Be a Microservices Hero | ContainerCon’15
Dragos Dascalita Haut | Adobe

https://github.com/adobe-apiplatform

Presentation scripts: https://gist.github.com/ddragosd/608bf8d3d13e3f688874
A Creative Cloud Microservice Sample: Content-Aware fill
A CreativeCloud Microservice Sample: send to Photoshop from the mobile device
Lifecycle of a microservice
Growing the microservices ecosystem
API Platform

- Security
- App & User Authorization
- Analytics
- Throttling & Rate Limiting
How to make it easier to scale Microservices in this model?
Microservices, Containers & Apache Mesos

write
write
write
containerize
containerize
containerize
&
&
&
deploy
deploy
deploy

Java
node.js

Docker
Mesos
Apache Mesos
Apache Mesos & Microservices

“Program for the data center Just like you program for the OS”

Computer: Data Center
Kernel: Mesos
OS: Mesos Frameworks, Marathon, Mesosphere’s DCOS
Services: Microservices
Traffic Ctrl: API Gateway
  • Facilitates inter-API communication
  • Routing, Traffic Shaping, Filtering
#DEMO
Apache Mesos & Microservices

SETUP A MINI-DATA-CENTER
4 VMs, 1 Leader, 2 Slaves, 1 Admin
Simple ways to get started:
- https://mesosphere.com/product/
- https://open.mesosphere.com/getting-started/datacenter/install/

Setup a Mini-Data-Center
4 VMs, 1 Leader, 2 Slaves, 1 Admin
#DEMO
Apache Mesos & Microservices

THE "KERNEL": MESOS
3 VMs, 1 Leader
THE “OS” : MARATHON

Apache Mesos & Microservices

(it will start our microservices)
START A "SERVICE" IN THE "OS":
Hello-world microservice

```
curl "http://<marathon_url>/v2/apps" 
-H "Content-Type: application/json" 
-H "Accept:application/json"
--data @/tmp/hello_world_app.json
```
START A “SERVICE” IN THE “OS”:
hello-world microservice

Apache Mesos & Microservices
#DEMO

Apache Mesos & Microservices

THE “TRAFFIC CTRL”: API GATEWAY
- DESIGN PRINCIPLES -

Simple
Scalable
Secure
Super fast
The "Traffic Ctrl": API Gateway

Apache Mesos & Microservices

#DEMO

**OPENRESTY**
- Nginx Lua Module
- Nginx Redis
- Headers more
- Set misc
- LuaJIT
- .....

**NGINX**
- Upstream
- HTTP Proxy
- PCRE
- SSL
- .....

**Custom Modules**
- NAXSI – WAF
- api-gateway request-validation
- api-gateway-async-logger
# DEMO

Apache Mesos & Microservices

THE "TRAFFIC CTRL" : API GATEWAY

Simple configuration blending in Nginx configuration

server {
  listen 80;
  server_name hello-world.api.container-con.org;

  location / {
    # -----------------------------------------
    # Specify what to validate for this location
    # -----------------------------------------
    set $validate_api_key on;
    set $validate_oauth_token on;
    set $validate_user_profile on;
    set $validate_service_plan on;
    ... 
    # -----------------------------------------
    # Proxy the request to the actual microservice
    # -----------------------------------------
    proxy_pass $microservice$request_uri;
  }
}
#DEMO

Apache Mesos & Microservices

THE "TRAFFIC CTRL" : API GATEWAY

Service Discovery Example

```bash
curl -s ${marathon_host}/v2/tasks -H "Accept:text/plain" | \ awk 'NF>2' | grep -v :0 | awk '!seen[$1]++' | \ awk '{s=""; for (f=3; f<=NF; f++) s = s "\n server " $f ";" ; print "upstream "$1 "{" s "\n keepalive 16;" }"
```
Apache Mesos & Microservices

