entity

```
{
  "id" : "53bd7f13e4b0a7304e226998",
  "groupId" : "525ec8394f5e625c80c7404a",
  "clusterId" : "53bc556ce4b049c88baec825",
  "snapshotId" : "53bd439ae4b0774946a16490",
  "created" : "2014-07-09T17:42:43Z",
  "timestamp" : {
    "date" : "2014-07-09T09:24:37Z",
    "increment" : 1
  },
  "statusName" : "FINISHED",
  "pointInTime" : false,
  "delivery" : {
    "methodName" : "HTTP",
    "url" : "https://api-backup.mongodb.com/backup/restore/v2/pull/ae6bc7a8bfdd5a99a0c118c73845dc75/5
    "expires" : "2014-07-09T18:42:43Z",
    "statusName" : "READY"
  },
  "links" : [ ... ]
}
```

Entity Fields

Name	Type	Description
groupId	string	ID of the group that owns the restore job.
clusterId	string	ID of the cluster represented by the restore job.
snapshotId	string	ID of the snapshot to restore.
batchId	string	ID of the batch to which this restore job belongs. Only present for a restore of a sharded cluster.
created	timestamp	When the restore job was requested.
timestamp	BSON timestamp	Timestamp of the latest oplog entry in the restored snapshot.
statusName	enum	Current status of the job. Possible values are: `FINISHED IN_PROGRESS BROKEN KILLED`
pointInTime	boolean	Is this job for a point-in-time restore?
delivery	object	Additional details about how the restored snapshot data will be delivered.
delivery.methodName	enum	How the data will be delivered. Possible values are: `HTTP`
delivery.url	string	The URL from which the restored snapshot data can be downloaded. Only present if `methodName` is `HTTP`.
delivery.expires	timestamp	Date after which the URL will no longer be available. Only present if `methodName` is `HTTP`.
delivery.statusName	enum	Current status of the downloadable file. Possible values are: `READY EXPIRED MAX_DOWNLOADS_EXCEEDED`. Only present if `methodName` is `HTTP`.

Links

Relation	Description
self	Me
cluster	The cluster that is to restore.
snapshot	The snapshot that is to restore.
group	The group that owns the cluster.

Operations

- `GET /api/public/v1.0/groups/GROUP-ID/clusters/CLUSTER-ID/restoreJobs` - Get all restore jobs for a cluster. `CLUSTER-ID` must be the ID of either a replica set or a sharded cluster.
- `GET /api/public/v1.0/groups/GROUP-ID/clusters/CLUSTER-ID/restoreJobs?batchId=BATCH-ID` - Get all restore jobs in the specified batch. When creating a restore job for a sharded cluster, MMS creates a separate job for each shard, plus another for the config server. Each of those jobs will be part of a batch. A restore job for a replica set, however, will not be part of a batch.
- `GET /api/public/v1.0/groups/GROUP-ID/clusters/CLUSTER-ID/restoreJobs/JOB-ID` - Get a single restore job.
- `POST /api/public/v1.0/groups/GROUP-ID/clusters/CLUSTER-ID/restoreJobs` - Create a restore job for the specified `CLUSTER-ID`. You can create a restore job for either an existing snapshot or for a specific recent point-in-time. (The recency depends on the size of your "point-in-time window.") See below for examples of each. The response body includes an array of restore jobs. When requesting a restore of a replica set, the array will contain a single element. For a sharded cluster, the array will contain one element for each shard, plus one for the config server. Each element will also include the `batchId` representing the batch to which the jobs belong.

Examples Get all restore jobs:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/525ec8394f5e625

