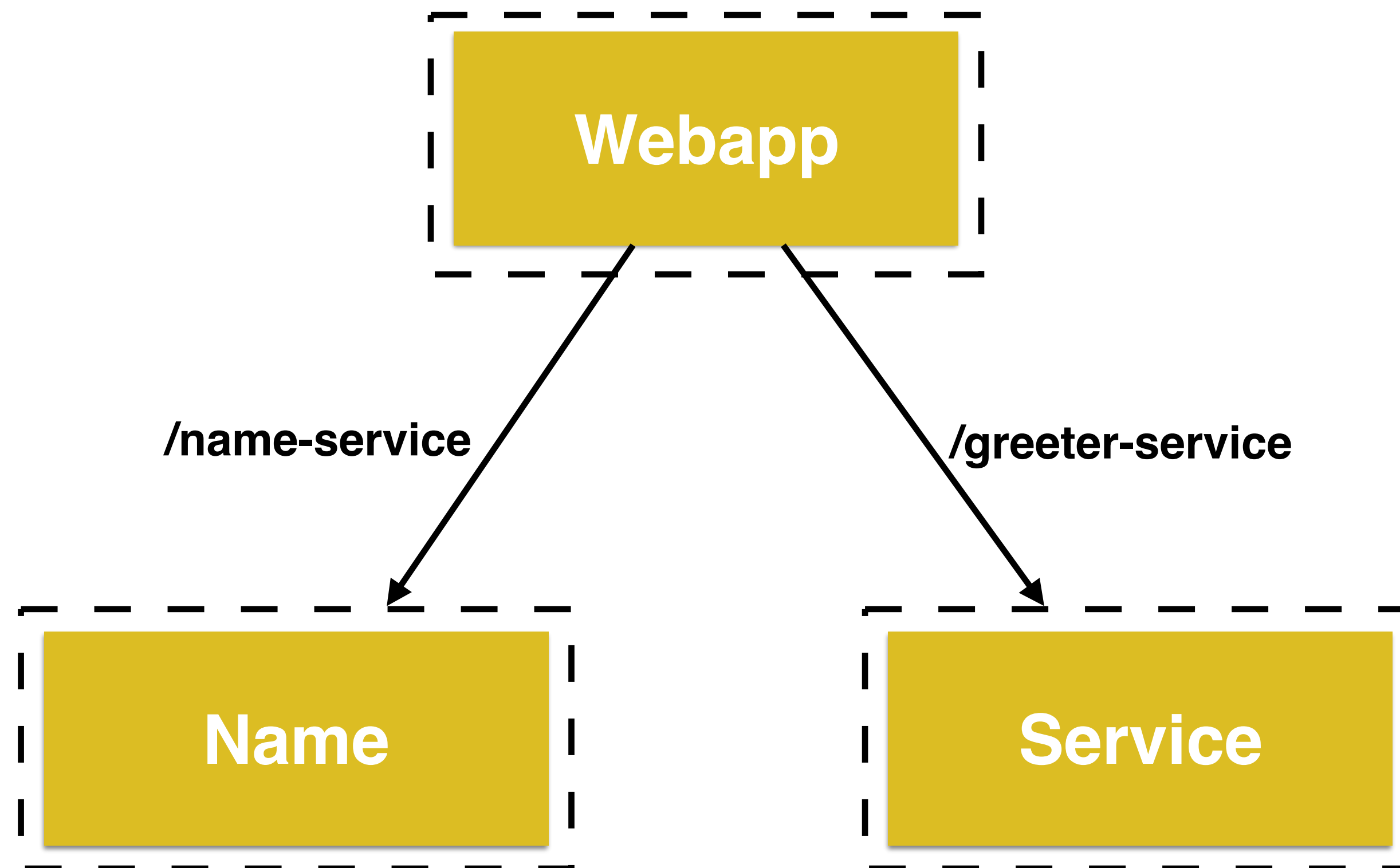# Service Discovery
In
Container Orchestration Frameworks

Arun Gupta, @arungupta

Docker Captain

Java Champion

JavaOne Rock Star (4 years)

