



Full infrastructure as code and CI/CD using AWS Elastic Container Service

who needs Kubernetes?

Mark van Holsteijn – CTO

ACTIEF ALS TELECOMAANBIEDER?

COIN helpt je verder

U bent actief als telecomaanbieder. Dat betekent dat u moet voldoen aan een aantal wettelijke verplichtingen. Het aanbieden van nummerbehoud en nummerafscherming aan uw klanten zijn enkele voorbeelden daarvan. Maar wat is wel en niet verplicht en hoe regelt u deze zaken? Vereniging COIN helpt u snel en goed op weg met een aantal praktische diensten.

Wie is COIN?

Vereniging COIN is een vereniging van Nederlandse en internationale leden die in de



NUMMERPORTABILITEIT

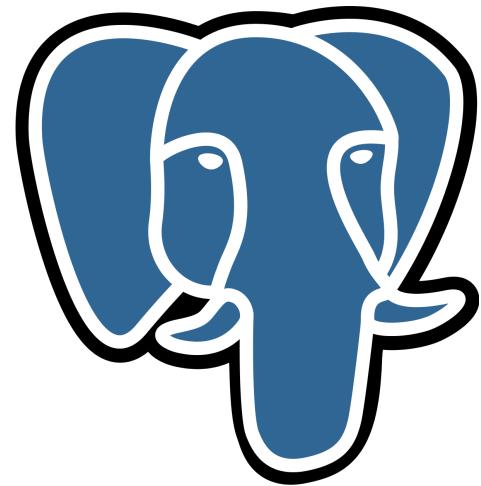
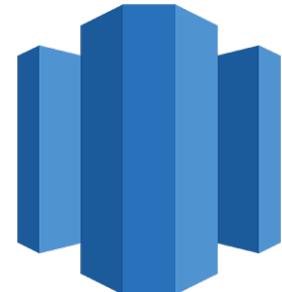


OVERSTAPPEN



**Full Automated Creation of Cloud Application Landscape -
No Manual Intervention Needed**

The Landscape components



ORACLE®

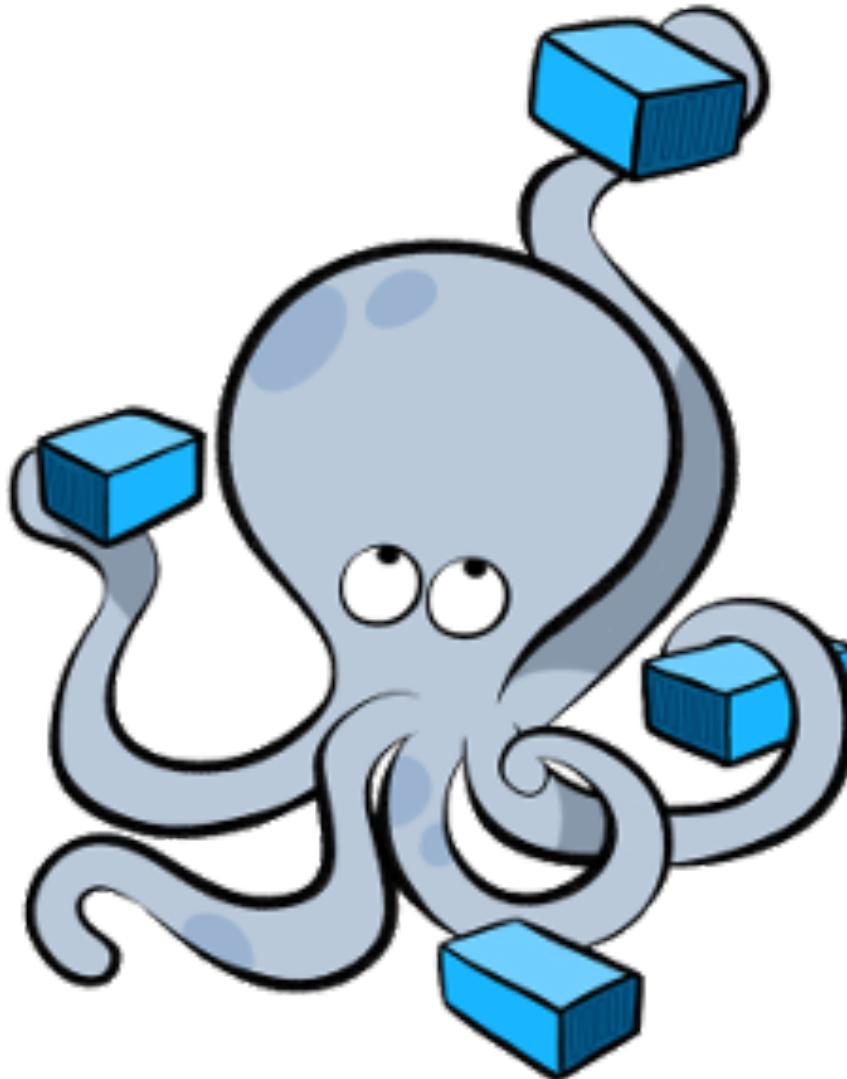
All Infrastructure as code in CloudFormation templates

```
  "timestamp": "2017-06-03T18:42:18.018", "deltaStartMillis": "0", "method": "handle", "requestID": "8249868e-afd8-46ac-9745-839146a20f09", "durationMillis": "5022", "sizeChars": "5022", "message": "Duration Log", "webURL": "/app/page/analyze", "webParams": "null", "sessionID": "144o2n620jm9trnd3s3n7wg0k", "class": "com.orgmanager.handlers.RequestHandler", "durationMillis": "36"}, {"timestamp": "2017-06-03T18:43:335.030", "deltaStartMillis": "2113", "method": "handle", "requestID": "789d89cb-bfa8-4e7d-8047-498454af885d", "durationMillis": "7", "sizeChars": "48455", "message": "Duration Log", "webURL": "/app/page/report", "webParams": "file=chartdata_new.json", "sessionID": "144o2n620jm9trnd3s3n7wg0k", "class": "com.orgmanager.handlers.RequestHandler", "durationMillis": "0"}, {"timestamp": "2017-06-03T18:46:921.000", "deltaStartMillis": "0", "method": "handle", "requestID": "7ac6ce95-19e2-4a60-88d7-6ead86e273d1", "durationMillis": "10"}, {"timestamp": "2017-06-03T18:42:18.018", "deltaStartMillis": "0", "method": "handle", "requestID": "b8861", "durationMillis": "23"}, {"timestamp": "2017-06-03T18:43:335.030", "deltaStartMillis": "2113", "method": "handle", "requestID": "789d89cb-bfa8-4e7d-8047-498454af885d", "durationMillis": "7"}, {"timestamp": "2017-06-03T18:46:921.000", "deltaStartMillis": "0", "method": "handle", "requestID": "7ac6ce95-19e2-4a60-88d7-6ead86e273d1", "durationMillis": "0"}]
```