HTTP/1.1 200 OK

{
  "totalCount" : 2,
  "results" : [ {
    "id" : "53bd7f38e4b0a7304e226b3f",
    "groupId" : "525ec8394f5e625c80c7404a",
    "clusterId" : "53bc556ce4b049c88baec825",
    "snapshotId" : "53bd4356e4b0774946a16455",
    "created" : "2014-07-09T17:43:20Z",
    "timestamp" : {
      "date" : "2014-07-08T21:24:37Z",
      "increment" : 1
    },
    "statusName" : "FINISHED",
    "pointInTime" : false,
    "delivery" : {
      "methodName" : "HTTP",
      "url" : "https://api-backup.mongodb.com/backup/restore/v2/pull/ae6bc7a8bfdd5a99a0c118c73845dc75
      "expires" : "2014-07-09T18:43:21Z",
      "statusName" : "READY"
    },
    "links" : [ ... ]
  }, {
    "id" : "53bd7f13e4b0a7304e226998",
    "groupId" : "525ec8394f5e625c80c7404a",
    "clusterId" : "53bc556ce4b049c88baec825",
    "snapshotId" : "53bd439ae4b0774946a16490",
    "created" : "2014-07-09T17:42:43Z",
    "timestamp" : {
      "date" : "2014-07-09T09:24:37Z",
      "increment" : 1
    },
    "statusName" : "FINISHED",
    "pointInTime" : false,
    "delivery" : {
      "methodName" : "HTTP",
      "url" : "https://api-backup.mongodb.com/backup/restore/v2/pull/ae6bc7a8bfdd5a99a0c118c73845dc75
      "expires" : "2014-07-09T18:42:43Z",
      "statusName" : "READY"
    },
    "links" : [ ... ]
  } ],
  "links": [ ... ]
}
```

Get a single restore job:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/525ec8394f5e625

HTTP/1.1 200 OK

{
  "id" : "53bd7f13e4b0a7304e226998",
  "groupId" : "525ec8394f5e625c80c7404a",
  "clusterId" : "53bc556ce4b049c88baec825",
  "snapshotId" : "53bd439ae4b0774946a16490",
```

```
  "created" : "2014-07-09T17:42:43Z",
  "timestamp" : {
    "date" : "2014-07-09T09:24:37Z",
    "increment" : 1
  },
  "statusName" : "FINISHED",
  "pointInTime" : false,
  "delivery" : {
    "methodName" : "HTTP",
    "url" : "https://api-backup.mongodb.com/backup/restore/v2/pull/ae6bc7a8bfdd5a99a0c118c73845dc75/5
    "expires" : "2014-07-09T18:42:43Z",
    "statusName" : "READY"
  },
  "links" : [ ... ]
}
```

Create a restore job for an existing snapshot:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/525ec8394f5e625
{
  "snapshotId": "53bd439ae4b0774946a16490"
}

HTTP/1.1 200 OK

{
  "totalCount" : 1,
  "results" : [ {
    "id" : "53bd9f9be4b0a7304e23a8c6",
    "groupId" : "525ec8394f5e625c80c7404a",
    "clusterId" : "53bc556ce4b049c88baec825",
    "snapshotId" : "53bd439ae4b0774946a16490",
    "created" : "2014-07-09T20:01:31Z",
    "timestamp" : {
      "date" : "2014-07-09T09:24:37Z",
      "increment" : 1
    },
    "statusName" : "IN_PROGRESS",
    "pointInTime" : false,
    "links" : [ ... ]
  } ]
}
```

Create a point-in-time restore job:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/groups/525ec8394f5e625
{
  "timestamp": {
    "date": "2014-07-09T09:20:00Z",
    "increment": 0
  }
}

HTTP/1.1 200 OK

{
  "totalCount" : 1,
  "results" : [ {
    "id" : "53bda0dfe4b0a7304e23b54a",
```

```
    "groupId" : "525ec8394f5e625c80c7404a",
    "clusterId" : "53bc556ce4b049c88baec825",
    "created" : "2014-07-09T20:06:55Z",
    "timestamp" : {
      "date" : "2014-07-09T09:20:00Z",
      "increment" : 0
    },
    "statusName" : "IN_PROGRESS",
    "pointInTime" : true,
    "links" : [ ... ]
  } ]
}
```

Whitelist

The resource modification operations `POST` and `DELETE` are whitelisted. For example, you can only add an IP address to a whitelist if the request originates from an IP address on the existing whitelist.

Sample Entity

```
{
  "ipAddress": "1.2.3.4",
  "created": "2014-01-02T12:34:56Z",
  "lastUsed": "2014-03-12T02:03:04Z",
  "count": 1234
}
```

Entity Fields

Name	Type	Description
ipAddress	string	A whitelisted IP address.
created	date	The date this IP address was added to the whitelist.
lastUsed	date	The date of the most recent request that originated from this IP address. Note that this field is *only* updated when a resource that is protected by the whitelist is accessed.
count	integer	The total number of requests that originated from this IP address. Note that this field is *only* updated when a resource that is protected by the whitelist is accessed.

Links

Relation	Description
self	Me
user	The user that owns this whitelist.

Operations

- `GET /api/public/v1.0/users/USER-ID/whitelist` - Gets the whitelist for the specified user. You can only access your own whitelist, so the `USER-ID` in the URL *must* match the ID of the user associated with the API Key.
- `GET /api/public/v1.0/users/USER-ID/whitelist/IP-ADDRESS` - Gets the whitelist entry for a single IP address.
- `POST /api/public/v1.0/users/USER-ID/whitelist` - Add one or more IP addresses to the user's whitelist.

