Logging & Monitoring

of a Kubernetes Cluster and its running applications – implemented on AWS

Structure

1. Used Services
2. Kubernetes
   - What is K8s?
   - Terminology
   - Architectural Overview of a Kubernetes Cluster
3. My Kubernetes Cluster on AWS „myEKS“
   1. Architectural Overview
   2. Components of EKS Cluster (esp. node group)
   3. Workflow
   4. Deployments / Pods
   5. (RBAC)
   6. Monitoring (Kubernetes-Dashboard, Grafana, Prometheus)
   7. Logging with CloudWatch
4. Summary & Outlook

1. Used Services

- AWS Cost Management
- IAM (Identity and Access Management)
- AWS Cloud9 (EC2 + EBS)
- EKS (Elastic Kubernetes Service)
- ECR (Elastic Container Registry)
- Amazon CloudWatch

Amazon Cost Management

AWS Identity and Access Management (IAM)

AWS Cloud9

Amazon EC2

Amazon EKS

Amazon ECR

Amazon CloudWatch

Amazon Virtual Private Cloud (Amazon VPC)

Amazon Elastic Block Store (Amazon EBS)

AWS CloudFormation

2. Kubernetes (K8s)

OBJECTS Overview:

- **Pod:** a wrapper around one or more containers
- **DaemonSet:** implements a single instance of a pod on a worker node
- **Deployments:** Details how to roll out (or roll back) across version of your application

- **ReplicaSet**: ensures a defined number of pods are always running
- **Job:** ensures a pod properly runs to completions
- **Service:** maps a fixed IP adress to a logical group of pods

