MICRO-SERVICES SAMPLE PROJECT

Components Document & Tools Usage

VIJAYENDRA MUDIGAL
https://vijayendra.mudigal.com/

TABLE OF CONTENTS

API GATEWAY

Netflix Zuul is a the reverse proxy server which acts as the API Gateway for accessing the micro services behind the gateway which routes the request to the respective service. Microservice's stay behind reverse proxy server and needs to be consumed via api gateway. The api-gateway micro service with docker profile runs on port 8080 and can be accessed by http://localhost:8080 .

Configuration done in API Gateway for Routing:

```
zuul:
  ignoredServices: '*'
  routes:
    one:
      path: /service-one/**
      serviceId: Service-One
    two:
      path: /service-two/**
      serviceId: Service-Two
```

SERVICE REGISTRATION AND DISCOVERY

Registration and discovery is taken care by the HashiCorp's Consul. During the startup of the individual services, they register with service registration service those details such as host name, port etc. by which the services can be accessed. Once the service is registered to the consul, consul checks for the health of the service by sending a heartbeat for the health check path and health check interval that has been registered with Consul. Requests to the micro-services has to be routed through api-gateway during with the api-gateway contacts discovery service to get the information required to send the request to the intended microservice.

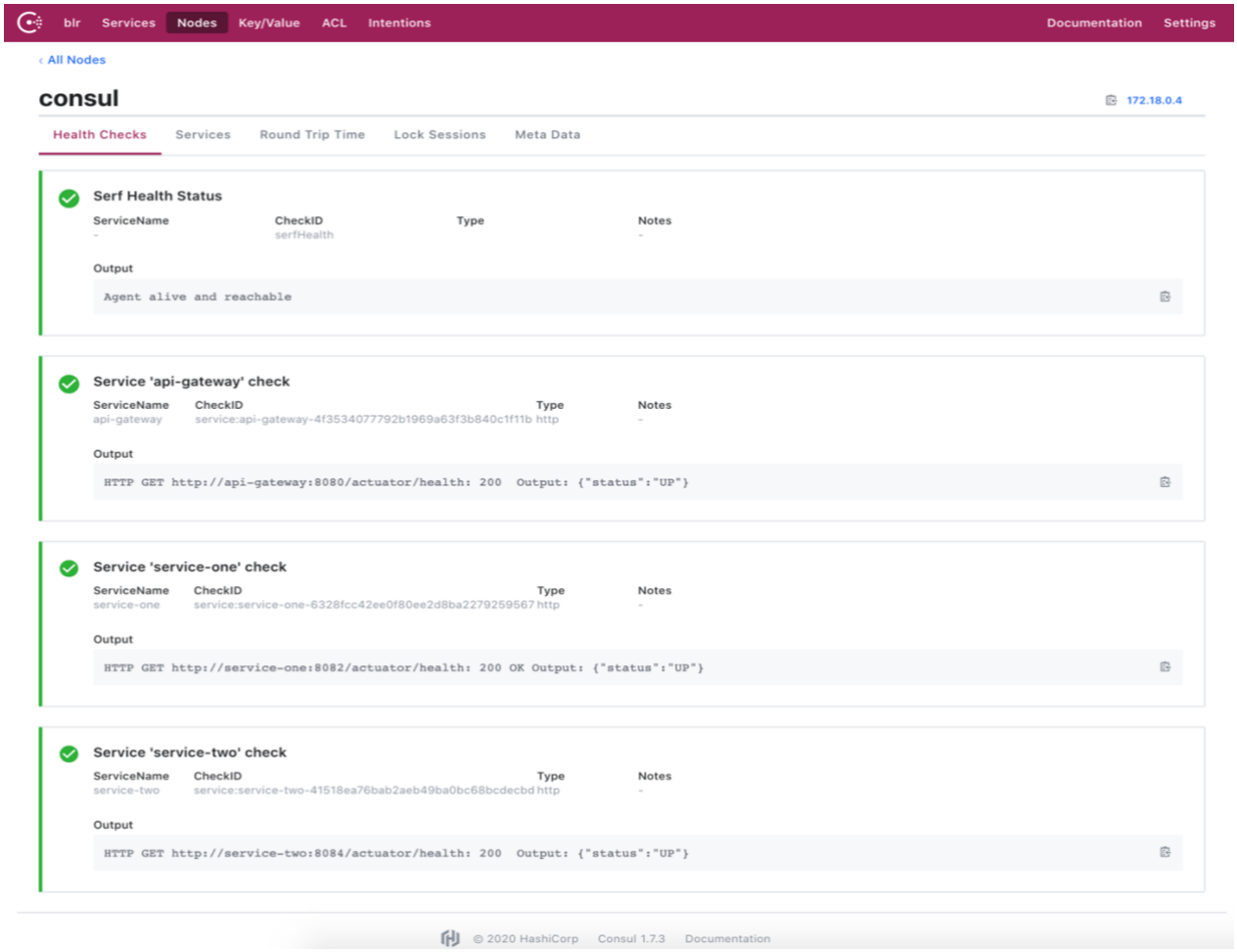
Configuration done in micro services to register to Consul:

```
management:
  contextPath: /actuator

spring:
  application.name: service-one
  cloud:
    consul:
      host: consul
      port: 8500
      discovery:
        hostName: service-one
        instanceId:${spring.application.name}:${spring.application.i
nstance_id:${random.value}}
        healthCheckPath: ${management.contextPath}/health
        healthCheckInterval: 15s
```