  - The entity body must be an array of whitelist entities, even if there is only one. The only field you need to specify for each entity is the `ipAddress`.
  - If an IP address is already in the whitelist, it will be ignored.
- `DELETE /api/public/v1.0/users/USER-ID/whitelist/IP-ADDRESS` - Remove an IP address from the whitelist.
  - You cannot remove your current IP address from the whitelist.

Examples Get a user's whilelist:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/users/5356823b3004dee3

HTTP/1.1 200 OK

{
  "totalCount" : 1,
  "results" : [ {
    "ipAddress" : "12.34.56.78",
    "created" : "2014-04-23T16:17:44Z",
    "count" : 482,
    "links" : [ ... ]
  } ],
  "links" : [ ... ]
}
```

Get a single whitelist entry:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/users/5356823b3004dee3

HTTP/1.1 200 OK

{
  "ipAddress" : "12.34.56.78",
  "created" : "2014-04-23T16:17:44Z",
  "count" : 482,
  "links" : [ ... ]
}
```

Add entries to a user's whitelist:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/users/5356823b3004dee3
[ {
    "ipAddress" : "76.54.32.10"
  }, {
    "ipAddress" : "2.3.4.5"
} ]

HTTP/1.1 201 Created
Location: https://mms.mongodb.com/api/public/v1.0/users/5356823b3004dee37132bb7b/whitelist

{
  "totalCount" : 3,
  "results" : [ {
    "ipAddress" : "12.34.56.78",
    "created" : "2014-04-23T16:17:44Z",
    "count" : 0,
    "links" : [ ... ]
  }, {
```

```
    "ipAddress" : "76.54.32.10",
    "created" : "2014-04-23T16:23:44Z",
    "count" : 0,
    "links" : [ ... ]
  }, {
    "ipAddress" : "2.3.4.5",
    "created" : "2014-04-23T16:23:44Z",
    "count" : 0,
    "links" : [ ... ]
  } ],
  "links" : [ ... ]
}
```

Delete an entry from a user's whitelist:

```
curl -i -u "username:apiKey" --digest "https://mms.mongodb.com/api/public/v1.0/users/5356823b3004dee3
```

```
HTTP/1.1 200 OK
```

7.9 Monitoring Agent Configuration

Connection Settings

For the Monitoring Agent communication with the MMS Servers, the following connection settings are **required**:

mmsApiKey
Type: string

The MMS agent API key for a MMS group. The API Key can be found in the MMS interface on the *Settings* in the *Agent API Settings* section. For example:

```
mmsApiKey=abc123
```

mmsBaseUrl
Type: string

The URL of the MMS Web Server.

Set this to the URL of your MMS HTTP Service. For example:

```
mmsBaseUrl=http://example.com:8080
```

HTTP Proxy Settings

httpProxy
Type: string

To connect to the MMS HTTP Service via a proxy, enter the URL of the proxy here. For example:

```
httpProxy=http://example-proxy.com:8080
```

New in version 2.3.1 of the Monitoring Agent.

MongoDB SSL Settings

Specify these settings when the Monitoring Agent is connecting to MongoDB instances with SSL.

useSslForAllConnections
Type: boolean

Set to `true` to override any preferences expressed in the MMS UI and use SSL for all MongoDB connections. If set to true `sslTrustedServerCertificates` is required.

sslTrustedServerCertificates
Type: string

The path on disk that contains the trusted certificate authority certificates in PEM format. These certificates will verify the server certificate returned from any MongoDBs running with SSL. For example:

```
sslTrustedServerCertificates=/etc/mongodb-mms/mongodb-certs.pem
```

sslRequireValidServerCertificates
Type: boolean

Use this option to disable certificate verification by setting this value to `false`. That configuration is only recommended for testing purposes as it makes connections susceptible to MITM attacks.

MongoDB Kerberos Settings

See *Connect to Hosts with Kerberos Authentication*

krb5Principal
Type: string

The Kerberos principal used by the agent. For example:

```
krb5Principal=mmsagent/myhost@EXAMPLE.COM
```

krb5Keytab
Type: string

The *absolute* path to Kerberos principal's keytab file. For example:

```
krb5Keytab=/etc/mongodb-mms/mms-monitoring-agent.keytab
```

MMS Server SSL Settings

Advanced SSL settings used by the Monitoring Agent when communicating to the MMS HTTP Service.

sslTrustedMMSServerCertificate
By default the Monitoring Agent will use the trusted root CAs installed on the system. If the agent cannot find the trusted root CAs, configure these settings manually.

If the MMS HTTP Service is using a self-signed SSL certificate this setting is required.

The path on disk that contains the trusted certificate authority certificates in PEM format. The agent will use this certificate to verify that the agent is communicating with the designated MMS HTTP Service. For example:

```
sslTrustedMMSServerCertificate=/etc/mongodb-mms/mms-certs.pem
```

sslRequireValidMMSServerCertificates
Type: boolean

You can disable certificate verification by setting this value to `false`. That configuration is only recommended for testing purposes as it makes connections susceptible to *man-in-the-middle* attacks.

Munin Settings

See *Monitoring Configuration* for information on configuring Munin-node.

enableMunin
Type: boolean

Set to `false` if you do not with the Monitoring Agent to collect hardware statistics via Munin-node. The default is `true`. If Munin-node is detected, hardware statistics will be collected.

Deprecated Settings

MongoDB Authentication Settings

If all monitored MongoDB instances utilize the same MONGODB-CR username and password, these settings may be used. Setting the username and password here will override any configuration in the MMS UI.

See *MMS Agent Authentication Requirements* for information on the privileges needed for his user.

globalAuthUsername
Type: string

The MongoDB username that the Monitoring Agent will use to connect. For example:

```
globalAuthUsername=mms-monitoring-agent
```

globalAuthPassword
Type: string

The password for the `globalAuthUsername` user. For example:

```
globalAuthPassword=somePassword
```

7.10 Backup Agent Configuration

Connection Settings

For the Backup Agent communication with the MMS Servers, the following connection settings are **required**:

mmsApiKey
Type: string

The MMS agent API key for a MMS group. The API Key can be found in the MMS interface on the *Settings* in the *Agent API Settings* section. For example:

```
mmsApiKey=abc123
```

mothership
Type: string

The hostname of the MMS Backup Web Server.

https
Type: boolean

Toggles communication with the MMS Backup web server over HTTPS.

MongoDB SSL Settings

Specify these settings when the Backup Agent is connecting to MongoDB instances with SSL.

sslTrustedServerCertificates
Type: string

The path on disk that contains the trusted certificate authority certificates in PEM format. These certificates will verify the server certificate returned from any MongoDBs running with SSL. For example:

```
sslTrustedServerCertificates=/etc/mongodb-mms/mongodb-certs.pem
```

sslRequireValidServerCertificates
Type: boolean

Use this option to disable certificate verification by setting this value to `false`. That configuration is only recommended for testing purposes as it makes connections susceptible to MITM attacks.

MongoDB Kerberos Settings

See *Connect to Hosts with Kerberos Authentication*

krb5Principal
Type: string

The Kerberos principal used by the agent. For example:

```
krb5Principal=mmsagent/myhost@EXAMPLE.COM
```

krb5Keytab
Type: string

The *absolute* path to Kerberos principal's keytab file. For example:

```
krb5Keytab=/etc/mongodb-mms/backup-agent.keytab
```

MMS Server SSL Settings

Advanced SSL settings used by the Backup Agent when communicating to the MMS Backup Web Server.

sslTrustedMMSBackupServerCertificate
By default the Backup Agent will use the trusted root CAs installed on the system. If the agent cannot find the trusted root CAs, configure these settings manually.

If the MMS Backup Server is using a self-signed SSL certificate this setting is required.

The path on disk that contains the trusted certificate authority certificates in PEM format. The agent will use this certificate to verify that the agent is communicating with the designated MMS Backup Server. For example:

