Quark Platform Installation Instructions

1. Quark Platform

Quark is a self-service bioinformatics platform that helps Enterprises accelerate drug discovery. This document captures the steps for launching the Quark Platform through AWS Marketplace. Platform uses AWS Cognito for the User management and OIDC provider and Route53 hosted zone for domain management.

2. Pre Requisites

Before launching the Quark Platform through AWS Marketplace, please make sure you have completed the following prerequisites.

2.1. Create Amazon EKS Cluster

Create a new EKS Cluster with an OIDC provider or use any existing cluster which has support for OIDC provider.

To create a new EKS Cluster using EKSCTL follow the link:
https://eksctl.io/usage/creating-and-managing-clusters/

2.2. EKS OIDC provider ID

Once the cluster is created, note down the OIDC provider ID (**EKS OIDC ID**).

2.3. Create IAM Role for Identity Manager

Quark Platform has a component called Identity Manager which automatically creates IAM Roles for Service Accounts (IRSA) for pods that need access to AWS Services.You can find more information about Identity Manager here.

Create an IAM role which will be used by the identity manager with the below trust and iam policy.

Trust policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::<AWS ACCOUNT
ID>:oidc-provider/oidc.eks.<REGION>.amazonaws.com/id/<OIDC ID>"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
```

```
            "Condition": {
                "StringEquals": {
                    "oidc.eks.<REGION>.amazonaws.com/id/<OIDC ID>:aud":
"sts.amazonaws.com",
                    "oidc.eks<REGION>.amazonaws.com/id/<OIDC ID>:sub":
"system:serviceaccount:gravity-system:identity-manager"
                }
            }
        }
    ]
}
```

Role policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:CreateRole",
                "iam:CreateInstanceProfile",
                "iam:CreateOpenIDConnectProvider",
                "iam:CreatePolicy",
                "iam:AddRoleToInstanceProfile",
                "iam:AttachRolePolicy",
                "iam:DeleteInstanceProfile",
                "iam:DeleteOpenIDConnectProvider",
                "iam:RemoveRoleFromInstanceProfile",
                "iam:TagOpenIDConnectProvider",
                "iam:TagRole",
                "iam:UpdateAssumeRolePolicy"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "IAMPermissions1"
        },
        {
            "Action": [
                "iam:DeleteRole",
                "iam:CreatePolicyVersion",
                "iam:DeletePolicyVersion",
           "iam:DeletePolicy",
                "iam:Untag*",
                "iam:Tag*",
                "iam:AttachRolePolicy",
```

```
                "iam:DetachRolePolicy",
                “iam:DeleteRolePolicy",
                "iam:PutRolePolicy",
                "iam:DeleteOpenIDConnectProvider",
                "iam:UpdateAssumeRolePolicy",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "IAMPermissions2"
        },
        {
            "Action": [
                "iam:GetRole",
                "iam:List*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "IAMPermissions3"
        }
    ]
}
```

```
aws iam create-role --role-name iam-manager-role --assume-role-policy-document
file://trust-policy.json

aws iam put-role-policy --role-name iam-manager-role --policy-name
quark-iam-manager-role-policy --policy-document file://role-policy.json
```

2.4. IAM Role ARN

Once you have created the IAM Role with the above trust and IAM policy, note down the ARN (**IAM MANAGER ROLE ARN**).

2.5. Install Tools

Install the below tools by following the links.

1. Kubectl: https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/
2. Helm: https://helm.sh/docs/intro/install/

2.6. Configure AWS CLI

Follow the instructions from here https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html to install AWS CLI.

2.7. Configure Kubeconfig

Follow the link https://docs.aws.amazon.com/eks/latest/userguide/create-kubeconfig.html to download and connect to EKS Cluster.

Note: Ensure that you configure the AWS CLI and have access to the AWS EKS Cluster through kubectl

2.8. Have a Domain ready for the platform

You need to have a domain name that you can provide for hosting the Quark platform. During the installation process, Gravity platform will primarily create two endpoints:

1. Platform endpoint: **https://*example.com***
2. Authentication callback endpoint: **https://*example.com*/sso/callback**

Quark platform needs a public hosted zone to expose the platform endpoints. Based on the domain name, platform endpoints will be generated automatically.

Create a hosted zone for your domain using the below command.

```
aws route53 create-hosted-zone --name <DOMAIN_NAME> --caller-reference <UNIQUE_VALUE>
```

2.9. Create Cognito User Pool, Client and Domain

Quark uses AWS Cognito for user authentication into the platform. Users added in Cognito can use their Cognito Username and Password to login to Gravity.

