Grey Matter Installation Training

Unstar(S)

These are instructions on how to install Grey Matter on a single EC2 instance for use in service development. They assume you have an AWS account and know how to use it to provision an instance (though nothing precludes the user from adapting these instructions to a similar cloud instance running on, e.g., Azure).

To successfully complete this tutorial you will need

- Access to AWS Management Console with permission to create EC2s.
- AWS credentials in the form of an AWS access key and secret key
- Decipher Docker registry credentials (i.e., to docker.production.deciphernow.com) in the form of a username and password
- SSH or PuTTY (instructions will be given for SSH, and PuTTY users will need to adapt them to their needs)

Launching a machine

Begin by launching a

1. the default Ubuntu Server AMI
2. into a t2.2xlarge
3. with 32 GB storage (the default is 8).

Leave all other parameters set to their defaults, including the default security group. (We will update it later once we know what port to use.)

At time of writing, this can be accomplished by clicking Launch Instance from the EC2 console, selecting the first Ubuntu Server 18.04 LTS AMI and clicking Select, choosing t2.2xlarge and clicking Review and Launch, clicking the “Edit storage” link near the bottom-right, and changing the Size from 8 to 32, clicking Review and Launch again, clicking Launch, and setting up or reusing a keypair. The default security group should already allow SSH, so we can proceed.

Go to Running Instances in the EC2 console, select the instance you just launched, and copy out the IP address. Use this to SSH into your server. The Ubuntu EC2 AMI uses user “ubuntu”.

For example, if the public IP address of your instance is `34.227.100.211`, and you stored your key pair file as `~/.ssh/minikube-aws.pem`, (`chmod 600`) log in with

```
ssh -i ~/.ssh/minikube-aws.pem ubuntu@34.227.100.211
```

We are now ready to install the prerequisites for this tutorial on the EC2 instance.

Installing prerequisites

We will install Docker, kubectl, Helm, and Minikube on our freshly launched EC2 server.

```
sudo apt update

# Docker (and socat, a dependency of helm)
sudo apt install docker.io socat -y

# Kubectl
sudo snap install kubectl --channel=1.6/stable --classic

# Helm
sudo snap install helm --channel=stable --classic

# Minikube
wget https://storage.googleapis.com/minikube/releases/latest/minikube-lin
ux-amd64
chmod +x minikube-linux-amd64
sudo mv minikube-linux-amd64 /usr/local/bin/minikube
```

Minikube setup

Now we start Minikube with enough CPU and memory allocated for a full installation of Grey Matter:

```
sudo minikube start --vm-driver=none --memory 16384 --cpus 6 --kubernetes
-version='v1.15.5'
```

> NOTE: If anything goes wrong, it should be safe to do `sudo minikube delete` to return you to the step right before Minikube was started, ready to retry.

And initialize Helm by installing Tiller into our new Minikube installation:

```
sudo helm init
```

Confirm that Tiller has actually started before moving on. `tiller-deploy` should be `1/1 Running`:

```
sudo kubectl get pods -n kube-system
```

Then setup Voyager (for exposing an ingress port - more on this later).

```
sudo helm repo add appscode https://charts.appscode.com/stable/

sudo helm repo update

sudo helm install appscode/voyager --name voyager-operator --version 10.
0.0 \
  --namespace kube-system \
  --set cloudProvider=minikube \
  --set enableAnalytics=false \
  --set apiserver.enableAdmissionWebhook=false
```

Confirm that Voyager has started before moving on. `voyager-voyager-operator` should be `1/1 Running`:

```
sudo kubectl get pods -n kube-system
```

Grey Matter configuration

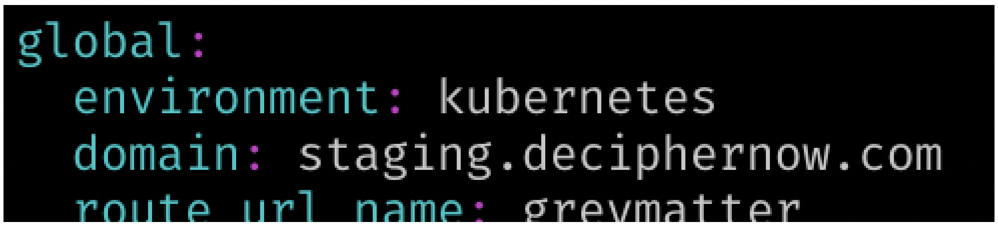
Helm overrides

Next you will make yourself custom Helm configuration overrides using two Decipher templates. Run `wget` from your EC2 instance to get the templates:

```
wget https://raw.githubusercontent.com/DecipherNow/helm-charts/release-2.
0/greymatter.yaml
```