```
sslTrustedMMSBackupServerCertificate=/etc/mongodb-mms/mms-certs.pem
```

sslRequireValidMMSBackupServerCertificate
Type: boolean

You can disable certificate verification by setting this value to `false`. That configuration is only recommended for testing purposes as it makes connections susceptible to *man-in-the-middle* attacks.

8 Release Notes

MMS Server Changelog A record of changes to the MMS Mpplication.

Monitoring Agent Changelog A record of changes to the Monitoring Agent.

Backup Agent Changelog A record of changes to the Backup Agent.

8.1 MMS Server Changelog

On-Prem MongoDB Management Service Server 1.4.3

Released 2014-07-22

- Addressed issues related to Backup Job assignment for 2.6.x clusters that used the `clusterMonitor` role to support MMS Monitoring.
- Fixed problem importing email addresses for users for deployments that use LDAP integration.
- Fixed rare race condition caused high CPU usage in the MMS HTTP Service if the application cannot connect to one of the backing databases.
- Additional security enhancements.

On-Prem MongoDB Management Service Server `1.4.2`

Released 2014-05-29

- Critical bug fix for backing up MongoDB 2.6 deployments that include user or custom role definitions:
  - The `system.version` collection in the admin database will be included in all future snapshots.
  - The `system.roles` collection in the admin database will be included after a new initial sync is performed.

  Users capturing backups of MongoDB 2.6 replica sets or clusters with MMS that include custom role definitions should perform a new initial sync. Taking a new initial sync will ensure that the role definitions are included in the backup.
- Disable MongoDB `usePowerOf2Sizes` for insert-only MMS Backup collections.
- Speed optimization for MMS Backup HTTP pull restores.
- Fix for LDAP integration, MMS now passes full `dn` correctly when authenticating the user.

On-Prem MongoDB Management Service Server `1.4.1`

Released 2014-04-28

- Ability to Backup replica sets or clusters using Kerberos authentication
- Ability to Backup replica sets or clusters running on custom MongoDB builds
- Fix validation issue preventing Backup of MongoDB 2.6.0 clusters
- Reduced log noise from Monitoring Agent when monitoring MongoDB 2.0 or unreachable mongods

On-Prem MongoDB Management Service Server `1.4.0`

Released 2014-04-08

- Includes MMS Backup: continuous backup with point-in-time recovery of replica sets and cluster-wide snapshots of sharded clusters.
- Finer-grained roles and permissions.
- Improved user interface for alerts.
- Enhanced Activity Feed for auditing of all activity.
- Monitoring Agent distributed as OS-specific binary. Python dependency removed.
- LDAP integration for managing users and groups.

MMS OnPrem 1.4.0 requires MongoDB 2.4.9+ instances for *backing storage*.

On-Prem MongoDB Management Service Server `1.3.0`

Released 2013-12-01

- Packaging/support for Debian and SUSE Linux.
- Kerberos authentication support between MMS server and backing MongoDBs, as well as between Monitoring Agent and the MongoDBs it monitors.
- OnPrem users can be overall site administrators. (MMS Admins)
- New admin section where MMS Admins can manage user roles and message banners.
- Tunable advanced password and session management configurations.
- Encryption key rotation, more specific CORS policy, auth tokens removed from chart URLs, and other security enhancements.

On-Prem MongoDB Management Service Server `1.2.0`

Released 2013-07-24

- Redesigned user interface and enhanced algorithm to auto-discover hosts and derive host topology.
- SNMP monitoring.
- Ability to export charts.
- Option to store encrypted authentication credentials in the `mmsDb` property in the configuration file.
- Ability to classify users within an MMS Group as group administrators or read-only users.

8.2 Monitoring Agent Changelog

Monitoring Agent `2.3.1.89-1`

Released with OnPrem 1.4.3

- Improved logging for MongoDB 2.6 config servers when connecting with a user that has the built-in clusterMonitor role.
- Fixes issues with connecting to replica set members that use auth with an updated Go client library.

- Added support for HTTP proxy configuration in the agent configuration file.
- Agent includes support for an Offline data collection mode.

Monitoring Agent `2.1.4.51-1`

Released with MMS OnPrem 1.4.2

Prevent high CPU use when monitoring unreachable `mongod`.

Monitoring Agent `2.1.3.48-1`

Released with OnPrem 1.4.1

Reduction in unnecessary log messages for unsupported operations on monitored MongoDB 2.2 instances.

Monitoring Agent `2.1.1.41-1`

Released with OnPrem 1.4.0

Ability to monitor hosts using Kerberos authentication.

Monitoring Agent `1.6.6`

Released with OnPrem1.3

- Added kerberos support for agents running on Python 2.4.x.
- Added logging when the `dbstats` command fails.

8.3 Backup Agent Changelog

Backup Agent `1.5.1.83-1`

Released with MMS OnPrem 1.4.2

Critical update for users running the MongoDB 2.6 series that use authorization.

The Backup Agent now includes `system.version` and `system.role` collections from the admin database in the initial sync.

Backup Agent `1.5.0.57-1`

Released with OnPrem 1.4.1

Support for backing up Kerberos-authenticated replica sets and clusters

Backup Agent `1.4.6.42-1`

Released with OnPrem 1.4.0

- Major stability update.
- Prevent a file descriptor leak.
- Correct handling of timeouts for connections hung in the SSL handshaking phase.

Index

N

O

P

Q

R

S

T

U