Use the below commands to create a cognito user pool and app client.

```
aws cognito-idp create-user-pool --pool-name <POOL_NAME>
--admin-create-user-config  AllowAdminCreateUserOnly=true
```

```
aws cognito-idp create-user-pool-client --user-pool-id <USER_POOL_ID>
--client-name gravity --explicit-auth-flows ALLOW_REFRESH_TOKEN_AUTH
--generate-secret

 aws cognito-idp update-user-pool-client --user-pool-id <USER_POOL_ID>
--client-id <CLIENT_ID> --client-name "gravity" --callback-urls <CALLBACK_URL>
--allowed-o-auth-flows-user-pool-client --allowed-o-auth-flows  code
--allowed-o-auth-scopes email openid profile --supported-identity-providers
COGNITO --explicit-auth-flows ALLOW_REFRESH_TOKEN_AUTH
--prevent-user-existence-errors ENABLED
```

Use the below command to create a Cognito domain name. Provide a domain name prefix. This prefix can be a unique identifier.

```
aws cognito-idp create-user-pool-domain --user-pool-id <USER_POOL_ID> --domain
<PREFIX_DOMAIN_NAME>

Ex: aws cognito-idp create-user-pool-domain --user-pool-id us-east-1_assasaC13N
--domain quarkdemo
```

Note down the Client ID, Client secret, Cognito issuer URL and Cognito Pool ID.

2.10. Default Project Name and Platform Owner

Quark has the concept of “Projects” to organize resources. You can create multiple projects within Quark to organize resources by team, customer, etc.

1. When you install the platform for the first time, a “default” project needs to be created.
2. Groups in Cognito need to match the project name so that users in a Cognito Group can seamlessly access respective projects in Gravity.
3. During installation, you will provide a user who is the platform owner. This user will be the first user to login to the platform, has full rights to the platform and can onboard other users to the platform.

Have the following values handy before proceeding to the next step.

1. **DEFAULT_PROJECT_NAME**: A name for the default project. Constraints:
    a. Max 16 characters in length.

   b. Must start with an alphabet and can contain only lowercase alphanumeric characters.
   c. Cannot be “default”, “system”
2. **PLATFORM_OWNER_EMAIL**: Email address of the platform owner
3. **PLATFORM_OWNER_USERNAME**: Name of the platform owner

Create a group in cognito with the same name as the project name. Invites functionality will search users in the project name as group name.

2.11. Create Cognito Group and Platform Owner User

Use the below commands to create a user and group and set password for the user.

```
aws cognito-idp create-group --user-pool-id <USER_POOL_ID> --group-name
<DEFAULT_PROJECT_NAME> --description "Quark Default Project Group"
```

```
aws cognito-idp admin-create-user --user-pool-id <USER_POOL_ID> --username
<PLATFORM_OWNER_USERNAME> --user-attributes
Name=email,Value=<PLATFORM_OWNER_EMAIL> --message-action SUPPRESS
```

```
aws cognito-idp  admin-set-user-password --user-pool-id <USER_POOL_ID>
--username <PLATFORM_OWNER_USERNAME> --password <PASSWORD> --permanent
```

```
aws cognito-idp admin-add-user-to-group --user-pool-id <USER_POOL_ID> --username
<PLATFORM_OWNER_USERNAME> --group-name <DEFAULT_PROJECT_NAME>
```

2.12 Base64 Encoded Configs

Create Base64 encoded values of the following and keep it handy. You will require them in the next steps.

COGNITO_CLIENT_ID	<BASE_64_ENCODED_VALUE>
COGNITO_CLIENT_SECRET	<BASE_64_ENCODED_VALUE>
COGNITO_ISSUER	<BASE_64_ENCODED_VALUE>

COGNITO_ISSUER format will be
`https://cognito-idp.<REGION>.amazonaws.com/<USER_POOL _ID>`
`Ex: https://cognito-idp.us-east-1.amazonaws.com/us-east-1_sasashYDP`

Note: You can use the following command to create Base64 encoded values

```
echo -n '<STRING_TO_BE_BASE64_ENCODED>' | base64
```

3. Create Platform Secrets in EKS Cluster

Create the below yaml files by substituting the required values.

