MANTA Flow on Amazon EKS (39.x)

- Overview
- Prerequisites
- Amount of Time Needed to Complete This Guide
- Steps to Launch MANTA Flow in Amazon Elastic Kubernetes Service (Amazon EKS)
  - AWS CLI Initialization
  - AWS CLI Login
  - kubectl Initialization
  - Cluster Check
- MANTA Flow Deployment
  - Namespace Initialization
  - Secret Initialization
  - Ingress Initialization
  - MANTA Flow Deployment
- Check the Health of the MANTA Flow Deployment
- Accessing MANTA Flow
- MANTA Flow Scaling
  - Vertical Scaling
- Upgrading MANTA Flow
- Troubleshooting MANTA Flow
  - Troubleshooting Information to be Provided in MANTA Helpdesk Tickets
- Steps to Delete MANTA Flow
- Recommended Sizing
  - Worker Node (EC2) Sizing
  - EBS Volume Sizing
- Backup and Disaster Recovery
  - Backup
  - Disaster Recovery
  - Recovery in Case of an Availability Zone (AZ) Failure

Overview

For architecture details on MANTA Flow in Kubernetes, please read https://mantatools.atlassian.net/wiki/spaces/MTKB/pages/3544645633/MANTA+Flow+Container+Architecture+39.x.

Prerequisites

- Amazon EKS cluster with:
  - EBS CSI plugin installed—see https://docs.aws.amazon.com/eks/latest/userguide/ebs-csi.html
  - ALB controller installed—see https://docs.aws.amazon.com/eks/latest/userguide/alb-ingress.html
- The following locally installed tools (on laptop/VM/terminal):
  - AWS CLI installed—see https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-install.html for instructions
  - kubectl installed—see https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/ for instructions
- Valid AWS credentials
- Valid credentials for the MANTA Docker repository `repo.getmanta.com` [view link]
- MANTA Flow Amazon EKS manifests downloaded from MANTA Portal
- Valid `license.key` for MANTA Flow

Amount of Time Needed to Complete This Guide

The estimated amount of time needed to complete this guide is approximately 10 minutes, if the prerequisites are met.

Steps to Launch MANTA Flow in Amazon Elastic Kubernetes Service (Amazon EKS)

AWS CLI Initialization

Verify that your AWS CLI is working.

```
aws --version
```

The versions should be printed similar to the example below.

```
aws-cli/1.22.22 Python/3.8.10 Linux/5.10.60.1-microsoft-standard-WSL2
botocore/1.23.22
```

AWS CLI Login

Log in to AWS services using:

```
aws configure
```

Provide the access key and secret key given to you by ITS as requested.

```
AWS Access Key ID [None]: <access key id>
AWS Secret Access Key [None]: <secret key>
Default region name [None]: eu-central-1
Default output format [None]: json
```

kubectl Initialization

Configure kubectl to work with the cluster. Use the correct `<region>` and `<cluster name>`.

```
aws eks update-kubeconfig --region "<region>" --name "<cluster name>"
```

Output:

```
Updated context arn:aws:eks:eu-central-1:071165050248:cluster/test3 in
/home/ubuntu/.kube/config
```

Cluster Check

Verify that kubectl is properly configured to work with the cluster.

```
kubectl cluster-info
```

Output:

```
Kubernetes master is running at https://A000E9DD98047F6200F072D8A1A54C5E.
gr7.eu-central-1.eks.amazonaws.com
CoreDNS is running at https://A000E9DD98047F6200F072D8A1A54C5E.gr7.eu-
central-1.eks.amazonaws.com/api/v1/namespaces/kube-system/services/kube-
dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info
dump'.
```

Next, verify that the ELB CSI plugin and ALB Controller, including cert-manager, are deployed properly.

```
kubectl -n kube-system get deploy
kubectl -n cert-manager get deploy
```

You should see an output similar to the one below with all deployments *READY* and *AVAILABLE*.

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   1/1     1            1           6m33s
coredns                        2/2     2            2           26m
ebs-csi-controller             2/2     2            2           6m51s

NAME                      READY   UP-TO-DATE   AVAILABLE   AGE
cert-manager              1/1     1            1           6m57s
cert-manager-cainjector   1/1     1            1           6m57s
cert-manager-webhook      1/1     1            1           6m57s
```

If there are errors, some of the *READY* and *AVAILABLE* numbers will be *0*.

Also, you can make sure the pods have the status *Running*.