Tools:

Consul Management console: http://localhost:8500/ui/

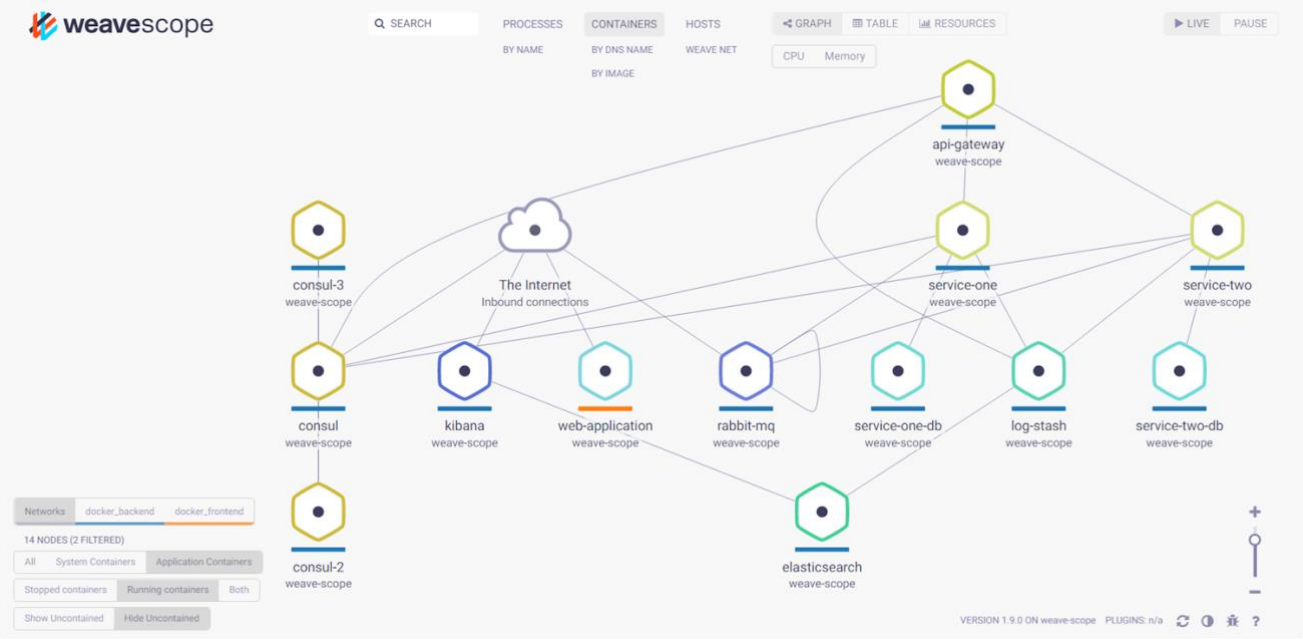


MONITORING AND VIZUALIZATION

Monitoring, visualisation & management of the container in docker is done by weave scope.