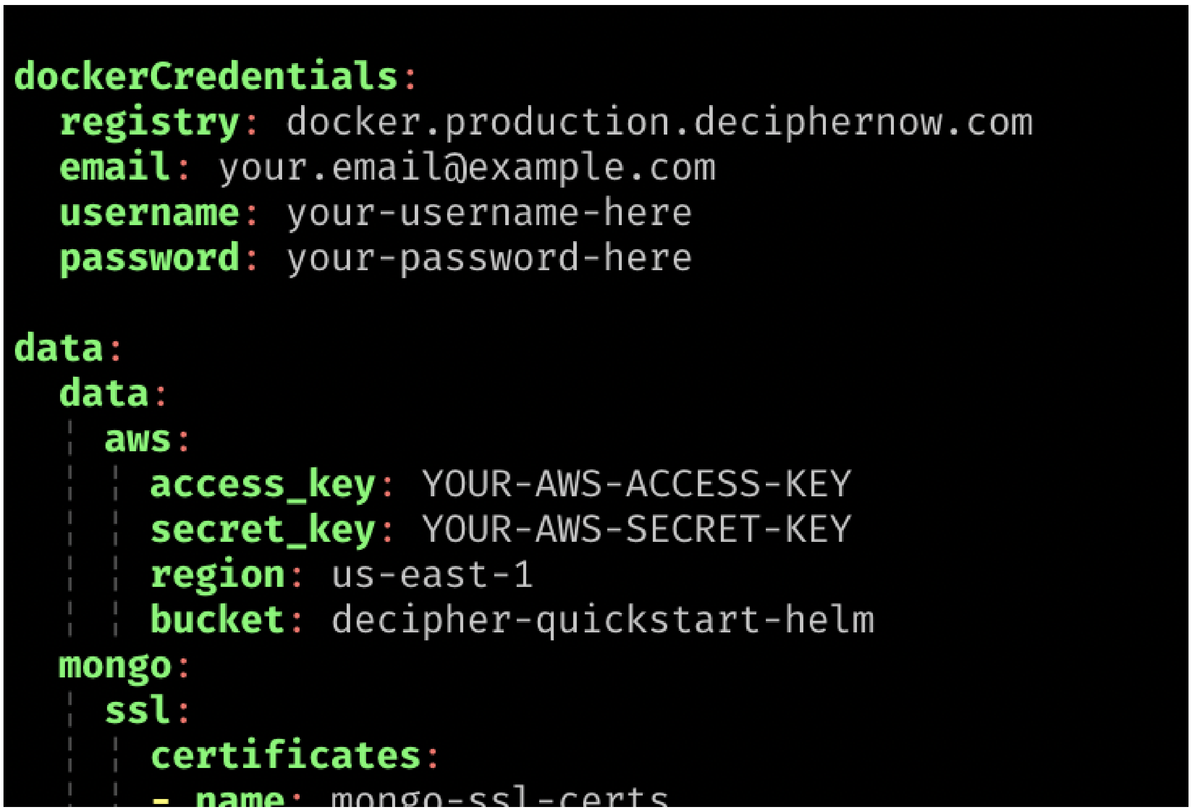
```
wget https://raw.githubusercontent.com/DecipherNow/helm-charts/release-2.
0/greymatter-secrets.yaml
```

- greymatter.yaml contains many Grey Matter options you may want to adjust some day, but our change is small: Simply replace the value of `global.environment` with “kubernetes”, as in this screenshot.



```
global:
  environment: kubernetes
  domain: staging.deciphernow.com
  route_url_name: greymatter
```

- greymatter-secrets.yaml contains secrets, passwords, and certificates. For our purposes here, just fill in your Docker registry credentials and AWS credentials, like so:



```
dockerCredentials:
  registry: docker.production.deciphernow.com
  email: your.email@example.com
  username: your-username-here
  password: your-password-here

data:
  data:
    aws:
      access_key: YOUR-AWS-ACCESS-KEY
      secret_key: YOUR-AWS-SECRET-KEY
      region: us-east-1
      bucket: decipher-quickstart-helm
  mongo:
    ssl:
      certificates:
      - name: mongo-ssl-certs
```

Save your changes to these two files and proceed.