```
kubectl -n kube-system get pods
kubectl -n cert-manager get pods
```

Output:

```
NAME                                           READY   STATUS
RESTARTS   AGE
aws-load-balancer-controller-657d78d85b-z9xfb   1/1     Running
0          2m5s
aws-node-q9khx                                 1/1     Running
0          20m
aws-node-s2g4f                                 1/1     Running
0          20m
coredns-7cfc675d7d-ndgbr                       1/1     Running
0          26m
coredns-7cfc675d7d-zxt5l                       1/1     Running
0          26m
ebs-csi-controller-854b867774-4jxl6            4/4     Running
0          3m19s
ebs-csi-controller-854b867774-jj9jz            4/4     Running
0          3m19s
ebs-csi-node-b7c65                             3/3     Running
0          7m51s
ebs-csi-node-rgzz2                             3/3     Running
0          7m51s
kube-proxy-h6n9l                               1/1     Running
0          20m
kube-proxy-mlb74                               1/1     Running
0          20m

NAME                                      READY   STATUS    RESTARTS
AGE
cert-manager-68ff46b886-qpqvq             1/1     Running   0
10m
cert-manager-cainjector-7cdbb9c945-jj4jn  1/1     Running   0
10m
cert-manager-webhook-67584ff488-q8dt9     1/1     Running   0
10m
```

MANTA Flow Deployment

Namespace Initialization

It is recommended to use a dedicated namespace for MANTA Flow deployment. Please make sure you use a unique name for the namespace. We will use `<namespace>` in the guide below, so please replace `<namespace>` with the name chosen.

```
kubectl create namespace <namespace>
```

Output:

```
namespace/<namespace> created
```

Secret Initialization

Create the credentials for `repo.getmanta.com`.

```
kubectl -n <namespace> create secret docker-registry manta-registry-
credentials --docker-server=repo.getmanta.com --docker-username=<your
username> --docker-password=<your password>
```

Output:

```
secret/manta-registry-credentials created
```

Create the secret for the MANTA Flow license key. **(The name of the .key file must be "license.key".)**

```
kubectl -n <namespace> create secret generic mantaflow-keys --from-
file=<path to license.key>/license.key
```

Output:

```
secret/mantaflow-keys created
```

Provide the initial `<username>` and `<password>` and a `<password for masterkeystore encryption>`. **Please escape the username and passwords if you use special characters.**

```
kubectl -n <namespace> create secret generic mantaflow-credentials \
  --from-literal=MANTA_USER=<manta admin username> \
  --from-literal=MANTA_PASSWORD=<manta admin password> \
  --from-literal=MANTA_MASTERPASSWORD=<password for masterkeystore
encryption>
```

Output:

```
secret/mantaflow-credentials created
```

You can verify that the credentials were successfully created.

```
kubectl -n <namespace> get secret
```

Output:

```
NAME                           TYPE                                  DATA
AGE
...
manta-registry-credentials     kubernetes.io/dockerconfigjson        1
2m56s
mantaflow-credentials          Opaque                                3
21s
mantaflow-keys                 Opaque                                1
58s
```

Ingress Initialization

Download the Kubernetes manifests for EKS from MANTA Portal `manta-dataflow-manifests-eks-39.x.x.zip` and extract their contents. The contents should be unpacked in the `manta-dataflow` directory.

```
unzip manta-dataflow-manifests-eks-39.x.x.zip
```

Deploy ingress.

```
kubectl -n <namespace> apply -f ./manta-dataflow/ingress.yaml
```

Output:

```
ingress.extensions/manta-dataflow-ingress created
```

Obtain the URL of the deployment by saving it to a variable `ingressUrl`.

```
export ingressUrl=$(kubectl -n <namespace> get ingress manta-dataflow-
ingress -o jsonpath='{.status.loadBalancer.ingress[0].hostname}')
echo $ingressUrl
```

You should see something like the example below.

```
internal-k8s-mantadat-mantadat-ebd43728ff-1576178726.eu-central-1.elb.
amazonaws.com
```

Create the following config map.