#DEMO

THE “TRAFFIC CTRL”: API GATEWAY
(OPENRESTY & NGINX based)

```yaml
server {
  listen 80;

  # listens on <app_name>.api.any.domain with the assumption that the app_name has been defined in marathon.
  server_name ~^(?<app_name>.[^\./]+).api\.(?<domain>\.+);

location / {
  proxy_connect_timeout 10s;  # timeout for establishing a connection with a proxied server
  proxy_read_timeout 10s;    # Defines a timeout for reading a response from the proxied server.
      # The timeout is set only between two successive read operations,
      # not for the transmission of the whole response.
  proxy_send_timeout 10s;    # Sets a timeout for transmitting a request to the proxied server.
      # The timeout is set only between two successive write operations,
      # not for the transmission of the whole request.
  keepalive_timeout 10s;    # timeout during which a keep-alive client connection stays open on the server side
  proxy_buffering off;    # enables or disables buffering of responses from the proxied server.
  proxy_http_version 1.1;  # Version 1.1 is recommended for use with keepalive connections.
  proxy_set_header Connection "";

  proxy_pass http://$app_name$request_uri;
}
```
#DEMO

Apache Mesos & Microservices

THE "TRAFFIC CTRL" : API GATEWAY DEPLOYMENT

curl -X POST -H "Content-Type:application/json" ${MARATHON_HOST}/v2/apps?force=true --data '  
{
  "id": "api-gateway",
  "container": {
    "type": "DOCKER",
    "docker": {
      "image": "adobeapiplatform/apigateway:latest",
      "forcePullImage": true,
      "network": "HOST"
    }
  },
  "cpus": 4,
  "mem": 4028.0,
  "env": { "MARATHON_HOST": "http://<marathon_host>" },
  "acceptedResourceRoles": ["slave_public"],
  "constraints": [ [ "hostname", "UNIQUE" ] ],
  "ports": [ 80 ],
  "instances": 1
}
curl -X POST -H "Content-Type:application/json" ${MARATHON_HOST}/v2/apps?force=true --data ' 

{ 
  "id": "api-gateway",
  "container": {  
    "type": "DOCKER",
    "docker": {  
      "image": "apiplatform/apigateway:latest",
      "forcePullImage": true,
      "network": "HOST"
    }
  },
  "cpus": 4,
  "mem": 4028.0,
  "env": { "MARATHON_HOST": "http://<marathon_host>" },
  "acceptedResourceRoles": ["slave_public"],
  "constraints": [ [ "hostname", "UNIQUE" ] ],
  "ports": [ 80 ],
  "instances": 1,
  "healthChecks": [
    {
      "protocol": "HTTP",
      "portIndex": 0,
      "path": "/health-check",
      "gracePeriodSeconds": 3,
      "intervalSeconds": 10,
      "timeoutSeconds": 10
    }
  ]
} 

Optionally you can include health-check
#DEMO

Apache Mesos & Microservices

• Demo: API KEY Management
#DEMO
Apache Mesos & Microservices

THE "TRAFFIC CTRL" : API GATEWAY
API KEY Management with Redis

curl -X POST -H "Content-Type:application/json" ${MARATHON_HOST}/v2/apps?force=true --data '
{
    "id": "api-gateway-redis",
    "container": {
        "type": "DOCKER",
        "docker": {
            "image": "redis:latest",
            "forcePullImage": true,
            "network": "HOST"
        }
    },
    "cpus": 0.5,
    "mem": 1024.0,
    "constraints": [ [ "hostname", "UNIQUE" ] ],
    "ports": [ 6379 ],
    "instances": 1
}'
Create a new Vhost for `server_name -hello-world.api.(?<domain>.+)`;

set $marathon_app_name hello-world;

location / {
    # identify the service
    set $service_id $marathon_app_name;

    # identify the api key
    #     either from the query params or from the "Api-Key" header
    set $api_key $arg_api_key;
    set_if_empty $api_key $http_x_api_key;