Architectural Overview

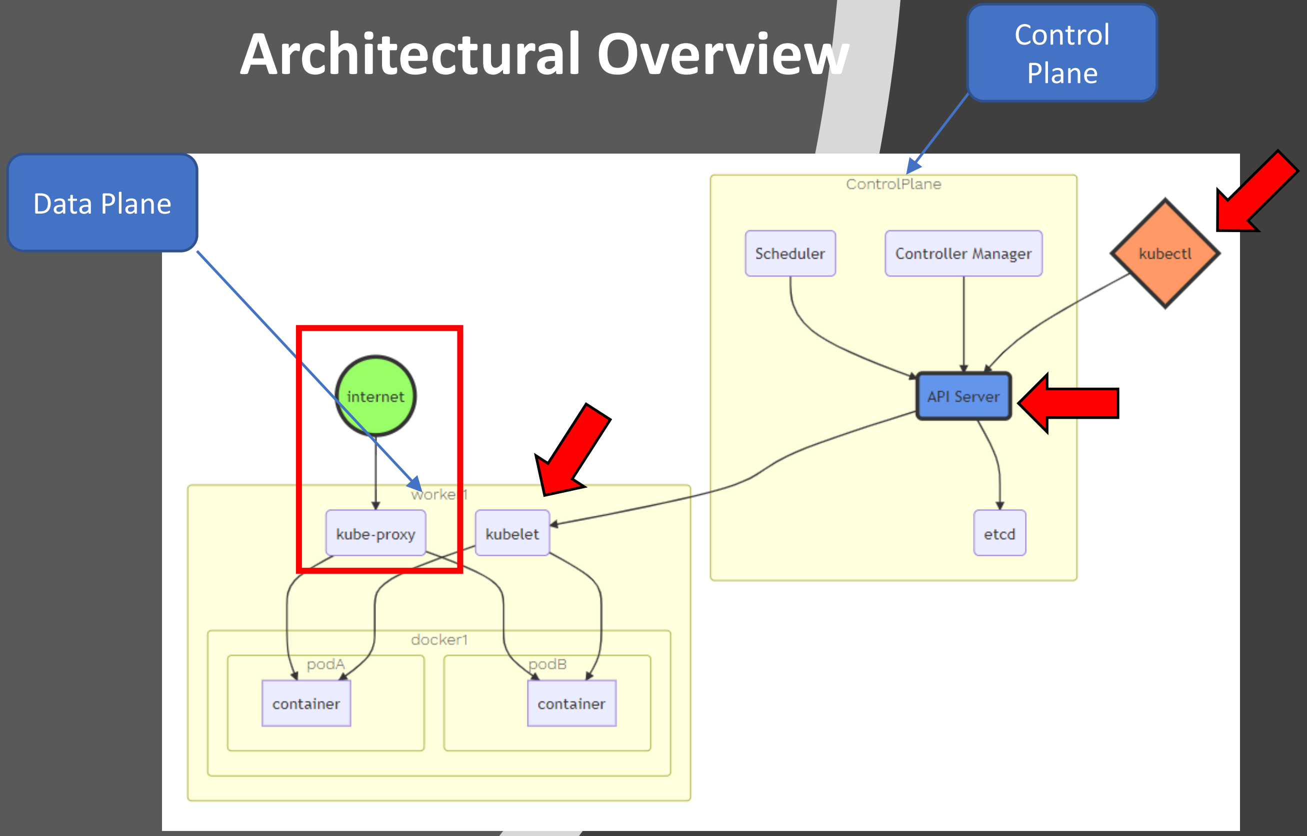


Image 1: Architectural Overview

3. My Kubernetes Cluster on AWS „myEKS“

3. My Kubernetes Cluster on AWS „myEKS“
    1. Architectural Overview
    2. Components of EKS Cluster (esp. node group)