```
kubectl -n <namespace> create configmap manta-config \
  --from-literal MANTA_AUTH=http://$ingressUrl/auth/ \
  --from-literal MANTA_CONFIGURATION_SERVICE_URL=http://manta-
configuration-service:8083 \
  --from-literal MANTA_FLOW_SERVER_URL=http://$ingressUrl/manta-dataflow-
server \
  --from-literal MANTA_ADMIN_UI_URL=http://$ingressUrl/manta-admin-gui \
  --from-literal MANTA_ARTEMIS_HOST=manta-artemis \
  --from-literal MANTA_ARTEMIS_PORT=61616
```

MANTA Flow Deployment

Deploy MANTA Flow by running the following commands.

```
kubectl -n <namespace> apply -f ./manta-dataflow/pvc.yaml
kubectl -n <namespace> apply -f ./manta-dataflow/manta-auth.yaml
kubectl -n <namespace> apply -f ./manta-dataflow/manta-configuration-
service.yaml
kubectl -n <namespace> apply -f ./manta-dataflow/manta-artemis.yaml
kubectl -n <namespace> apply -f ./manta-dataflow/manta-dataflow.yaml
kubectl -n <namespace> apply -f ./manta-dataflow/manta-admin-gui.yaml
kubectl -n <namespace> apply -f ./manta-dataflow/manta-flow-agent.yaml
```

Output:

```
persistentvolumeclaim/keycloak-data-claim created
persistentvolumeclaim/server-logs-claim created
persistentvolumeclaim/server-conf-claim created
persistentvolumeclaim/server-temp-claim created
persistentvolumeclaim/server-gfx-claim created
persistentvolumeclaim/admin-conf-claim created
persistentvolumeclaim/admin-logs-claim created
persistentvolumeclaim/admin-data-claim created
persistentvolumeclaim/cli-platform-conf-claim created
persistentvolumeclaim/cli-scenarios-conf-claim created
persistentvolumeclaim/cli-lib-ext-claim created
persistentvolumeclaim/cli-data-claim created
deployment.apps/manta-auth created
service/manta-keycloak created
deployment.apps/manta-artemis created
service/manta-artemis created
service/manta-artemis-management created
deployment.apps/manta-dataflow created
service/manta-admin-gui created
service/manta-dataflow-server created
```

Check the Health of the MANTA Flow Deployment

You can then check the status of the volumes and pods using the following command.

```
kubectl -n <namespace> get pvc,pod
```

You should see that all the volumes have the status *Bound* and that the pods have the status *Running* in the output.

```
NAME                                                STATUS
VOLUME                                        CAPACITY   ACCESS MODES
STORAGECLASS   AGE
persistentvolumeclaim/admin-conf-claim              Bound    pvc-e817ce53-
085d-4e2e-ac22-855e536042e0   1Gi          RWO               ebs-sc            24m
persistentvolumeclaim/admin-data-claim              Bound    pvc-0cfa7f1e-
32f2-473b-9b48-57873218f13c   10Gi         RWO               ebs-sc            24m
persistentvolumeclaim/admin-logs-claim              Bound    pvc-f4d6b4e1-
935b-40e2-a740-0eba1a4ec608   1Gi          RWO               ebs-sc            24m
persistentvolumeclaim/cli-data-claim                Bound    pvc-eed3f3f6-
14f0-46c8-bb44-4ad22a1d7f2d   10Gi         RWO               ebs-sc            24m
persistentvolumeclaim/cli-lib-ext-claim             Bound    pvc-1c7a421b-
66aa-4e38-9541-69610bbf0444   1Gi          RWO               ebs-sc            24m
persistentvolumeclaim/cli-platform-conf-claim       Bound    pvc-f4218290-
e44d-461a-8d84-06fb5251bc26   1Gi          RWO               ebs-sc            24m
persistentvolumeclaim/cli-scenarios-conf-claim      Bound    pvc-a30f74bb-
7208-45c9-be2b-a5c9a6a8b87a   1Gi          RWO               ebs-sc            24m
persistentvolumeclaim/keycloak-data-claim           Bound    pvc-f561fafa-
d634-4a43-bf03-b8452796d5b8   1Gi          RWO               ebs-sc            25m
persistentvolumeclaim/server-conf-claim             Bound    pvc-720fbab7-
90fe-4400-8fa0-161fbd6c4929   1Gi          RWO               ebs-sc            24m
persistentvolumeclaim/server-logs-claim             Bound    pvc-f8104b25-
9010-4e8d-b6a5-647041c95c4e   1Gi          RWO               ebs-sc            22m
persistentvolumeclaim/server-temp-claim             Bound    pvc-611bf847-
061a-4e67-898f-7fa741493fcc   10Gi         RWO               ebs-sc            24m