All applications packaged into Docker Images



Docker Compose creates a complete Integration Environment on the desktop



All secrets randomly generated and stored in AWS Parameter Store

SECRET 1: 4C70
SECRET 2: 80
SECRET 3: A10H
SECRET 4: C41OAH
SECRET 5: F9DRSYO0
SECRET 6: 1XAZD
SECRET 7: 130F5H
SECRET 8: 130F5H
SECRET 9: 130F5H
SECRET 10: 130F5H
SECRET 11: 130F5H
SECRET 12: 130F5H
SECRET 13: 130F5H
SECRET 14: 130F5H
SECRET 15: 130F5H
SECRET 16: 130F5H
SECRET 17: 130F5H
SECRET 18: 130F5H
SECRET 19: 130F5H
SECRET 20: 130F5H
SECRET 21: 130F5H
SECRET 22: 130F5H
SECRET 23: 130F5H
SECRET 24: 130F5H
SECRET 25: 130F5H
SECRET 26: 130F5H
SECRET 27: 130F5H
SECRET 28: 130F5H
SECRET 29: 130F5H
SECRET 30: 130F5H
SECRET 31: 130F5H
SECRET 32: 130F5H
SECRET 33: 130F5H
SECRET 34: 130F5H
SECRET 35: 130F5H
SECRET 36: 130F5H
SECRET 37: 130F5H
SECRET 38: 130F5H
SECRET 39: 130F5H
SECRET 40: 130F5H
SECRET 41: 130F5H
SECRET 42: 130F5H
SECRET 43: 130F5H
SECRET 44: 130F5H
SECRET 45: 130F5H
SECRET 46: 130F5H
SECRET 47: 130F5H
SECRET 48: 130F5H
SECRET 49: 130F5H
SECRET 50: 130F5H
SECRET 51: 130F5H
SECRET 52: 130F5H
SECRET 53: 130F5H
SECRET 54: 130F5H
SECRET 55: 130F5H
SECRET 56: 130F5H
SECRET 57: 130F5H
SECRET 58: 130F5H
SECRET 59: 130F5H
SECRET 60: 130F5H
SECRET 61: 130F5H
SECRET 62: 130F5H
SECRET 63: 130F5H
SECRET 64: 130F5H
SECRET 65: 130F5H
SECRET 66: 130F5H
SECRET 67: 130F5H
SECRET 68: 130F5H
SECRET 69: 130F5H
SECRET 70: 130F5H
SECRET 71: 130F5H
SECRET 72: 130F5H
SECRET 73: 130F5H
SECRET 74: 130F5H
SECRET 75: 130F5H
SECRET 76: 130F5H
SECRET 77: 130F5H
SECRET 78: 130F5H
SECRET 79: 130F5H
SECRET 80: 130F5H
SECRET 81: 130F5H
SECRET 82: 130F5H
SECRET 83: 130F5H
SECRET 84: 130F5H
SECRET 85: 130F5H
SECRET 86: 130F5H
SECRET 87: 130F5H
SECRET 88: 130F5H
SECRET 89: 130F5H
SECRET 90: 130F5H
SECRET 91: 130F5H
SECRET 92: 130F5H
SECRET 93: 130F5H
SECRET 94: 130F5H
SECRET 95: 130F5H
SECRET 96: 130F5H
SECRET 97: 130F5H
SECRET 98: 130F5H
SECRET 99: 130F5H
SECRET 100: 130F5H

All secrets randomly generated and stored in AWS Parameter Store

DBPassword:

Type: Custom::Secret

Properties:

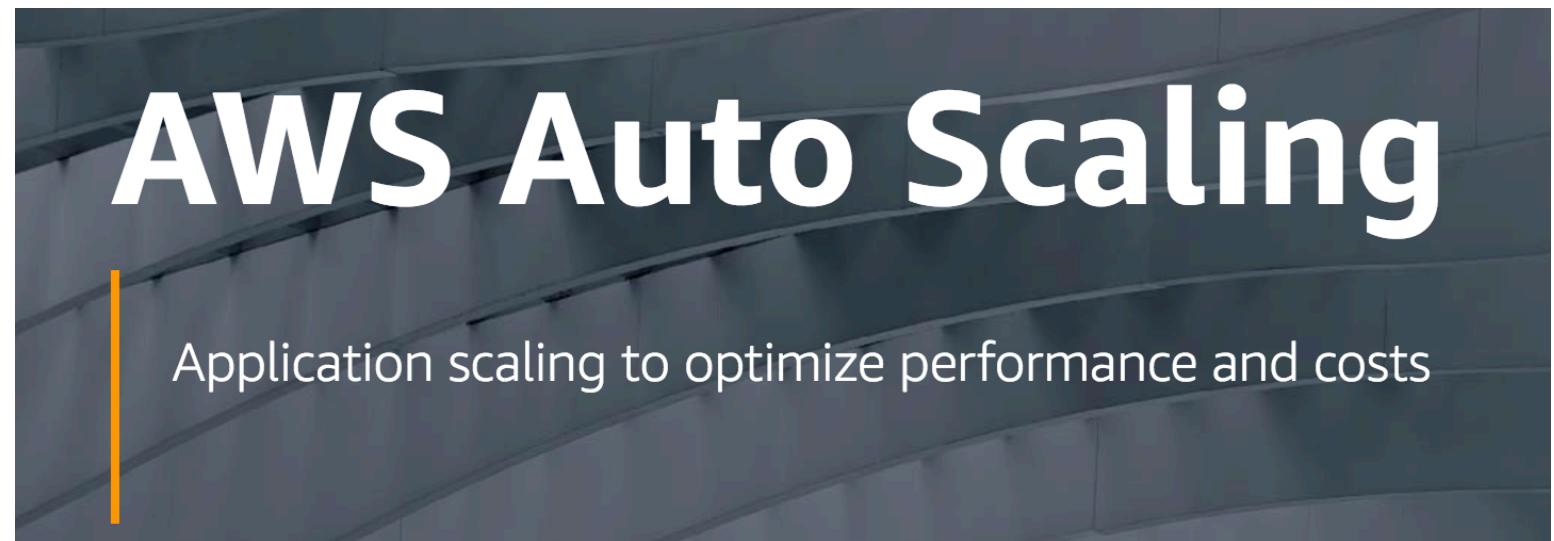
Name: !Sub '/\${AWS::StackName}/oracle/root/password'