NetBeans Dream Team

Silicon Valley JUG Leader
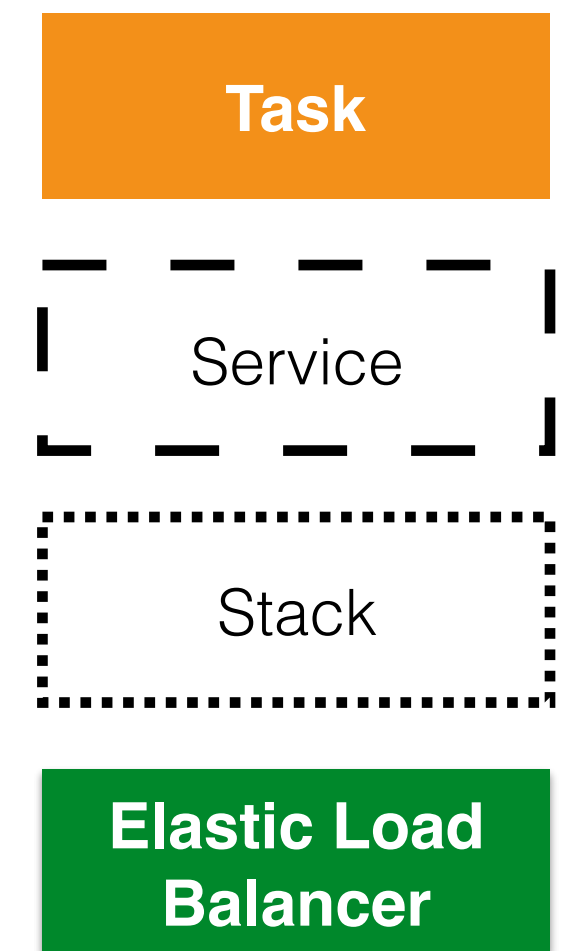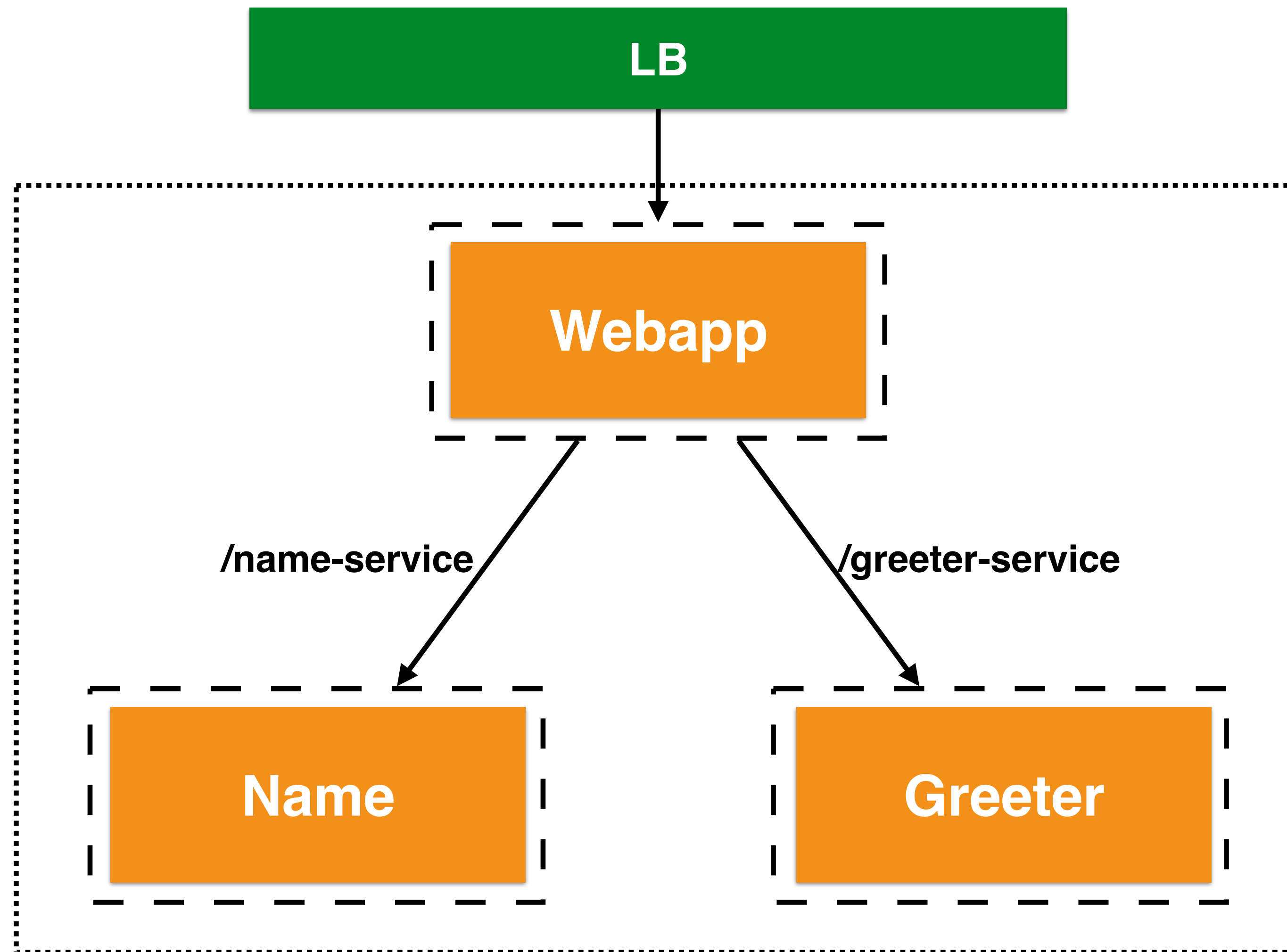
Author