Decipher Helm repository

We’re now going to add another Helm repository, this time for Grey Matter itself, and use it to install the latest release. You will need your Docker registry username and password again.

```
sudo helm repo add decipher https://nexus.production.deciphernow.com/repo
sitory/helm-hosted --username 'YOUR USERNAME' --password 'YOUR PASSWORD'

sudo helm repo update
```

Grey Matter installation

And now, the moment we've all been waiting for… We will use the two configuration override files we created earlier to install Grey Matter. We'll start with the `--dry-run` flag to check our configs *only*, then remove it to do the actual installation.

```
sudo helm install decipher/greymatter -f greymatter.yaml -f greymatter-se
crets.yaml --name gm --version 2.0.4 --dry-run
```

If you see no errors with `--dry-run`, and all you see is `NAME: gm`, then it's safe to remove `--dry-run` flag and re-run. Do so now.

Grey Matter should now be starting. This takes a while to stabilize, and you can monitor the process with

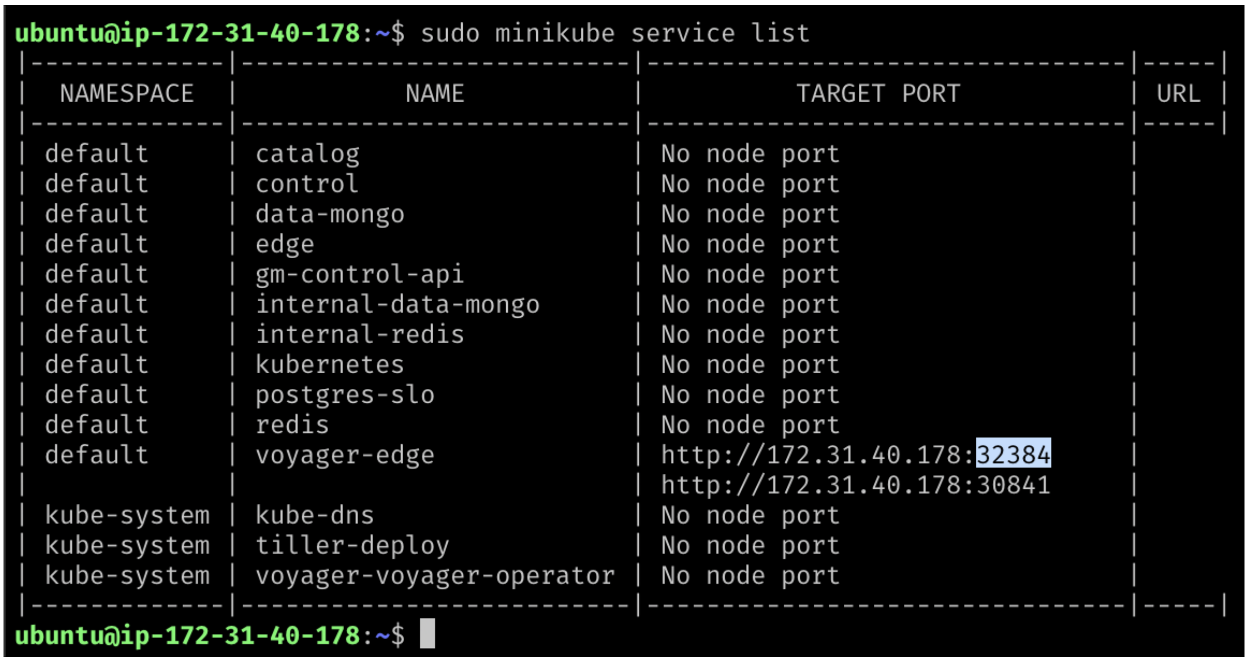
```
watch sudo kubectl get pods
# Ctrl-C to escape
```

Don't worry if you see `Error` and `CrashLoopBackOff`. This is normal, as each component of the system retries until its dependencies are available. `catalog-init` especially takes a while to execute without error.

