Quo Vadis Netflix Stack?

// Baris Cubukcuoglu & Fabian Keller
Baris Cubukcuoglu
Software Engineer
Photographer
@bariscubuk__

Fabian Keller
Software Engineer
Woodworker
@_fabiankeller
Netflix OSS – Well known Projects

Zuul
API Gateway

Eureka
Service Registry

Archaius
Centralized Config

Microservice A
Hystrix
Ribbon
Feign

Microservice B
Hystrix
Ribbon
Feign
Eureka, Feign & Ribbon (with Spring Cloud)

```java
@SpringBootApplication
@EnableFeignClients
@RestController
public class MomentsApplication {

    public static void main(String[] args) {
        SpringApplication.run(MomentsApplication.class, args);
    }

    @Autowired
    private PhotoClient photoClient;

    @GetMapping("/moments/{ownerId}")
    public Photo moments(@PathVariable("ownerId") UUID ownerId) {
        return photoClient.getPhoto(ownerId);
    }
}
```
Hystrix (with Spring Cloud)

```java
@SpringBootApplication
@RestController
@EnableCircuitBreaker
public class MomentsApplication {

    public static void main(String[] args) {
        SpringApplication.run(MomentsApplication.class, args);
    }

    @GetMapping("/moments/{ownerId}"
    @HystrixCommand(fallbackMethod = "getCachedResult")
    public EmbeddedResponse moments(@PathVariable("ownerId") UUID ownerID) {
        // do something fragile
        return response;
    }

    public EmbeddedResponse getcachedResult(UUID ownerID) {
        // load cached data
        return response;
    }
}
```
Zuul (with Spring Cloud)

```yaml
zuul:
  routes:
    echo:
      url: http://localhost:8080/moments/**
      path: /moments/**
```

Archaius (with Spring Cloud)

```java
@Component
public class MomentsService {

    private DynamicStringProperty property = DynamicPropertyFactory.getInstance()
        .getStringProperty("de.fakeller.api.endpoint", "not found!");

    public String getValue() {
        return property.get();
    }
}
```
Maintenance Mode
Which Projects are affected?

- Archaius
- Centralized Config
- Zuul
- API Gateway
- Eureka
- Service Registry
- Hystrix
- Ribbon
- Feign
- Microservice A
- Microservice B

**Maintenance Mode**

2019-07-04 / Java Forum Stuttgart / Baris Cubukcuoglu & Fabian Keller
Are there any alternatives?
TL;DR: yes!

For Hystrix:
Resilience4j, Sentinel, Spring Retry

For Hystrix Dashboard / Turbine:
Micrometer + Monitoring System (e.g. Prometheus etc.)

For Archaius:
Spring Cloud Config, HashiCorp Vault

For Ribbon:
Spring Cloud Load Balancer, gRPC
Fallacies of Distributed Computing
see https://bit.ly/1IEpFC0

- The network is reliable
- Latency is zero
- Bandwidth is infinite
- The network is secure
- Topology doesn’t change
- There is one administrator
- Transport cost is zero
- The network is homogeneous
Patterns

• Stability
• Capability
• Transparency
NETFLIX
15% Internet Traffic
Service Discovery

- Client
  - Lookup
  - Register
  - Load balance
- Registry
- Service A
- Service B
Load Balancing

- **Round Robin**
- **Availability Filtering**
- **Weighted Response Time**

**Filters:**
- Service
- Service

- Service 0.7
  - Service 0.3
Circuit Breaker

Execute Command

Fallback

Run
Bulkhead
Bulkhead
API Gateway
API Gateway

Client -> API Gateway -> Resource
Resource
Resource
Resource
The Modern Cloud Stack

Photo by Pero Kalimero on Unsplash
Where do we begin with?

A homogeneous microservice landscape, yeah right...

Netflix OSS

Java

Netflix OSS

Java

Netflix OSS

Java

Netflix OSS

Java

Netflix OSS

node.js

?!
So how do we solve all these challenges?
Using sidecars instead of libraries

Application Layer
Cross-cutting concerns

Infrastructure Layer
Cross-cutting concerns

Application
Netfix OSS

Ingress
Application
Egress
Adding the sidecars to all containers

We can have the exact same architecture as before
Cloud Foundry
What is Cloud Foundry?