Runner

Lifelong learner

Service Discovery

Docker for AWS

- CloudFormation templates
  - With a new VPC or a pre-existing VPC
- Integrated with
  - Autoscaling Groups - one for manager and worker each
  - Traffic routing using ELB
  - Container logs in CloudWatch
- Available in Docker CE and Docker EE
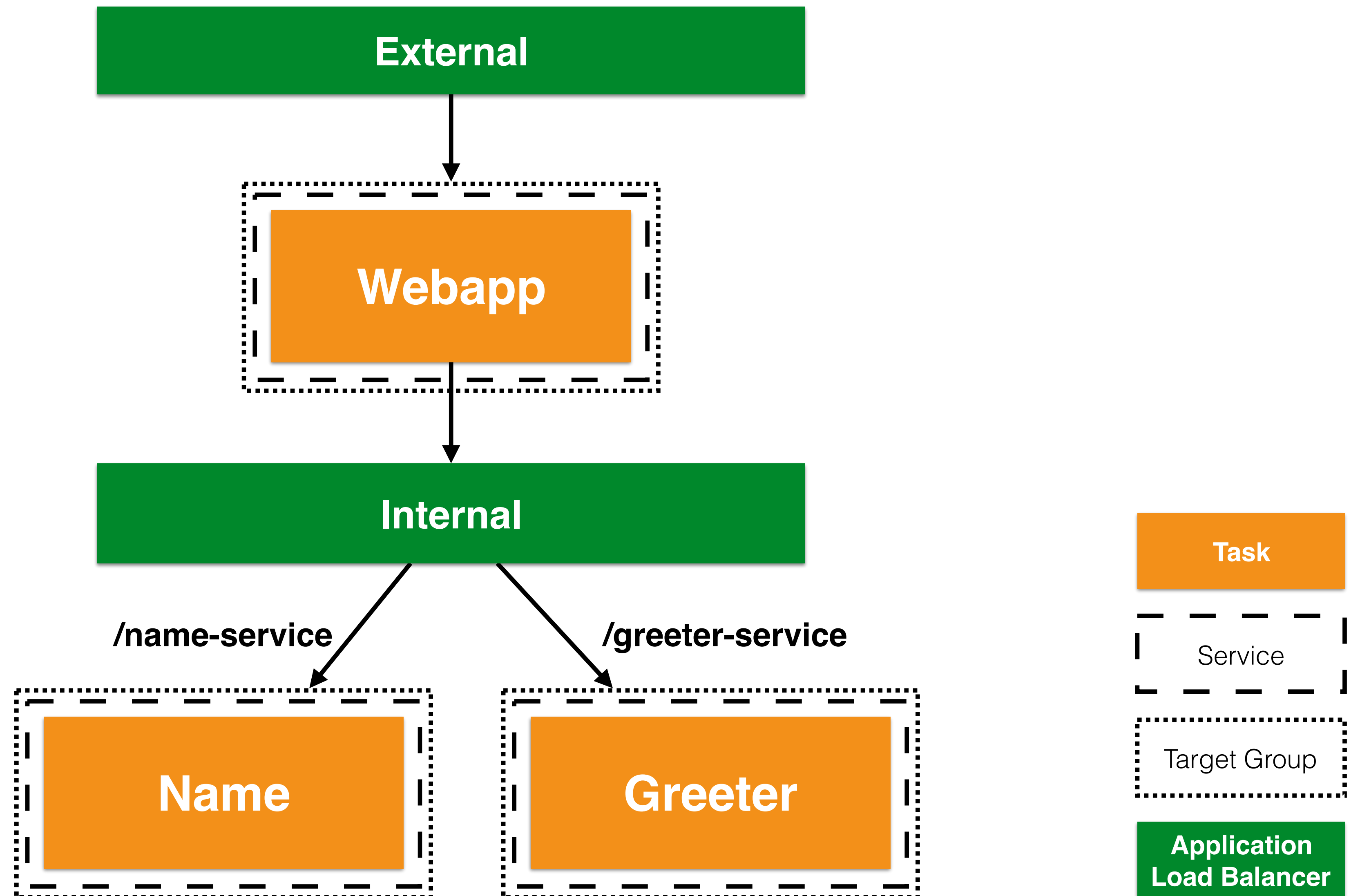
Docker

```yaml
1   version: '3'
2   services:
3     greeter-service:
4       build: ./services/greeter
5       image: arungupta/greeter-service:latest
6     name-service:
7       build: ./services/name