Alphabet: abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

MaxLength: 30

ReturnSecret: true

CI Runners are stateless in auto scaling groups

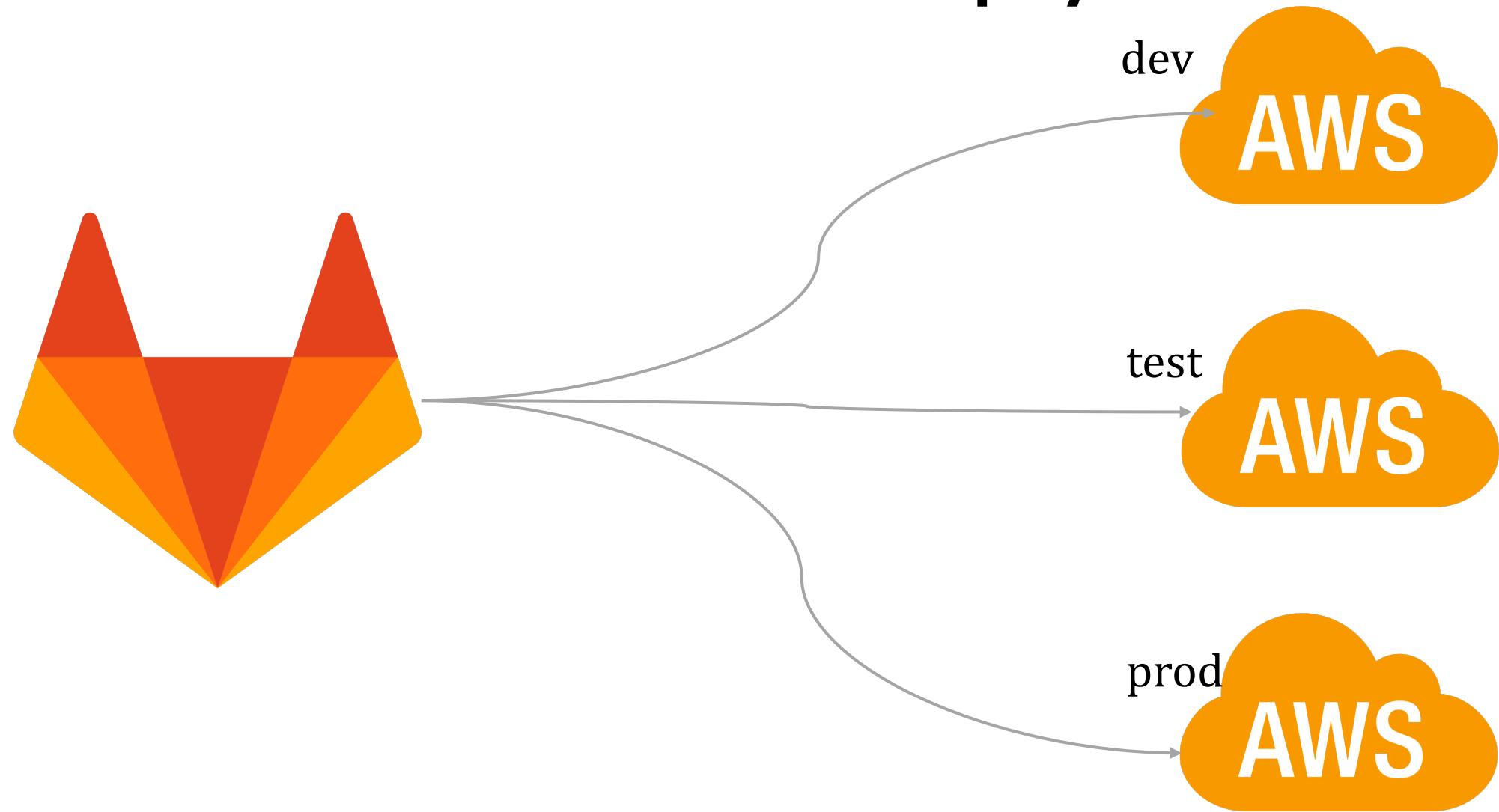


CI Runners are stateless in auto scaling groups

```
LaunchConfiguration:
  Type: AWS::AutoScaling::LaunchConfiguration
  DependsOn: PrivateKey
  Properties:
    ImageId: !Ref 'AMI'
    SecurityGroups:
      - !Ref 'VpcSecurityGroup'
    InstanceType: 't3.small'
    IamInstanceProfile: !Ref 'InstanceProfile'
    UserData:
      Fn::Base64: |
        #!/bin/bash
        export AWS_DEFAULT_REGION=eu-west-1
        export DEFAULT_IMAGE=gitlab-runner:0.1.2

## register the gitlab runner
mkdir /etc/gitlab-runner
docker run -v "/etc/gitlab-runner:/etc/gitlab-runner" \
  gitlab/gitlab-runner:latest register \
  -n --url "https://gitlab.com" \
  --description "aws runner" \
  --executor "docker" \
  --docker-image "$DEFAULT_IMAGE" \
  --docker-volumes /home/gitlab/.ssh:/home/gitlab-runner/.ssh \
  --registration-token \
  "$(aws --query Parameter.Value --output text ssm get-parameter --name /git"
```