    3. Workflow
    4. Deployments / Pods
    5. (RBAC)
    6. Monitoring (Kubernetes-Dashboard, Grafana, Prometheus)
    7. Logging with CloudWatch

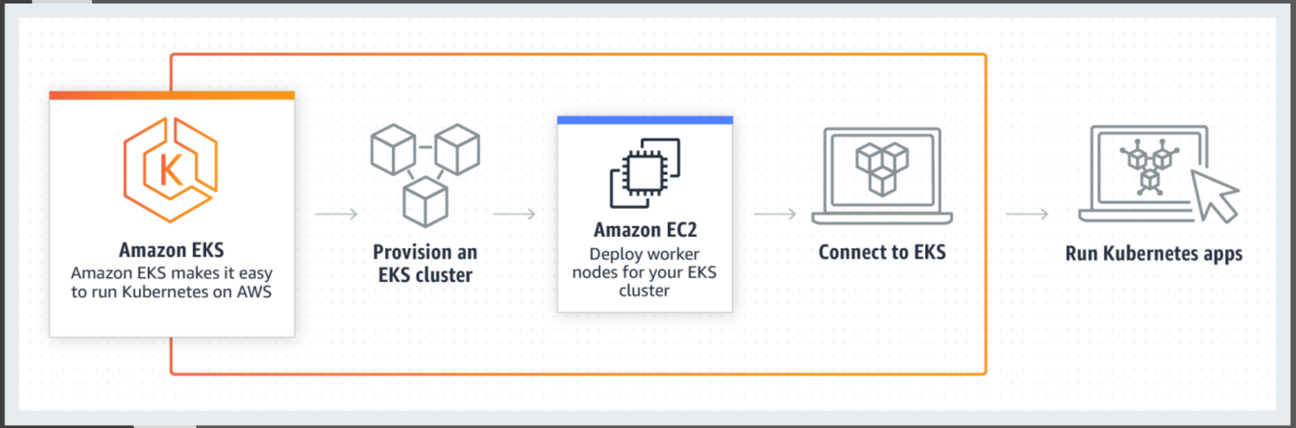
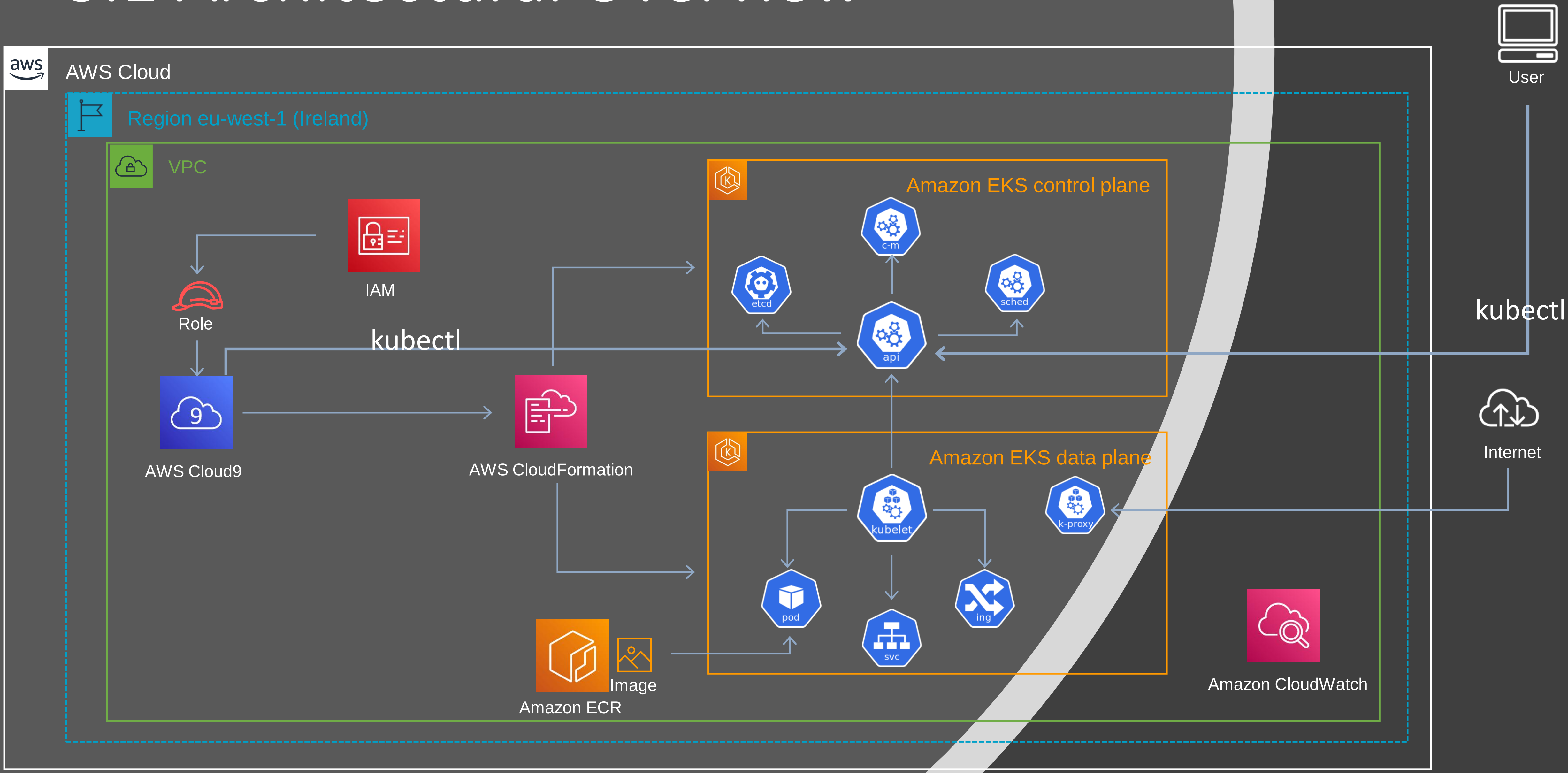