NAME                                  READY   STATUS    RESTARTS   AGE
pod/manta-auth-69cccf6f9d-4vtl5       1/1     Running   0          25m
pod/manta-dataflow-59c484895b-wbtjr   1/1     Running   0          2m30s
pod/manta-artemis-677b6bbc66-gf5gw    1/1     Running   0          2m56s
```

To further debug any problems in the resources created, you can use:

```
kubectl -n <namespace> describe pvc,svc,deploy,ingress
```

Accessing MANTA Flow

MANTA Flow is deployed through an ingress controller that maps both Admin UI and Server on different ports to the same port, 80, and automatically registers the ingress URL. See above for information on how to obtain the ingress URL that is shown in the example below for the deployment.

- `internal-k8s-mantadat-mantadat-ebd43728ff-1576178726.eu-central-1.elb.amazonaws.com`

Now you should be able to reach MANTA Server and MANTA Admin UI at the ingress address.

- `http://<ingressUrl>/manta-dataflow-server/`
- `http://<ingressUrl>/manta-admin-gui/app/`

So, for the example above:

- `http://internal-k8s-mantadat-mantadat-ebd43728ff-1576178726.eu-central-1.elb.amazonaws.com/manta-dataflow-server/`
- `http://internal-k8s-mantadat-mantadat-ebd43728ff-1576178726.eu-central-1.elb.amazonaws.com/manta-admin-gui/app/`

MANTA Flow Scaling

Vertical Scaling

Default resource requirement specifications are provided for all pods in the manifests. For example, for `manta-dataflow` pod you would see the following resource requests and limits set in `manta-dataflow.yaml`.

```
        resources:
          requests:
            memory: "4Gi"
            cpu: "3.5"
          limits:
            memory: "16Gi"
            cpu: "7"
```

Please update the `limits` values for the pods accordingly, if larger values are required. There is no reason to change the `requests` values, as those are the minimum values needed for the pods to start successfully.

Upgrading MANTA Flow

Please note that downgrading to an older version is not permitted. If you enter a version older than the current version of the deployment, the containers will fail to start.

Please back up all persistent volumes before upgrading.

Delete MANTA Flow deployments and apply the new version of the manifests.

```
kubectl delete deployments,services -l app=manta-dataflow
```

Output:

```
deployment.apps/manta-dataflow deleted
deployment.apps/manta-admin-gui deleted
deployment.apps/manta-configuration-service deleted
deployment.apps/manta-auth deleted
deployment.apps/manta-artemis deleted
```

Download and unpack the new version of the manifests. In this case, `manta-dataflow-manifests-eks-39.x.x.zip`.

```
unzip manta-dataflow-manifests-eks-39.x.x.zip
```

Follow the steps in the Ingress Initialization and MANTA Flow Deployment sections of this guide.

The resources will be overridden by the new version. After the deployment, please wait for the pods to start running. Once the pods are running, MANTA Flow should be running the new version.

Troubleshooting MANTA Flow

To check the logs of a running pod, please use the following command. You have to provide the correct pod ID from the list of kubectl get pods.

```
kubectl -n <namespace> logs <pod ID>
```

To troubleshoot problems, it is possible to connect to the pod using the command below. You have to provide the correct pod ID from the list of kubectl get pods.

```
kubectl -n <namespace> exec -it <pod ID> -- /bin/bash
```

Then, you will be in the container bash and you can check what you need to.

Troubleshooting Information to be Provided in MANTA Helpdesk Tickets

If you open a troubleshooting ticket in MANTA Helpdesk, please provide the following.

1. Output of `kubectl -n <namespace> get pods,pvc,svc,ingress`
2. Output of `kubectl -n <namespace> get pods -o yaml`
3. Output of `kubectl -n <namespace> describe pod <pod id>` for pods whose state is not *Running*
4. Output of `kubectl -n <namespace> logs <pod id>` for all pods, with each log provided as a separate file
5. MANTA Flow Server logs, which can be obtained by running `kubectl -n <namespace> cp manta-dataflow:server/logs ./server-logs`; the copied directory can be sent as a ZIP or TAR
6. MANTA Admin GUI logs, which can be obtained by running `kubectl -n <namespace> cp manta-admin-gui:serviceutility/log ./admin-gui-logs`; the copied directory can be sent as a ZIP or TAR

Steps to Delete MANTA Flow

Delete the deployment. Please note that this command will delete all persistent volumes. If you want to keep the volumes, please only delete the deployment, services, ingress, and persistent volume claims.