Open Source  Faster to iterate  Scalable platform

cf push -p target/spring-music.jar
Cloud Foundry Service Discovery

a.k.a. Container Networking. Also enables client-side load balancing via DNS
Cloud Foundry Service Discovery

How to setup CF container network

```bash
# create moments microservice
cf push moments -p moments.jar --no-route --no-start
cf map-route moments apps.internal --hostname moments
cf set-env moments MEDIA_API "http://media.apps.internal:8080/"
cf start moments

# create media microservice
cf push media -p media.jar --no-route
cf map-route media apps.internal --hostname media

# enable container networking
 cf add-network-policy moments --destination-app media --port 8080 --protocol tcp
```
Cloud Foundry Route Services
Taking care of ingress traffic

- Security
- Rate Limiting
- Caching
- Tracing
Cloud Foundry Dynamic App Config

CF SET-ENV

CF RESTART
Kubernetes Service Discovery

Deployments and Services

Environment

Cluster IP 10.254.40.156

DNS? moments
A 10.254.40.148

moments 172.10.40.156

media 172.10.40.148

moments 172.10.40.148

media 172.10.40.123

media 172.10.40.42
Kubernetes Service Discovery
Exposing a service

Kind: k8s/Service
moments.my-namespace.svc.cluster.local
10.98.61.186

Kind: k8s/Deployment
Pod 1: 172.10.40.148 moments-pod
Pod 2: 172.10.40.156 moments-pod

new RestTemplate().exchange("http://moments/api/v1/timeline", ...)

2019-07-04 / Java Forum Stuttgart / Baris Cubukcuoglu & Fabian Keller
Istio Architecture
Control Plane and Data Plane

Control Plane
- Pilot
- Citadel
- Mixer

Data Plane
- Pod 1
  - Envoy
  - App 1
- Pod 2
  - Envoy
  - App 2
Istio VirtualService
Adding a VirtualService layer to intelligently route traffic

Kind: istio/VirtualService
moments
Match routing rules:
/api/v1 → moments-v1
/api/v2 → moments-v2

Kind: k8s/Service
moments-v1
10.98.61.186

Kind: k8s/Service
moments-v2
10.98.61.193

Kind: k8s/Deployment
Pod 1
moments-v1-pod

Kind: k8s/Deployment
Pod 2
moments-v1-pod

Kind: k8s/Deployment
Pod 1
moments-v2-pod
Istio Destination Rules
Destination rules apply after routing rules are evaluated

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: httpbin
spec:
  host: httpbin
  trafficPolicy:
    connectionPool:
      tcp:
        maxConnections: 20
    http:
      http1MaxPendingRequests: 20
      maxRequestsPerConnection: 20
    outlierDetection:
      consecutiveErrors: 5
      interval: 10s
      baseEjectionTime: 3m
      maxEjectionPercent: 100
```

Apply various policies to traffic:

- Load Balancer Policy (RR, Random)
- Client TLS
- Circuit Breaker
K8s/Istio Dynamic App Config

VIM KUBE-APP.YAML
KUBECTL APPLY
Istio Additional Features
This was just the beginning...

- Automatic Retries
- Rate Limiting
- mTLS
- Policy Enforcement
- Observability & Monitoring
- Distributed Tracing
- Fault Injection
## Platform Support

<table>
<thead>
<tr>
<th></th>
<th>Eureka</th>
<th>Container Networking</th>
<th>Kubernetes Services</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Service Discovery</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Load Balancing</strong></td>
<td>Ribbon</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Circuit Breaker</strong></td>
<td>Hystrix / Turbine</td>
<td></td>
<td>Istio Destination Rule</td>
</tr>
<tr>
<td><strong>API Gateway</strong></td>
<td>Zuul</td>
<td>Route Services</td>
<td>Istio Virtual Service Istio Gateway</td>
</tr>
<tr>
<td><strong>Dynamic Config</strong></td>
<td>Archaius</td>
<td>cf set-env</td>
<td>vim kube-app.yaml kubectl apply</td>
</tr>
</tbody>
</table>

2019-07-04 / Java Forum Stuttgart / Baris Cubukcuoglu & Fabian Keller
Limitations
There is always a flipside

• We’re now hiring for a full-time YAML engineer
• Sometimes infrastructure shouldn’t take care (e.g. retries)
• Yet another markup language stack to learn
• People *will* start to write microservices in other languages
Thanks!

kubectl apply –f questions.yaml

Come visit our booth!