Once everything is either `Running` or `Completed`, Grey Matter is up and we have only to open the ingress port before we can connect from the outside. You can discover the ingress port with

```
sudo minikube service list
```

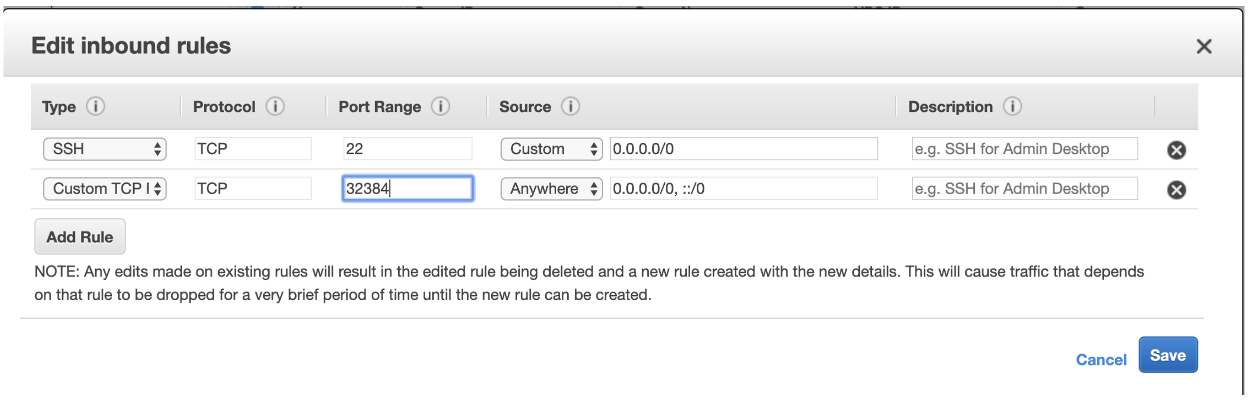
which should result in something like the following. Copy the first `voyager-edge` port so we can add it to the AWS Security Group for our EC2 instance.



```
ubuntu@ip-172-31-40-178:~$ sudo minikube service list
|-------------|--------------------------|--------------------------------|-----|
|  NAMESPACE  |           NAME           |          TARGET PORT           | URL |
|-------------|--------------------------|--------------------------------|-----|
| default     | catalog                  | No node port                   |
| default     | control                  | No node port                   |
| default     | data-mongo               | No node port                   |
| default     | edge                     | No node port                   |
| default     | gm-control-api           | No node port                   |
| default     | internal-data-mongo      | No node port                   |
| default     | internal-redis           | No node port                   |
| default     | kubernetes               | No node port                   |
| default     | postgres-slo             | No node port                   |
| default     | redis                    | No node port                   |
| default     | voyager-edge             | http://172.31.40.178:32384     |
|             |                          | http://172.31.40.178:30841     |
| kube-system | kube-dns                 | No node port                   |
| kube-system | tiller-deploy            | No node port                   |
| kube-system | voyager-voyager-operator | No node port                   |
|-------------|--------------------------|--------------------------------|-----|
ubuntu@ip-172-31-40-178:~$
```

> NOTE: This is the *port* you'll need, but the IP address listed there is your Minikube-internal IP address, and won't work for access from the outside. Replace this with your EC2's public IP to access Grey Matter from the browser.

Now add an inbound TCP rule for that port to the security group for your EC2. This is done in the AWS console.