8       image: arungupta/name-service:latest
9     webapp-service:
10      build: ./services/webapp
11      image: arungupta/webapp-service:latest
12      ports:
13        - 80:8080
14      depends_on:
15        - greeter-service
16        - name-service
17      environment:
18        - NAME_SERVICE_HOST=name-service
19        - NAME_SERVICE_PORT=8080
20        - NAME_SERVICE_PATH=
21        - GREETER_SERVICE_HOST=greeter-service
22        - GREETER_SERVICE_PORT=8080
23        - GREETER_SERVICE_PATH=
```
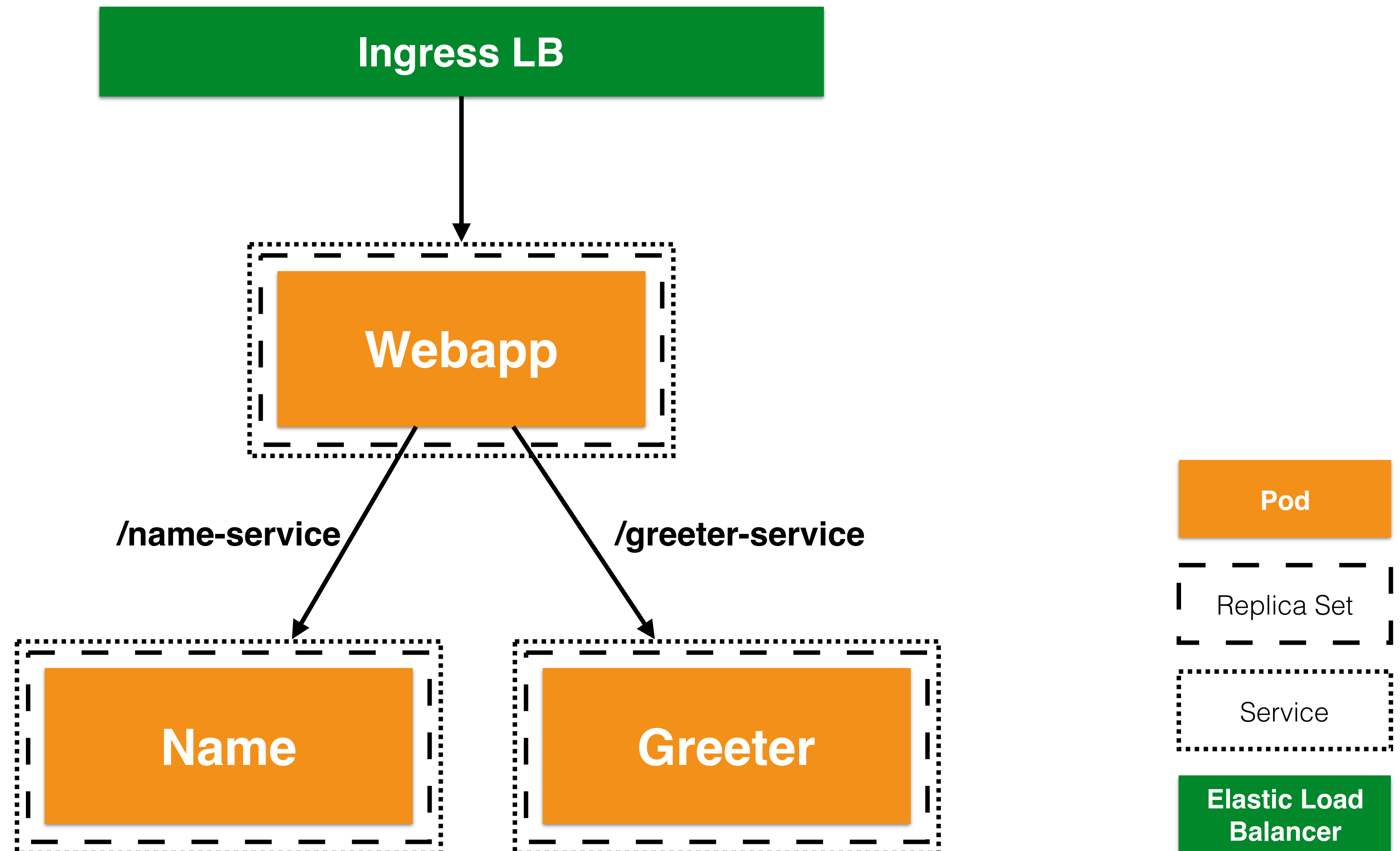
Amazon EC2 Container Service