Note: All the values in secrets should be Base64 encoded during creation.

gravity-mgmt-global-secret.yaml

```
apiVersion: v1
data:
  cognito_client_id: <BASE64_COGNITO_CLIENT_ID>
  cognito_client_secret: <BASE64_COGNITO_CLIENT_SECREt>
  cognito_issuer: <BASE64_COGNITO_ISSUER>
kind: Secret
metadata:
  name: gravity-mgmt-global-secret
  namespace: gravity-system
type: Opaque
```

Use the below commands to create the secrets in the EKS cluster.

```
kubectl create namespace gravity-system
kubectl create -f gravity-mgmt-global-secret.yaml
```

4.Create a new AWS S3 Bucket

Create a new S3 bucket or use any existing s3 bucket for the configuration.
Once bucket is created, note down the **bucket name**

To create a new s3 bucket using AWSCLI follow the link
https://docs.aws.amazon.com/cli/latest/reference/s3api/create-bucket.html

Example command:

```
aws s3api create-bucket --bucket <BUCKET_NAME> --region us-east-1
```

5.Create a new AWS OpenSearch service Domain

Create a new AWS OpenSearch service domain or use the existing AWS OpenSearch service domain.

Once the domain is created, note down the **name** of the domain and **url**

To create a new AWS OpenSearch Service domain using AWSCLI follow the link https://docs.aws.amazon.com/cli/latest/reference/opensearch/

Example command:

```
aws opensearch create-domain \
  --domain-name <DOMAIN_NAME> \
  --engine-version OpenSearch_2.11 \
  --cluster-config InstanceType=<INSTANCE_TYPE>,InstanceCount=<INSTANCE_COUNT> \
  --ebs-options
EBSEnabled=true,VolumeType=<VOLUME_TYPE>,VolumeSize=<VOLUME_SIZE> \
  --node-to-node-encryption-options Enabled=true \
  --encryption-at-rest-options Enabled=true \
  --domain-endpoint-options
EnforceHTTPS=true,TLSSecurityPolicy=Policy-Min-TLS-1-2-2019-07 \
  --advanced-security-options
Enabled=true,InternalUserDatabaseEnabled=true,MasterUserOptions='{MasterUserName
=<MASTER_USERNAME>,MasterUserPassword=<MASTER_PASSWORD>}' \
  --access-policies
'{"Version":"2012-10-17","Statement":[{"Effect":"Allow","Principal":{"AWS":["*"]
},"Action":["es:ESHttp*"],"Resource":"arn:aws:es:<AWS_REGION>:<AWS_ACCOUNT_ID>:d
omain/<DOMAIN_ID>/*"}]}' \
  --region <AWS_REGION>
```

6.Quark deployment in EKS Cluster

Create the below **config.yaml, .env & install.sh** files by replacing the placeholders with appropriate values that you noted down in the prerequisites step.

Please make sure the yaml file's indentation is properly set.

config.yaml

```
version: v1
config:
  email: <PLATFORM_OWNER_EMAIL>
  project: <DEFAULT_PROJECT_NAME>
  aws:
    account_id: <AWS_ACCOUNT_ID>
    region: <AWS_REGION>
  eks:
    name: <CLUSTER_NAME>
    version: "1.27"
    oidc_id: <EKS_OIDC_ID>
  r53:
    zone_id: <ZONE_ID>
    host: <DOMAIN_NAME>
  identityManager:
```

```
    roleARN: <IAM_MANAGER_ROLE_ARN>
  artifacts:
    bucket: <BUCKET_NAME>
  opensearch:
    name: <OPENSEARCH_NAME>
    url: <OPENSEARCH_URL>
```

.env

```
AWS_PROFILE=<AWS_PROFILE>
```

install.sh

Note the version of the Quark platform from Marketplace

Enter version in the placeholder of <PRODUCT_VERSION>

```
#!/bin/bash
set -o allexport; source .env; set +o allexport
wget
https://s3.amazonaws.com/dl.invisibl.io/installer/platform-installer-latest-linu
x_amd64 -O ./platform-installer
chmod +x ./platform-installer
./platform-installer \
        --kubeconfig "eks://<CLUSTER_NAME>?region=<AWS_REGION>" \
        --aws-marketplace \
        --name quarkdemo \
        --product quark \
        --version "<PRODUCT_VERSION>" \
        --config ./config.yaml
```

Perform the installation use the following command

```
./install.sh
```

The installation process of gravity and all other components that quark bootstraps will begin. This would take approximately 5 minutes.

Execute the following command to check the status of the installation. You should see all the Pods created in the “**gravity-system**” namespace “Running”.

```
kubectl get pods -n gravity-system -w
```

Congratulations! Quark is now up and running. You can access the platform at the platform endpoint (**https://*example.com***).