Edit inbound rules

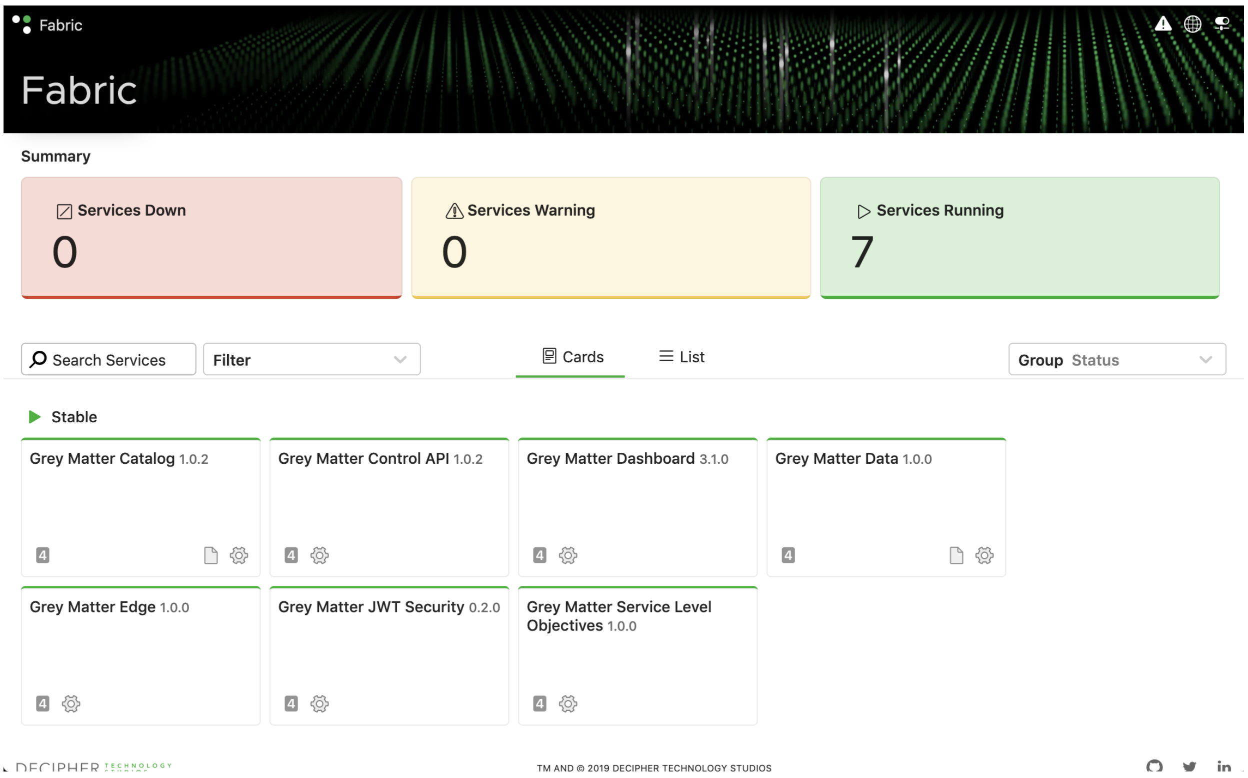
Type	Protocol	Port Range	Source		Description
SSH	TCP	22	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
Custom TCP I	TCP	32384	Anywhere	0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop

NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic that depends on that rule to be dropped for a very brief period of time until the new rule can be created.

The last bit of setup necessary is to add the Grey Matter quickstart certificate to your *local machine's* browser/keychain. The exact steps differ between operating systems and browsers, but for Chrome you should be able to simply download `quickstart.zip` from here, extract the certificates, and double-click on `quickstart.p12` to import it into your OS' keychain. The password is `password`.

> NOTE: In actual production deployments, you would be replacing the entire PKI certificate setup (in greymatter-secrets.yaml) with a secure one. The provided quickstart certificates are for demonstration purposes only.

Finally, navigate to `https://{your-ec2-public-ip}:{voyager-edge-port}`. Your browser will complain that the certificate is untrusted, but you may safely ignore that for now. Proceed with temporarily trusting the quickstart self-signed certificates. You should see your very own instances of the Grey Matter Intel 360 Dashboard, showing the seven core Grey Matter services.



Fabric

Fabric

Summary

Services Down
0

Services Warning
0

Services Running
7

Search Services | Filter | Cards | List | Group Status

Stable

Grey Matter Catalog 1.0.2	Grey Matter Control API 1.0.2	Grey Matter Dashboard 3.1.0	Grey Matter Data 1.0.0
4	4	4	4
Grey Matter Edge 1.0.0	Grey Matter JWT Security 0.2.0	Grey Matter Service Level Objectives 1.0.0	
4	4	4	

DECIPHER TECHNOLOGY

TM AND © 2019 DECIPHER TECHNOLOGY STUDIOS

Verifying the deployment

Although all services show green, it is always a good idea at this point to confirm the integrity of the deployment with test requests. Decipher maintains a test suite for this purpose that exercises the various components of the system, in the form of exported Postman collections, run with Postman's headless CLI, Newman. These can be run from the `docker.production.deciphernow.com/deciphernow/gm-integrity:latest` container, which is configured through environment variables.

Since the Grey Matter Integrity project defaults to the quickstart user certificate, to run all tests against your quickstart environment you need only pass it the URL:

```
sudo docker login docker.production.deciphernow.com

sudo docker run --rm -e URL={your-ec2-public-ip}:{port} \
docker.production.deciphernow.com/deciphernow/gm-integrity
```

Troubleshooting a deployment

This section covers some of the tools available for probing a misbehaving deployment.

> `TODO` : Expand and combine with service deployment troubleshooting