Gitlab runners assume role to deploy to AWS accounts



Gitlab runners assume role to deploy to AWS accounts

```
CiCdRole:  
  Type: AWS::IAM::ManagedPolicy  
  Properties:  
    ManagedPolicyName: CiCdRole  
    Description: allows CI/CD runners to deploy to test, acceptance, production and devops  
    PolicyDocument:  
      Version: '2012-10-17'  
      Statement:  
        - Effect: Allow  
          Action:  
            - sts:AssumeRole  
          Resource:  
            - !Sub 'arn:aws:iam::${DevOpsAccountId}:role/CloudFormationStackUpdater'  
            - !Sub 'arn:aws:iam::${TestAccountId}:role/CloudFormationStackUpdater'  
            - !Sub 'arn:aws:iam::${AcceptanceAccountId}:role/CloudFormationStackUpdater'  
            - !Sub 'arn:aws:iam::${ProductionAccountId}:role/CloudFormationStackUpdater'  
        - Effect: Allow  
          Action:  
            - s3:*
```

```
deploy_test:  
  stage: deploy  
  before_script:  
    script:  
      - assume_role -p test -a 111111111111 -r CloudFormationStackUpdater -R gitlab-runner  
      - sceptre --var-file environments/test.yaml launch-env web
```



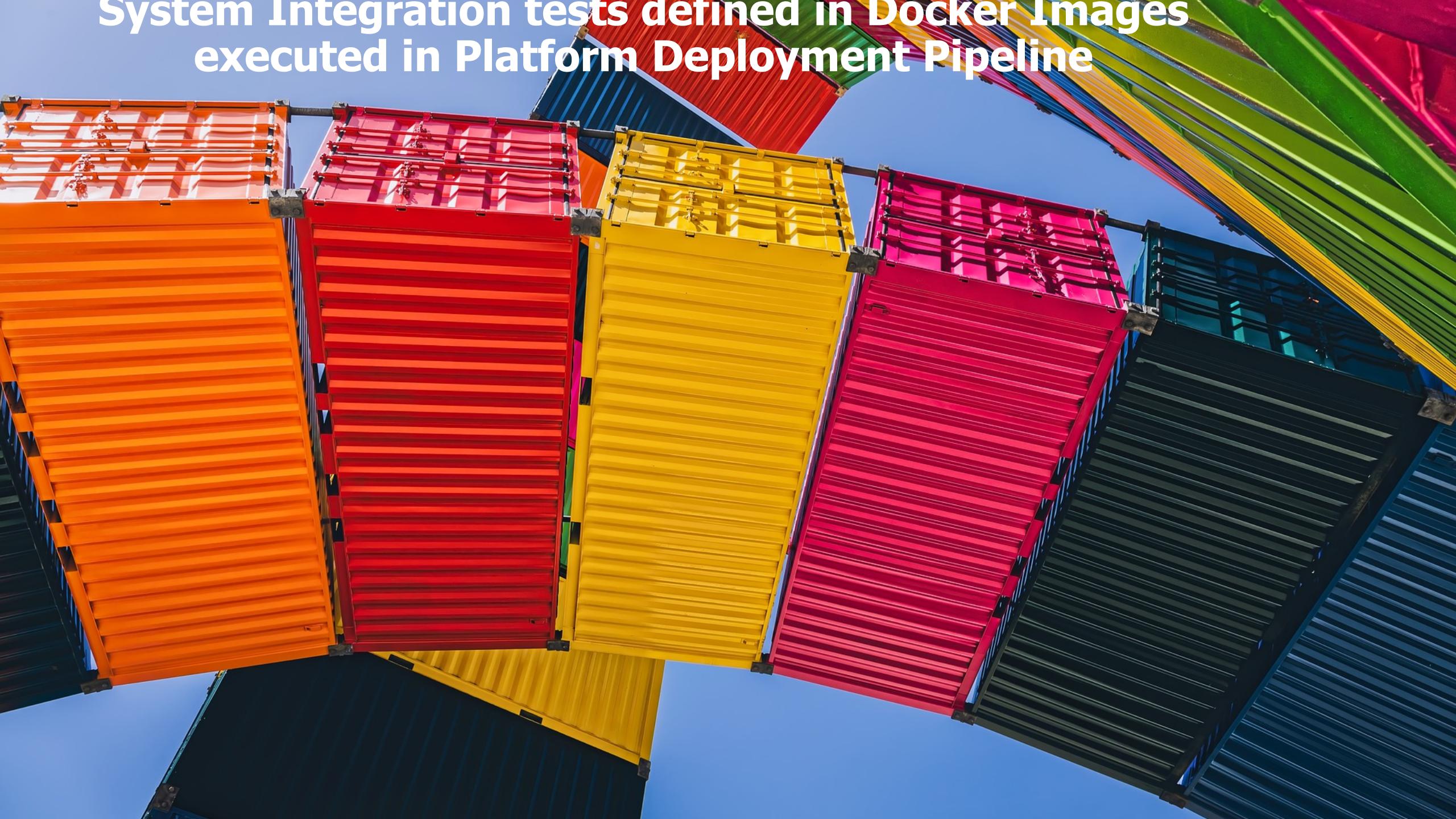
CI/CD of all users, groups, roles and policies to AWS accounts



CI/CD of all users, groups, roles and policies to AWS accounts

```
AcceptanceAdministratorRole:  
  Type: AWS::IAM::ManagedPolicy  
  Properties:  
    ManagedPolicyName: AllowAdministratorRoleOnAcceptance  
    Description: allows users to assume administrator role in acceptance accounts.  
    Groups:  
      - !Ref Administrators  
  PolicyDocument:  
    Version: '2012-10-17'  
    Statement:  
      - Effect: Allow  
        Action:  
          - sts:AssumeRole  
        Resource:  
          - !Sub 'arn:aws:iam:${AtAccountId}:role/Administrator'  
    Condition:  
      Bool:  
        aws:MultiFactorAuthPresent: 'true'
```

**System Integration tests defined in Docker Images
executed in Platform Deployment Pipeline**



System Integration tests defined in Docker Images executed in Platform Deployment Pipeline