Tools:

Weavescope Management Console: http://localhost:4040/



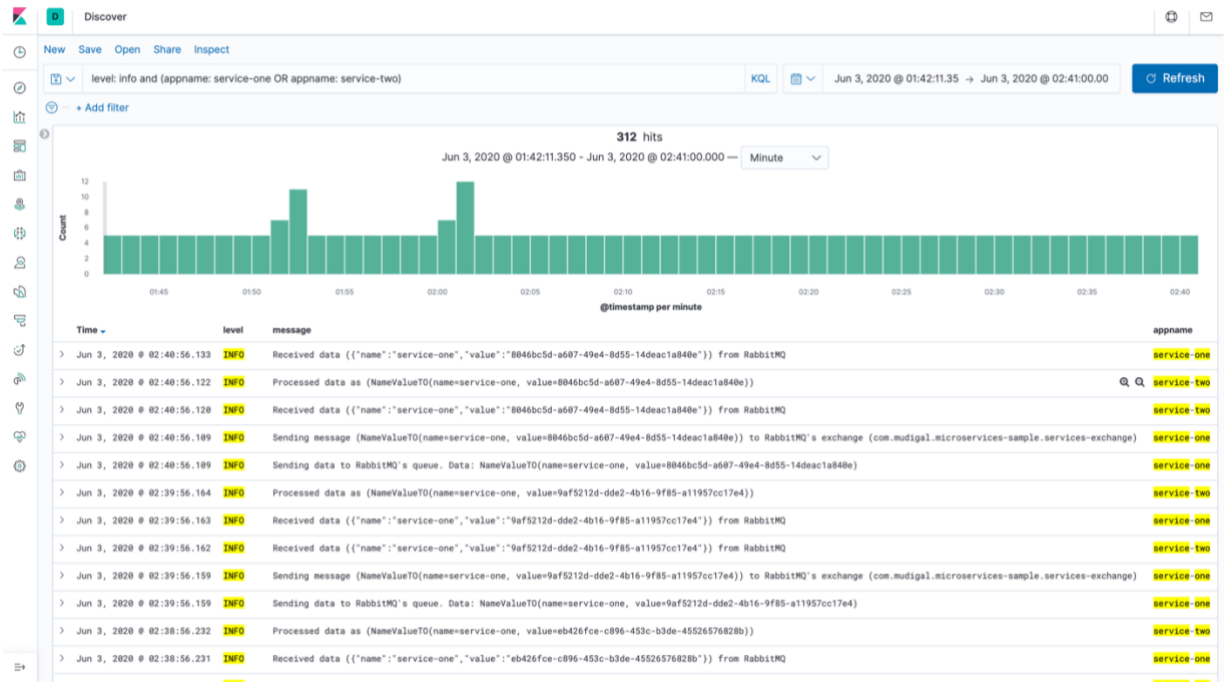
CENTRALIZED LOGGING USING ELK

Our services use Logback to create application logs and send the log data to the logging server (Logstash). Logstash formats the data and send it to the indexing server (Elasticsearch). The data stored in elasticsearch server can be beautifully visualized using Kibana.

Tools:

Elasticsearch: http://localhost:9200/_search?pretty

Kibana: http://localhost:5601/app/kibana

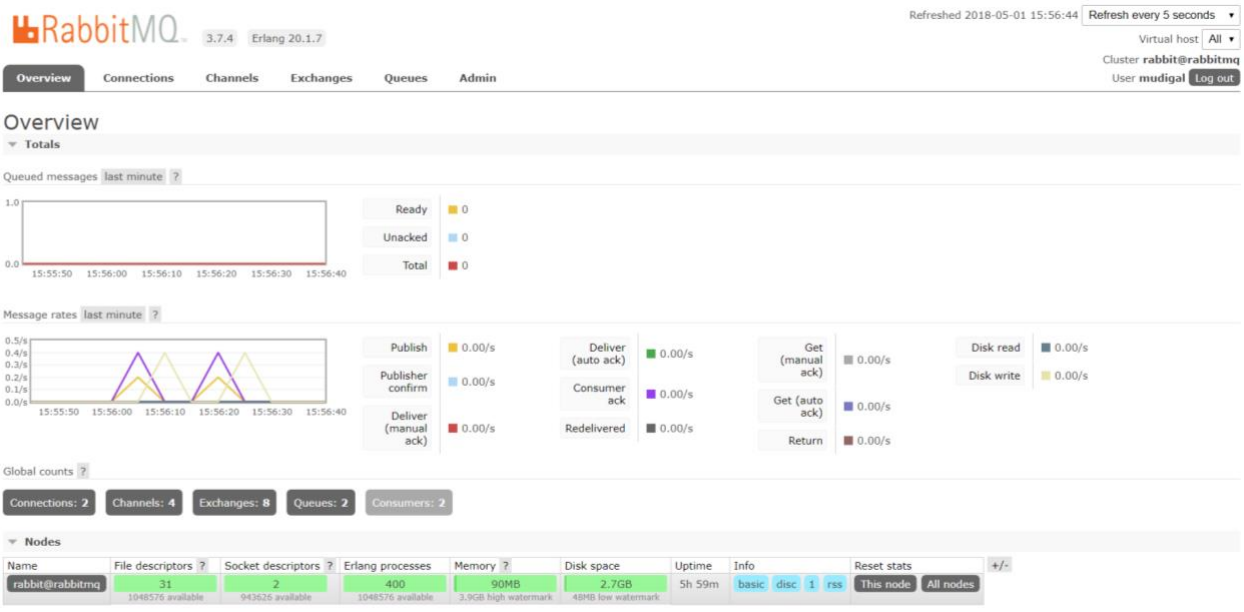


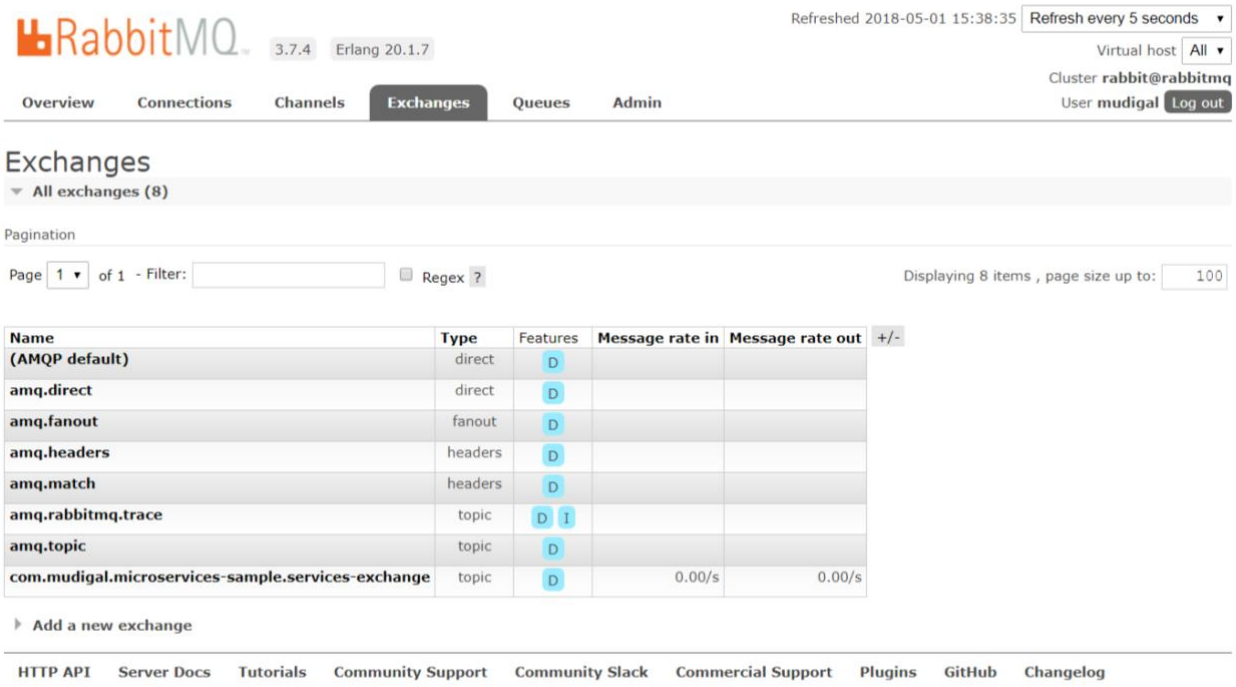
MICROSERVICES COMMUNICATION

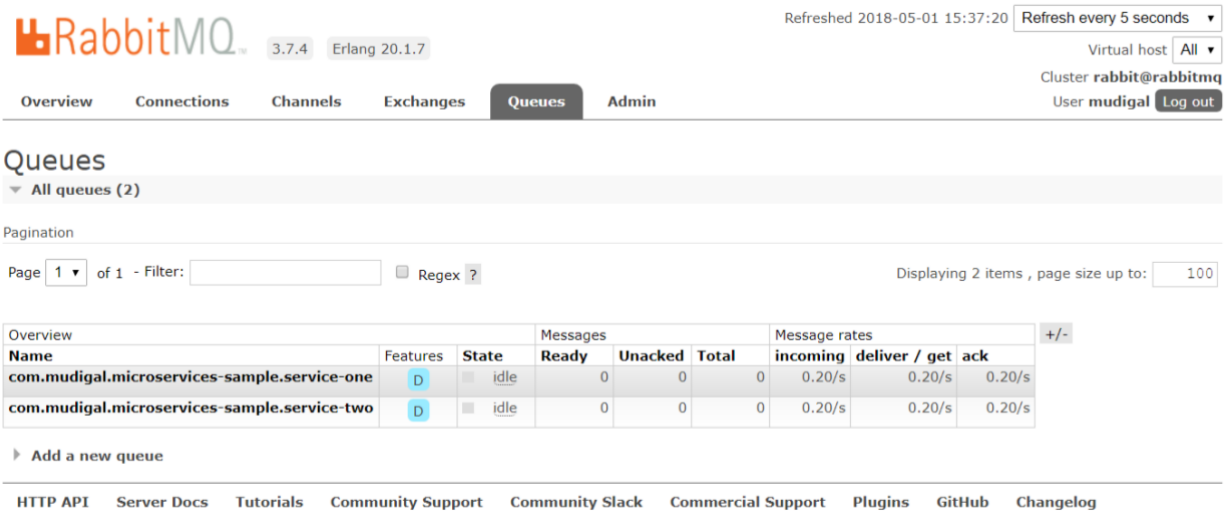
Intercommunication between microservices happens asynchronously with the help of RabbitMQ.

Tools:

RabbitMQ Management Console: http://localhost:15672/




RabbitMQ 3.7.4 Erlang 20.1.7
Refreshed 2018-05-01 15:38:35 Refresh every 5 seconds
Virtual host All
Cluster rabbit@rabbitmq
User mudigal Log out
Overview Connections Channels Exchanges Queues Admin
Exchanges
All exchanges (8)
Pagination
Page 1 of 1 - Filter: Regex ?
Displaying 8 items , page size up to: 100
Name Type Features Message rate in Message rate out +/-
(AMQP default) direct D
amq.direct direct D
amq.fanout fanout D
amq.headers headers D
amq.match headers D
amq.rabbitmq.trace topic D I
amq.topic topic D
com.mudigal.microservices-sample.services-exchange topic D 0.00/s 0.00/s
Add a new exchange
HTTP API Server Docs Tutorials Community Support Community Slack Commercial Support Plugins GitHub Changelog


RabbitMQ 3.7.4 Erlang 20.1.7
Refreshed 2018-05-01 15:37:20 Refresh every 5 seconds
Virtual host All
Cluster rabbit@rabbitmq
User mudigal Log out
Overview Connections Channels Exchanges Queues Admin
Queues
All queues (2)
Pagination
Page 1 of 1 - Filter: Regex ?
Displaying 2 items , page size up to: 100
Overview Messages Message rates +/-
Name Features State Ready Unacked Total incoming deliver / get ack
com.mudigal.microservices-sample.service-one D idle 0 0 0 0.20/s 0.20/s 0.20/s
com.mudigal.microservices-sample.service-two D idle 0 0 0 0.20/s 0.20/s 0.20/s
Add a new queue
HTTP API Server Docs Tutorials Community Support Community Slack Commercial Support Plugins GitHub Changelog