- Container management service on EC2 instances
- Fully-managed: no need to install, operate and scale your own infrastructure
- Resource management, scheduling, task placement
- Designed for use with other AWS services
  - ELB, VPC, CloudWatch, CloudTrail, IAM, …

Amazon EC2 Container Service

TBD
CloudFormation template?

Kubernetes

Ingress LB

Webapp

/name-service

/greeter-service

Name

Greeter

Pod

Replica Set

Service

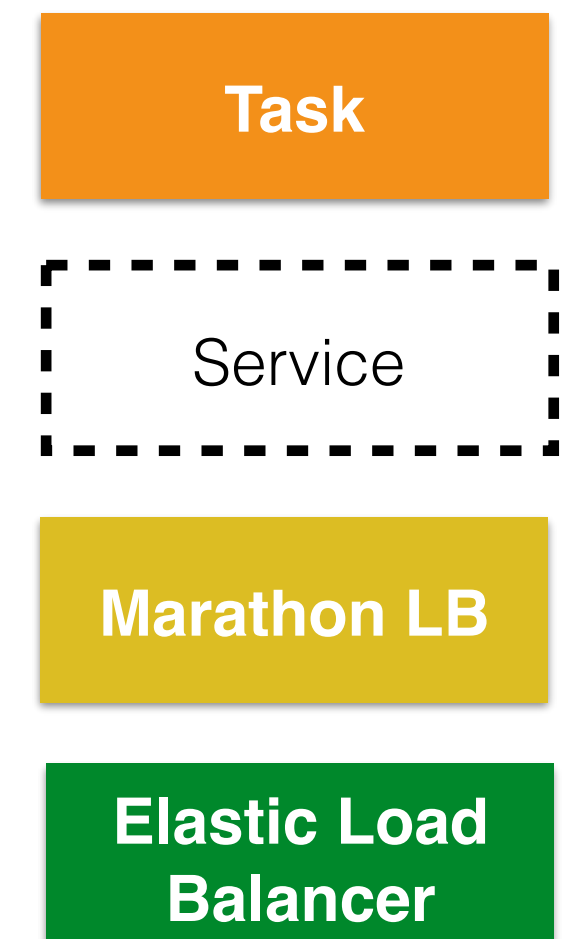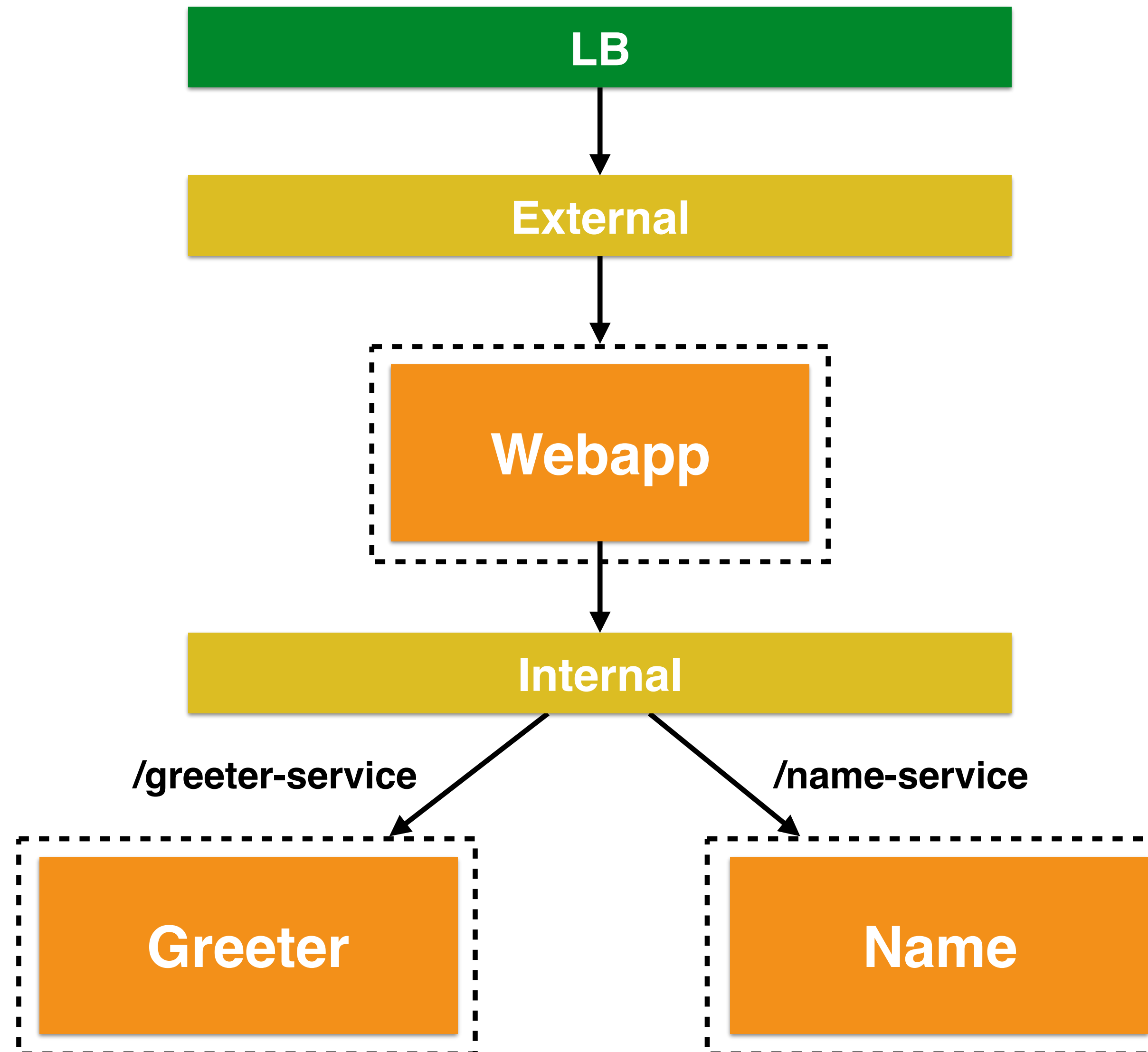Elastic Load Balancer