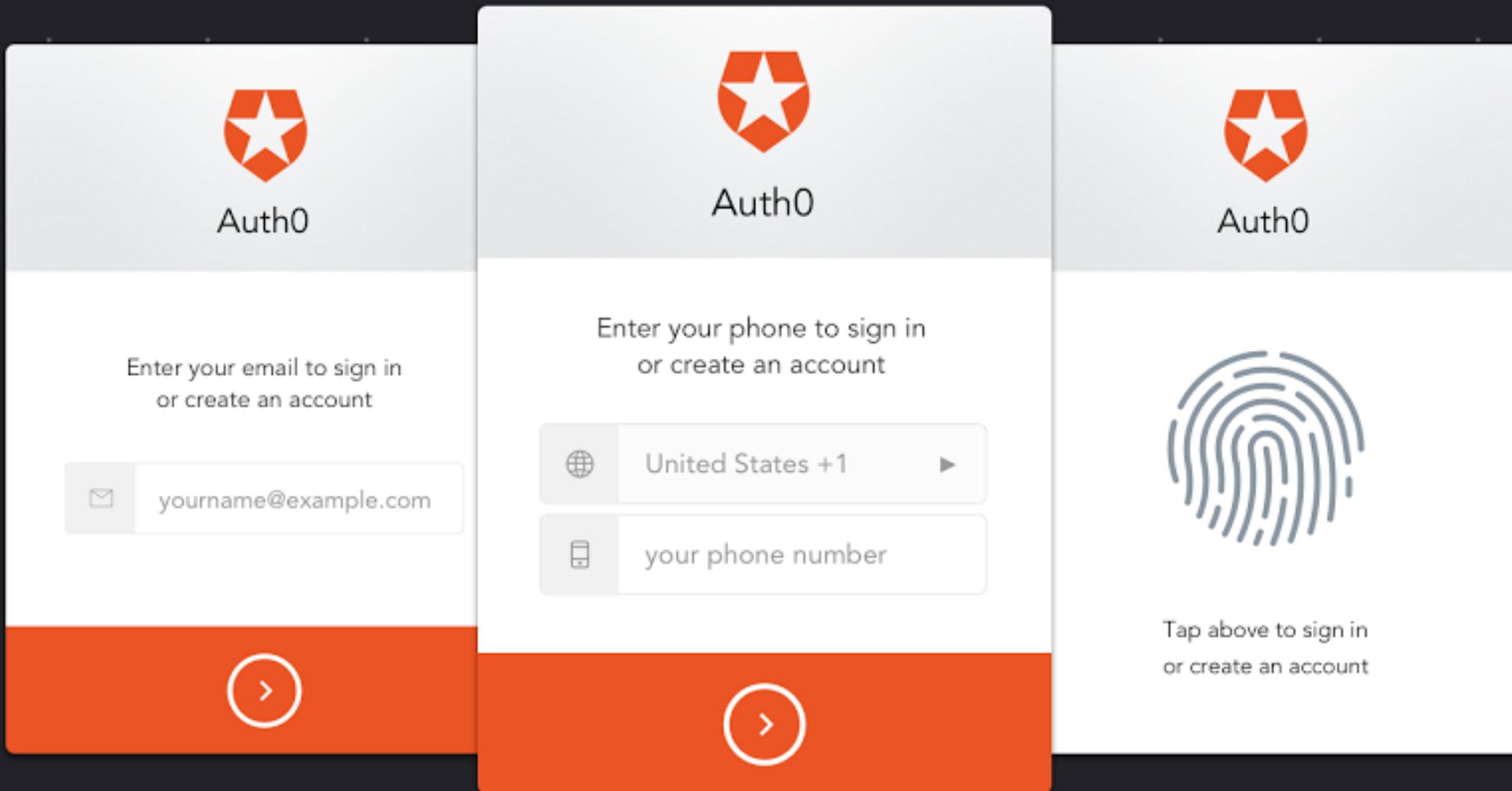
```
stages:
  - pre-deploy
  - deploy
  - test

admin-e2e:
  stage: test
  only:
    - master
  script:
    - docker run admin-e2e:latest

website-e2e:
  stage: test
  only:
    - master
  script:
    - docker run /website-e2e:latest

api-e2e:
  stage: test
  only:
    - master
  script:
    - docker run api-e2e:latest
```

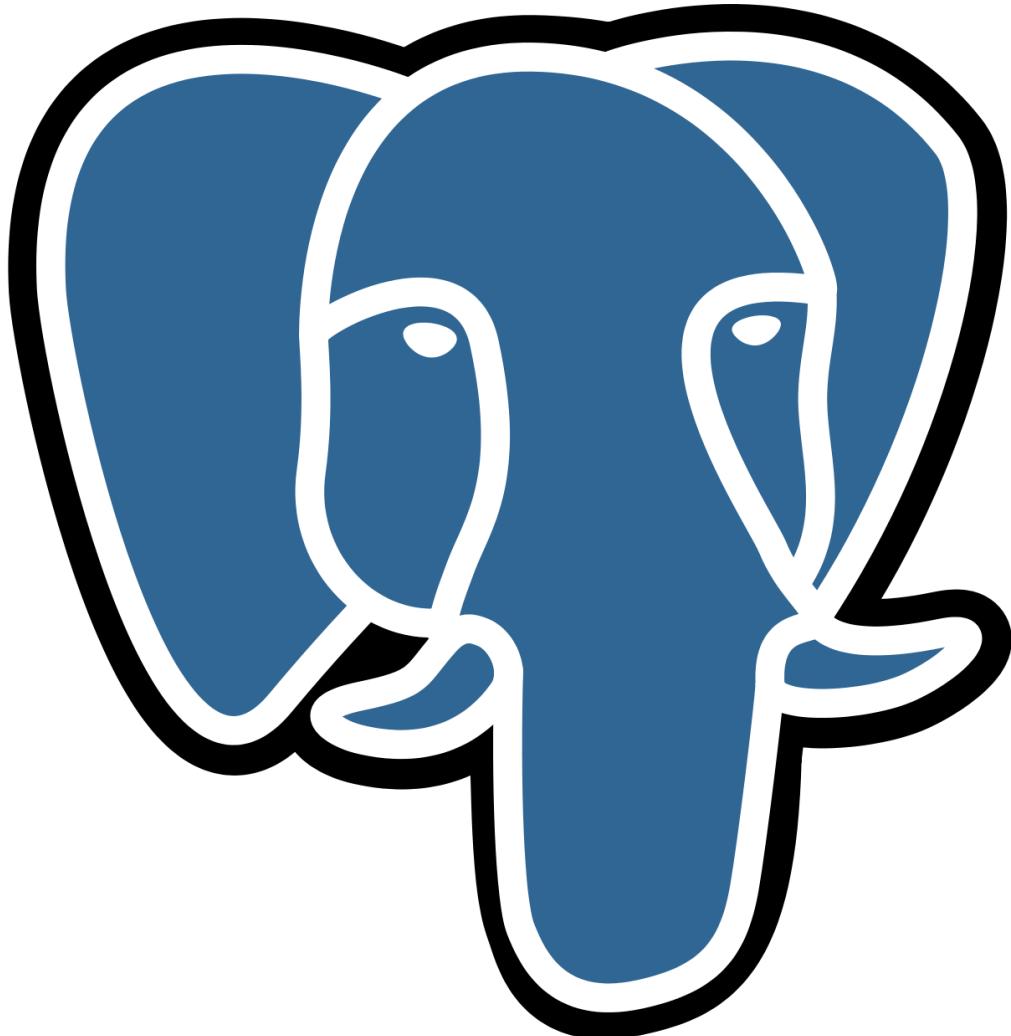
CI/CD of Auth0 Configuration with the Application