```
kubectl -n <namespace> delete -f manta-dataflow/
```

Output:

```
ingress.networking.k8s.io "manta-dataflow-ingress" deleted
deployment.apps "manta-auth" deleted
persistentvolumeclaim "keycloak-data-claim" deleted
service "manta-keycloak" deleted
deployment.apps "manta-dataflow" deleted
service "manta-admin-gui" deleted
service "manta-dataflow-server" deleted
persistentvolumeclaim "server-logs-claim" deleted
persistentvolumeclaim "server-conf-claim" deleted
persistentvolumeclaim "server-temp-claim" deleted
persistentvolumeclaim "admin-conf-claim" deleted
persistentvolumeclaim "admin-logs-claim" deleted
persistentvolumeclaim "admin-data-claim" deleted
persistentvolumeclaim "cli-platform-conf-claim" deleted
persistentvolumeclaim "cli-scenarios-conf-claim" deleted
persistentvolumeclaim "cli-lib-ext-claim" deleted
persistentvolumeclaim "cli-data-claim" deleted
```

The deployment will be stopped after a while.

Please delete the secrets and config map if necessary.

You can make sure there are no MANTA Flow resources using the command.

```
kubectl -n <namespace> get pv,pvc,svc,deploy,pods
```

Output:

```
NAME                  TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
service/kubernetes    ClusterIP   10.96.0.1     <none>         443/TCP
3d14h
```

Recommended Sizing

The minimum container requirements and storage requirements are described in MANTA Flow Container Architecture (39.x).

Worker Node (EC2) Sizing

The recommended instance types for worker nodes in the Amazon EKS cluster are as follows.

- t3.2xlarge
- m5.2xlarge
- m5d.2xlarge

You can use more performant instance types if needed.

EBS Volume Sizing

The suggested size is the minimum reasonable size for MANTA Flow to work successfully. The volumes may be larger depending on the amount of data being analyzed.

The recommended volume types are:

- gp3—providing at least 3000 IOPS
- io2—providing 500 IOPS per provisioned GB

Backup and Disaster Recovery

All MANTA Flow application data (metadata, internal database, configuration, logs, etc.) is stored in EBS volumes. Thus, MANTA Flow is backed up by backing up the EBS persistent volumes (PVs).

We recommend using an EBS volume snapshot, which can be used through Kubernetes Custom Resource Definitions—Kubernetes Volume Snapshot CRDs. AWS has published a thorough step-by-step guide to using EBS volume snapshots to back up EBS backed EKS persistent volumes at https://aws.amazon.com/blogs/containers/using-ebs-snapshots-for-persistent-storage-with-your-eks-cluster/.

Backup

To back up MANTA Flow:

1. Install volume snapshot custom resource definitions (CRDs) to the EKS cluster using the guide above—**Step 3: Install the volume snapshot custom resource definitions (CRDs) and volume snapshot controller**.
2. Create volume snapshots for all persistent volumes being used in MANTA Flow using the guide above—**Step 4: Create a volume snapshot class and snapshot**.

Disaster Recovery

For disaster recovery:

1. Recover all the volumes from their snapshots using the guide above—**Step 5: Create a persistent volume by restoring the volume snapshot**.
2. Update the definitions for persistent volume claims to use recovered persistent volumes (PVs) and deploy PVCs.
3. Launch MANTA Flow deployment again according to the instructions in this guide. The initialization containers will detect that the volumes have already been initialized, and thus, the initialization will be skipped and the main MANTA Flow container will be launched with the restored volumes.

Recovery in Case of an Availability Zone (AZ) Failure

To recover in case of an AZ failure, you have to start a new cluster using the availability zones available for the:

- Networks
- EKS cluster
- Node groups

Then, recover the MANTA Flow deployment the same way as before, from the volume snapshots, supposing that the volume snapshots are still available.