10

```yaml
1   apiVersion: v1
2   kind: Service
3   metadata:
4     name: name-service
5   spec:
6     selector:
7       app: name-pod
8     ports:
9       - port: 8080
10  ---
11  apiVersion: extensions/v1beta1
12  kind: ReplicaSet
13  metadata:
14    name: name-rs
15  spec:
16    replicas: 1
17    template:
18      metadata:
19        labels:
20          app: name-pod
21      spec:
22        containers:
23          - name: name
24            image: arungupta/name-service:latest
25            ports:
26              - containerPort: 8080
27  ---
```

```yaml
28  apiVersion: v1
29  kind: Service
30  metadata:
31    name: greeter-service
32  spec:
33    selector:
34      app: greeter-pod
35    ports:
36      - port: 8080
37  ---
38  apiVersion: extensions/v1beta1
39  kind: ReplicaSet
40  metadata:
41    name: greeter-rs
42  spec:
43    replicas: 1
44    template:
45      metadata:
46        labels:
47          app: greeter-pod
48      spec:
49        containers:
50          - name: name
51            image: arungupta/greeter-service:latest
52            ports:
53              - containerPort: 8080
54  ---
```

```yaml
55  apiVersion: v1
56  kind: Service
57  metadata:
58    name: webapp-service
59  spec:
60    selector:
61      app: webapp-pod
62    ports:
63      - name: web
64        port: 80
65    type: LoadBalancer
66  ---
67  apiVersion: extensions/v1beta1
68  kind: ReplicaSet
69  metadata:
70    name: webapp-rs
71  spec:
72    replicas: 1
73    template:
74      metadata:
75        labels:
76          app: webapp-pod
77      spec:
78        containers:
79          - name: webapp-pod
80            image: arungupta/webapp-service:latest
81            env:
82              - name: NAME_SERVICE_HOST
83                value: name-service
84              - name: NAME_SERVICE_PORT
85                value: "8080"
86              - name: NAME_SERVICE_PATH
87                value: /
88              - name: GREETER_SERVICE_HOST
89                value: greeter-service
90              - name: GREETER_SERVICE_PORT
91                value: "8080"
92              - name: GREETER_SERVICE_PATH
93                value: /
94            ports:
95              - containerPort: 8080
```

DC/OS

LB

External

Webapp

Internal

/greeter-service

/name-service

Greeter

Name

Task

Service

Marathon LB

Elastic Load Balancer

```json
{
  "marathon-lb":{
    "name":"marathon-lb-internal",
    "haproxy-group":"internal",
    "bind-http-https":false,
    "role":""
  }
}
```

```json
{
  "id": "/name",
  "cpus": 1,
  "mem": 1024,
  "instances": 1,
  "container": {
    "type": "DOCKER",
    "docker": {
      "image": "arungupta/name-service:latest",
      "network": "BRIDGE",
      "portMappings": [
        {
          "hostPort": 0,
          "containerPort": 8080,
          "servicePort": 10001
        }
      ]
    }
  },
  "labels": {
    "HAPROXY_GROUP": "internal"
  }
}
```

```json
{
  "id": "/greeter",
  "cpus": 1,
  "mem": 1024,
  "instances": 1,
  "container": {
    "type": "DOCKER",
    "docker": {
      "image": "arungupta/greeter-service",
      "network": "BRIDGE",
      "portMappings": [
        {
          "hostPort": 0,
          "containerPort": 8080,
          "servicePort": 10000
        }
      ]
    }
  },
  "labels": {
    "HAPROXY_GROUP": "internal"
  }
}
```

```json
{
  "id":"/webapp",
  "cpus":1,
  "mem":1024,
  "instances":1,
  "container":{
    "type":"DOCKER",
    "docker":{
      "image":"arungupta/webapp-service:latest",
      "network":"BRIDGE",
      "portMappings":[
        {
          "hostPort":0,
          "containerPort":8080,
          "servicePort": 10001,
          "protocol":"tcp"
        }
      ]
    }
  },
  "env":{
    "NAME_SERVICE_HOST":"marathon-lb-internal.marathon.mesos",
    "NAME_SERVICE_PORT":"10001",
    "NAME_SERVICE_PATH":"/",
    "GREETER_SERVICE_HOST":"marathon-lb-internal.marathon.mesos",
    "GREETER_SERVICE_PORT":"10000",
    "GREETER_SERVICE_PATH":"/"
  },
  "labels":{
    "HAPROXY_0_VHOST":"dcos-PublicSlaveLo-1B7KS7V0OI0Y6-1504556418.us-west-1.elb.amazo",
    "HAPROXY_GROUP":"external"
  }
}
```

References

- Slides: github.com/arun-gupta/container-service-discovery

- Docker: docker.io

- Amazon ECS: aws.amazon.com/ecs

- Kubernetes: kubernetes.io

- DC/OS: dcos.io