CI/CD of Auth0 Configuration with the Application

```
Auth0API:  
  Type: Custom::Auth0ResourceServer  
  Properties:  
    ServiceToken: !Sub 'arn:aws:lambda:${AWS::Region}:${AWS::AccountId}:function:cfn-auth0-provider'  
    Value:  
      name: consumer-api  
      identifier: https://api.binx.io/consumer-api  
      allow_offline_access: false  
      skip_consent_for_verifiable_first_party_clients: true  
      token_lifetime: 900  
      token_lifetime_for_web: 900  
      signing_alg: RS256  
      scopes:  
        - value: manage:consumers  
          description: manage all consumers  
        - value: manage:own-consumers  
          description: manage own consumers  
        - value: manage:users  
          description: manage users  
        - value: manage:groups  
          description: manage groups  
        - value: view:access  
          description: can view the application  
        - value: query:eventlog  
          description: can query the event logs
```

CI/CD of PostgreSQL and Oracle schemas and users



ORACLE®

CI/CD of PostgreSQL and Oracle schemas and users

```
KongUser:  
  Type: Custom::PostgreSQLUser  
  DependsOn:  
    - Database  
    - DBPassword  
    - KongPassword  
  Properties:  
    User: kong  
    PasswordParameterName: !Sub '${AWS::StackName}/postgres/kong/PGPASSWORD'  
  Database:  
    User: root  
    Host: !GetAtt 'Database.Endpoint.Address'  
    Port: !GetAtt 'Database.Endpoint.Port'  
    DBName: root  
    PasswordParameterName: !Sub '${AWS::StackName}/postgres/root/PGPASSWORD'  
  ServiceToken: !Sub 'arn:aws:lambda:${AWS::Region}:${AWS::AccountId}:function:binxio'
```



CI/CD of Kong API Gateway Configuration with Application



CI/CD of Kong API Gateway Configuration with Application

```
Resources:
  HeaderService:
    Type: Custom::KongService
    Properties:
      Service:
        name: header-service
        host: httpbin.org
        protocol: https
      AdminURL: !Ref 'AdminURL'
      ServiceToken: !Sub 'arn:aws:lambda:${AWS::Region}:${AWS::AccountId}:function:kong-header-service'
  HeaderRoute:
    Type: Custom::KongRoute
    Properties:
      Route:
        paths:
          - /headers
        service:
          id: !Ref 'HeaderService'
      AdminURL: !Ref 'AdminURL'
      ServiceToken: !Sub 'arn:aws:lambda:${AWS::Region}:${AWS::AccountId}:function:kong-header-route'
```

Automated updates of used Amazon Machine Images

Launch Actions ▾

Public images search : ecs Add filter 1

	AMI Name	AMI ID	Source	Owner	Visibility	Status	Cr
	amzn-ami-2015.09.c-amazon-ecs-optimized	ami-60627e0c	am...	amazon	Public	available	De
	amzn-ami-2015.09.d-amazon-ecs-optimized	ami-e1e6f88d	am...	amazon	Public	available	Ja
	amzn-ami-2015.09.e-amazon-ecs-optimized	ami-c3253caf	am...	amazon	Public	available	Ja
	amzn-ami-2015.09.f-amazon-ecs-optimized	ami-96b6adfa	am...	amazon	Public	available	Fe
	amzn-ami-2015.09.g-amazon-ecs-optimized	ami-341efb5b	am...	amazon	Public	available	Ma
	amzn-ami-2016.03.a-amazon-ecs-optimized	ami-9aeb0af5	am...	amazon	Public	available	Ap
	amzn-ami-2016.03.b-amazon-ecs-optimized	ami-1c769473	am...	amazon	Public	available	Ma

Image: ami-60627e0c

Details Tags

AMI ID	ami-60627e0c	AMI Name	amzn-ami-2015.09.c-amazon-ecs-optimized
Owner	591542846629	Source	amazon/amzn-ami-2015.09.c-amazon-ecs-
Status	available	State Reason	-

Automated updates of used Amazon Machine Images

The screenshot shows the AWS CloudFormation console interface. A modal window is open, displaying the configuration for a new stack. The configuration includes:

- AMI:**
 - Type: Custom::AMI
 - Properties:
 - Filters:
 - name: amzn-ami-2018.03.f-amazon-ecs-optimized
 - owner-alias: amazon
 - ServiceToken: !Sub 'arn:aws:lambda:\${AWS::Region}:\${AWS::AccountId}:function:aws-cfn-update-latest-ami'
- PortalECSLaunchConfiguration:**
 - Type: AWS::AutoScaling::LaunchConfiguration
 - Properties:
 - ImageId: !Ref 'AMI'
 - SecurityGroups:

Below the modal, the CloudFormation stack status is shown:

Image: ami-60627e0c update images:
stage: pre-deploy
only:
- schedules
script:
- aws-cfn-update latest-ami --ami-name-pattern amzn-ami-*ecs-optimized -

Visibility	Status	Cr
Public	available	De
Public	available	Ja
Public	available	Ja
Public	available	Fe
Public	available	Ma
Public	available	Ap
Public	available	Ma



**Rolling update of ECS server instances – Without
downtime**

Rolling update of ECS server instances – Without downtime

```
ASGSNSTopic:  
  Type: AWS::SNS::Topic  
  Properties:  
    Subscription:  
      - Endpoint: !GetAtt 'LambdaFunctionForASG.Arn'  
        Protocol: lambda  
    DependsOn: LambdaFunctionForASG  
  
LambdaFunctionForASG:  
  Type: AWS::Lambda::Function  
  Properties:  
    Code:  
      S3Bucket: system-lambdas-eu-central-1  
      S3Key: lambdas/cfn-ecs-lifecycle-service-0.1.3.zip  
    Description: Lambda code for the autoscaling hook triggers invoked when autoscaling events of launching and terminating instance occur  
    Handler: cfn_ecs_lifecycle_service.handler  
    Role: !GetAtt 'LambdaExecutionRole.Arn'  
    Runtime: python2.7  
    Timeout: '300'
```

All System and Application Logging to CloudWatch Logs

The screenshot shows the AWS CloudWatch Logs interface. On the left, a sidebar menu includes Dashboard, Alarms, ALARM (1), INSUFFICIENT (7), OK, Billing, Logs (selected), Metrics, Selected Metrics, Billing, EBS, EC2, RDS, and SNS. The Logs section has a red border. In the center, the path Log Groups > Streams for /var/log/messages > Events for i-f9cd4912 is displayed. A date/time selector shows 2014/09/16 17:15:51 UTC. The main area lists log events:

Creation Time	Event Data
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: imklog 5.8.10, log source =
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 rsyslogd: [origin software="rsyslogd"]
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] Initializing co
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] Initializing co
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] Initializing co
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] Linux version 3
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] Command line:
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] e820: BIOS-pro
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] BIOS-e820: [me]
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] BIOS-e820: [mem]
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] BIOS-e820: [mem]
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] BIOS-e820: [mem]
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] BIOS-e820: [mem]
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] NX (Execute Di
2014-09-16 17:15:51 UTC	▶ Sep 16 00:14:16 ip-10-144-226-204 kernel: [0.000000] SMBIOS 2.4 pres

All System and Application Logging to CloudWatch Logs

```
- path: /etc/awslogs/awslogs.conf
  permissions: '0644'
  content: |
    [general]
    state_file = /var/lib/awslogs/agent-state

    [/var/log/dmesg]
    file = /var/log/dmesg
    log_stream_name = ${instance_id}/var/log/dmesg
    log_group_name = /${Ref:DomainName}/ecs

    [/var/log/messages]
    file = /var/log/messages
    log_stream_name = ${instance_id}/var/log/messages
    log_group_name = /${Ref:DomainName}/ecs
    datetime_format = %b %d %H:%M:%S

    [/var/log/secure]
    file = /var/log/secure
    log_stream_name = ${instance_id}/var/log/secure
    log_group_name = /${Ref:DomainName}/ecs
    datetime_format = %b %d %H:%M:%S
```

for /var/log/messages > Events for i-f9cd4912

Time	Event Data
Sep 16 00:14:16	ip-10-144-226-204 kernel: [0.000000] BIOS-e820: [mem]
Sep 16 00:14:16	ip-10-144-226-204 kernel: [0.000000] NX (Execute Di
Sep 16 00:14:16	ip-10-144-226-204 kernel: [0.000000] SMBIOS 2.4 pres

TaskDefinition:

- Type: AWS::ECS::TaskDefinition

Properties:

- Family: !Sub '\${DomainName}-abc'
- NetworkMode: bridge
- TaskRoleArn: !Ref 'Role'
- ContainerDefinitions:

 - Name: service-abc
 - Cpu: 100

Essential: true

PortMappings:

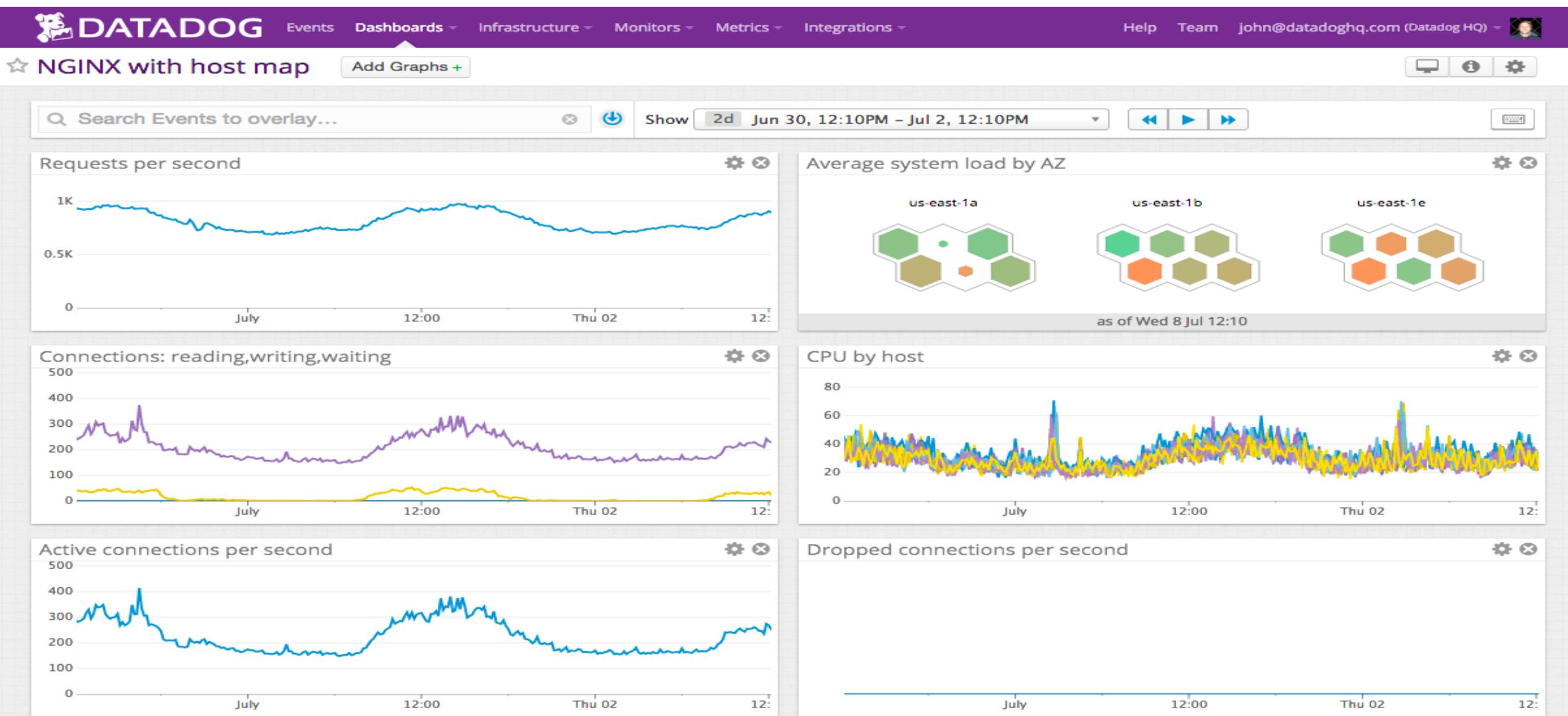
- ContainerPort: 8443
- Protocol: tcp

Image: service-abc:1.3.4

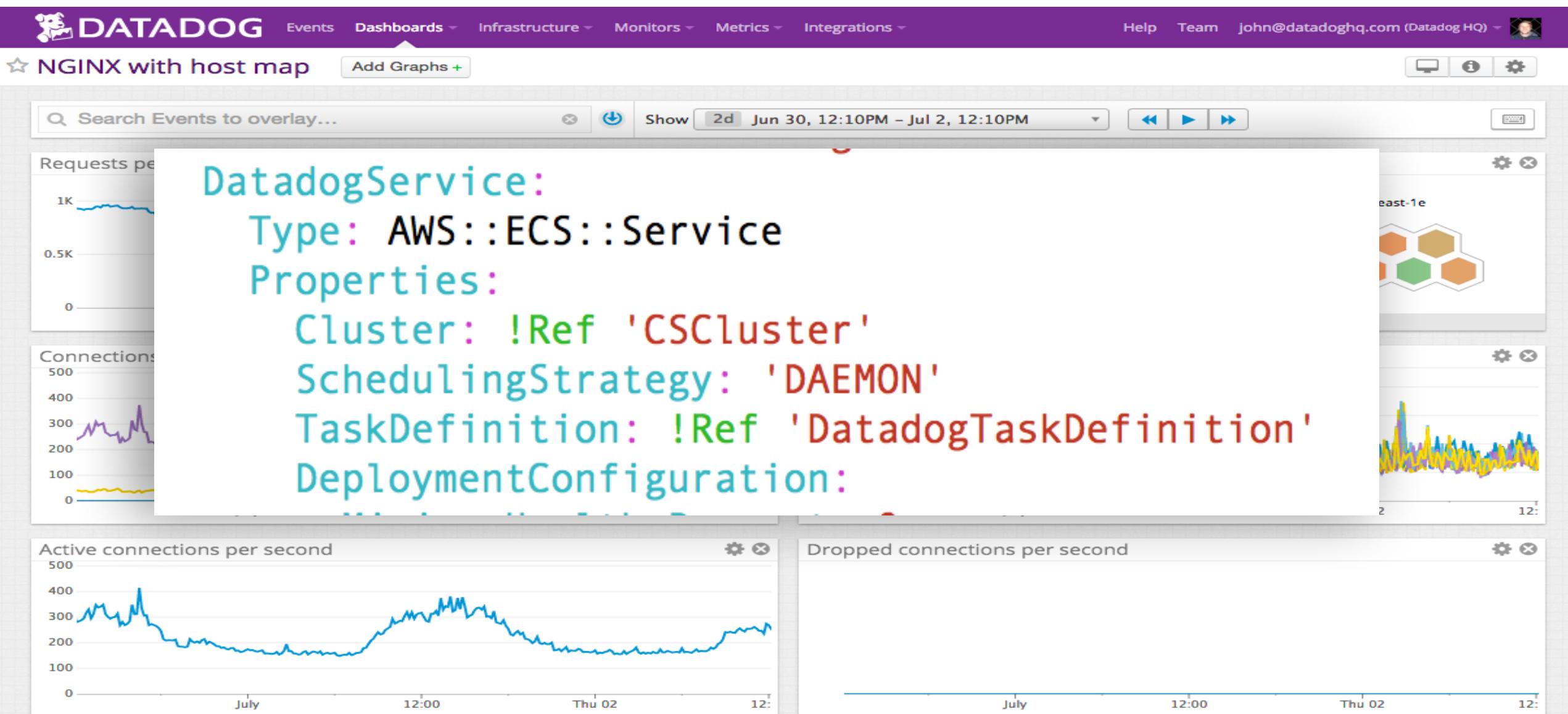
LogConfiguration:

- LogDriver: awslogs
- Options:
 - awslogs-group: !Ref 'ServiceLogGroup'
 - awslogs-region: !Ref 'AWS::Region'

Datadog used as metrics collector and alert monitor



Datadog used as metrics collector and alert monitor



Complete Infrastructure As Code setup replicated at the Harbor of Rotterdam in less than 7 days



❖ While replacing Gitlab with Github & AWS CodeBuild and Auth0 with a AWS MFA setup



**Full infrastructure as code and CI/CD
using AWS Elastic Container Service**