    # add the api-key validator
    set $validate_api_key on;

    # validate request
    access_by_lua "ngx.apiGateway.validation.validateRequest()";

    proxy_pass http://$marathon_app_name$request_uri;
# DEMO

Apache Mesos & Microservices

THE “TRAFFIC CTRL” : API GATEWAY

Protect a service with API-KEY

# ADD an API-KEY for the HELLO-WORLD service
# NOTE: this API SHOULD not be exposed publicly

curl -X POST "http://api-gateway.${API_DOMAIN}/cache/api_key?key=key-1&
app_name=app-1&
service_id=hello-world&
service_name=hello-world&
consumer_org_name=demo-consumer"

# update hello-world microservice to require an API-KEY

curl "http://hello-world.${API_DOMAIN}/hello"
# {"error_code":403000,"message":"Api Key is required"}

# make another call including the api-key

curl "http://hello-world.${API_DOMAIN}/hello" -H "X-Api-Key:key-1"
#DEMO
Apache Mesos & Microservices

THE "TRAFFIC CTRL" : API GATEWAY
Capture Analytics

• Demo: Analytics using Graphite and Grafana
set $marathon_app_name hello-world;

location / {
  ...
  
  proxy_pass http://$marathon_app_name$request_uri;

  ...

  # capture usage data
  log_by_lua '    
  if ( ngx.apiGateway.metrics ~= nil ) then
    ngx.apiGateway.metrics.captureUsageData()
  end
  ',';

}
curl -X POST -H "Content-Type:application/json" ${MARATHON_HOST}/v2/apps?force=true --data ' {
  "id": "api-gateway-graphite",
  "container": {
    "type": "DOCKER",
    "docker": {
      "image": "hopsoft/graphite-statsd:latest",
      "forcePullImage": true,
      "network": "BRIDGE",
      "portMappings": [
        {
          "containerPort": 80, "hostPort": 0, "protocol": "tcp"
        },
        {
          "containerPort": 8125, "hostPort": 8125, "protocol": "udp"
        }
      ]
    }
  },
  "cpus": 2,
  "mem": 4096.0,
  "instances": 1
}'}
# DEMO
Apache Mesos & Microservices

THE "TRAFFIC CTRL" : API GATEWAY Capture Analytics

# verify that the Graphite instance is up by accessing it through the API Gateway
```
curl "http://api-gateway-graphite.${API_DOMAIN}/render/?from=-5min&format=raw&target=carbon.aggregator.*.metricsReceived"
```

# to open Graphite in a browser
```
python -mwebbrowser "http://api-gateway-graphite.${API_DOMAIN}/"
```

# generate traffic for the hello-world service in order to capture metrics
docker run jordi/ab ab -k -n 10000 -c 500 "http://hello-world.${API_DOMAIN}/hello?api_key=key-1"

# then check the Graphite stats in the browser
```
```
# DEMO

Apache Mesos & Microservices

THE "TRAFFIC CTRL": API GATEWAY
View Analytics with Grafana

curl -X POST -H "Content-Type: application/json" ${MARATHON_HOST}/v2/apps?force=true --data '{
  "id": "api-gateway-grafana",
  "container": {
    "type": "DOCKER",
    "docker": {
      "image": "grafana/grafana:latest",
      "forcePullImage": true,
      "network": "BRIDGE",
      "portMappings": [ { "containerPort": 3000, "hostPort": 0, "protocol": "tcp" } ]
    }
  },
  "cpus": 1,
  "mem": 2048.0,
  "instances": 1
}'
#DEMO

Apache Mesos & Microservices

THE "TRAFFIC CTRL" : API GATEWAY

View Analytics with Grafana
API Gateway is Open Source
https://github.com/adobe-apiplatform/apigateway

Gist script used in this presentation:
https://gist.github.com/ddragosd/608bf8d3d13e3f688874