Manually Deploy MongoDB on EC2

The following steps can be used to deploy MongoDB on EC2 yourself. The instances will be configured with the following characteristics:

- Amazon Linux
- MongoDB installed via `yum`
- Individual PIOPS EBS volumes for data (1000 IOPS), journal (250 IOPS), and log (100 IOPS)
- Updated read-ahead values for each block device
- Update ulimit settings

Before continuing be sure to have the following:

- Install EC2 command line tools
- Generate an EC2 key pair for connecting to the instance via SSH
- Create a security group that allows SSH connections

Create the instance using the key pair and security group previously created and also include the `--ebs-optimized` flag and specify individual PIOPS EBS volumes (`/dev/xvdf` for data, `/dev/xvdg` for journal, `/dev/xvdh` for log). Refer to the documentation for ec2-run-instances for more information on devices and parameters.:

```
ec2-run-instances ami-05355a6c -t m1.large -g [SECURITY-GROUP] -k [KEY-PAIR] -b "/dev/xvd
```

You can use the returned instance-id to ascertain the IP Address or DNS information for the instance:

```
ec2-describe-instances [INSTANCE-ID]
```

Now SSH into the instance:

```
ssh -i /path/to/keypair.pem ec2-user@ec2-1-2-3-4.amazonaws.com
```

After login, update installed packages, add the MongoDB yum repo, and install MongoDB:

```
echo "[mongodb-org-3.2]
name=MongoDB Repository
baseurl=https://repo.mongodb.org/yum/amazon/2013.03/mongodb-org/3.2/x86_64/
gpgcheck=1
enabled=1
gpgkey=https://www.mongodb.org/static/pgp/server-3.2.asc" |
  sudo tee -a /etc/yum.repos.d/mongodb-org-3.2.repo
```

```
sudo yum -y update && sudo yum install -y mongodb-org-server \
    mongodb-org-shell mongodb-org-tools
```

Next, create/configure the mount points, mount each volume, set ownership (MongoDB runs under the `mongod` user/group), and set the `/journal` link:

```
sudo mkdir /data /log /journal

sudo mkfs.ext4 /dev/xvdf
sudo mkfs.ext4 /dev/xvdg
sudo mkfs.ext4 /dev/xvdh

echo '/dev/xvdf /data ext4 defaults,auto,noatime,noexec 0 0
/dev/xvdg /journal ext4 defaults,auto,noatime,noexec 0 0
/dev/xvdh /log ext4 defaults,auto,noatime,noexec 0 0' | sudo tee -a /etc/fstab

sudo mount /data
sudo mount /journal
sudo mount /log

sudo chown mongod:mongod /data /journal /log

sudo ln -s /journal /data/journal
```

Now configure the following MongoDB parameters by editing the configuration file `/etc/mongod.conf` so that it contains the following:

```
dbpath = /data
logpath = /log/mongod.log
```

If you don’t want MongoDB to start at boot, you can issue the following command:

```
sudo chkconfig mongod off
```

By default Amazon Linux uses `ulimit` settings that are not appropriate for MongoDB. To setup `ulimit` to match the documented ulimit settings, run the following command:

```
echo '* soft nofile 64000
* hard nofile 64000
* soft nproc 64000
* hard nproc 64000' | sudo tee /etc/security/limits.d/90-mongodb.conf
```

Additionally, default read ahead settings on EC2 are not optimized for MongoDB. As noted in the read-ahead settings from Production Notes, you should adjust the settings to read approximately 32 blocks (or 16 KB) of data. The following command will set the readahead appropriately (repeat for additional volumes):

```
sudo blockdev --setra 32 /dev/xvdf
```

To make this change persistent across system boot, issue the following command:

```
echo 'ACTION=="add", KERNEL=="xvdf", ATTR{bdi/read_ahead_kb}="16"' | sudo tee -a /etc/ude
```

Once again, repeat the above command for all required volumes (note: the device we created was named `/dev/xvdf` but the name used by the system is `xvdf`).

To start `mongod`, issue the following command:

```
sudo service mongod start
```

Then connect to the MongoDB instance using the `mongo` shell:

```
mongo
```

To have MongoDB startup automatically at boot issue the following command:

```
sudo chkconfig mongod on
```

For production deployments consider using Replica Sets or Sharding.