Amazon EKS
Amazon EKS makes it easy to run Kubernetes on AWS
Provision an EKS cluster
Amazon EC2
Deploy worker nodes for your EKS cluster
Connect to EKS
Run Kubernetes apps

Image 2: Amazon EKS

3.1 Architectural Overview



3.2 My Kubernetes Cluster on AWS „myEKS“

- Why no control plane? managed by aws eks
- only managed worker nodegroup deployed (ec2 instances)
- Configuring auto-scaling
- Assigning pods to nodes with labels

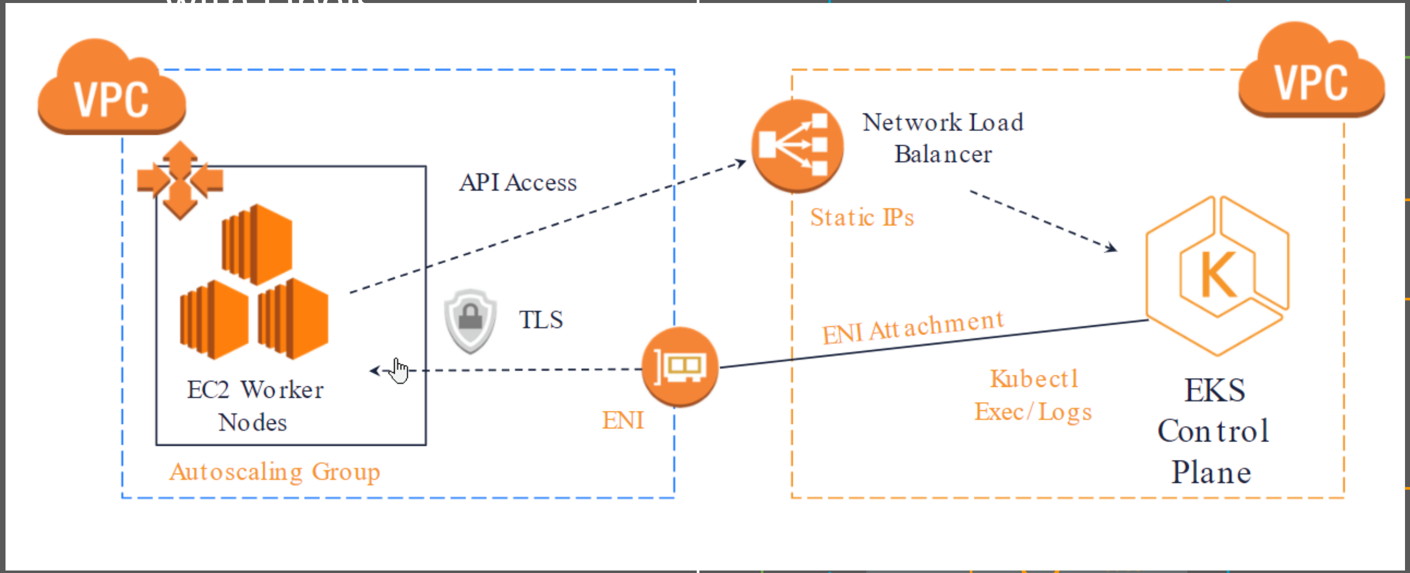
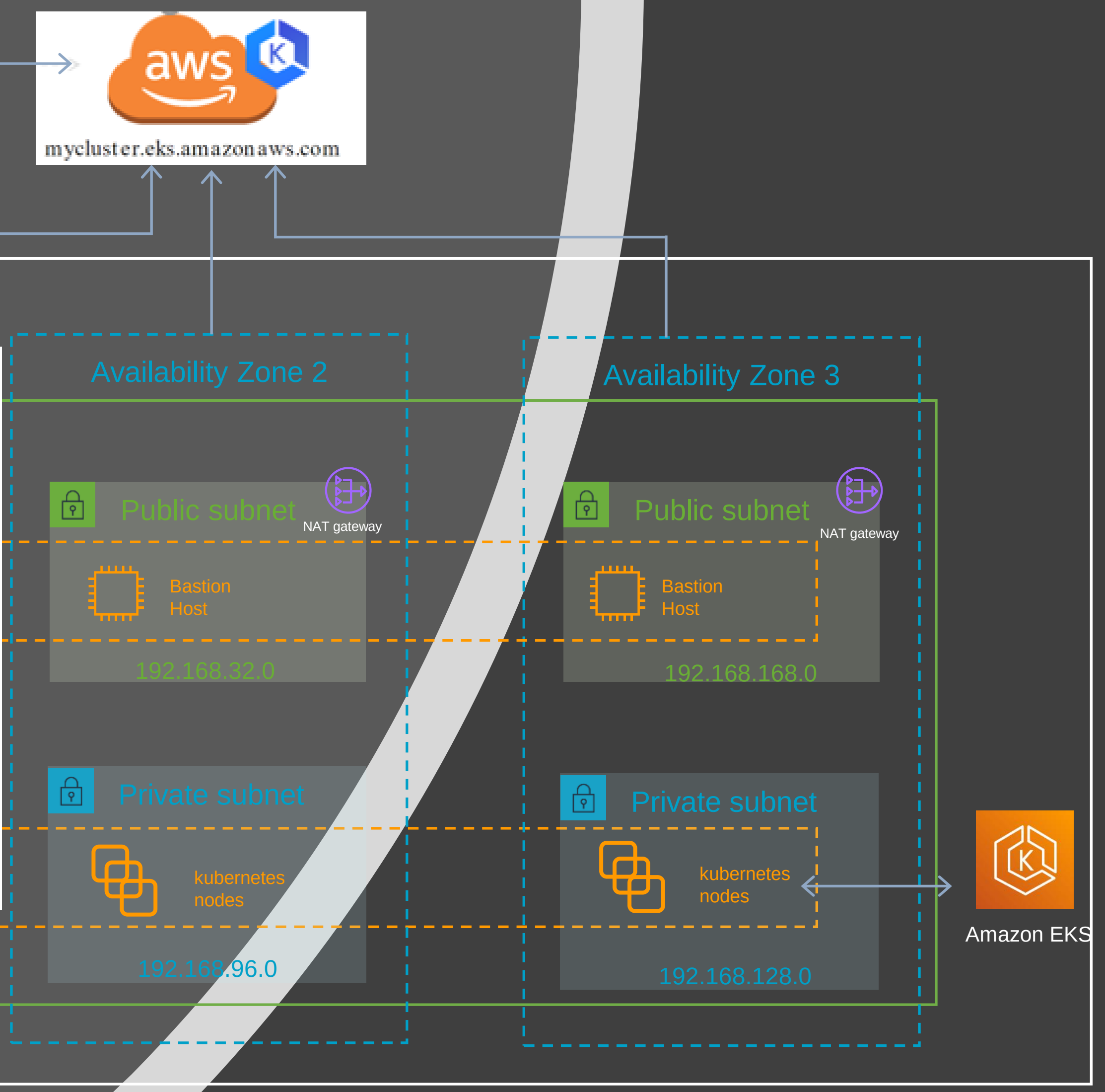
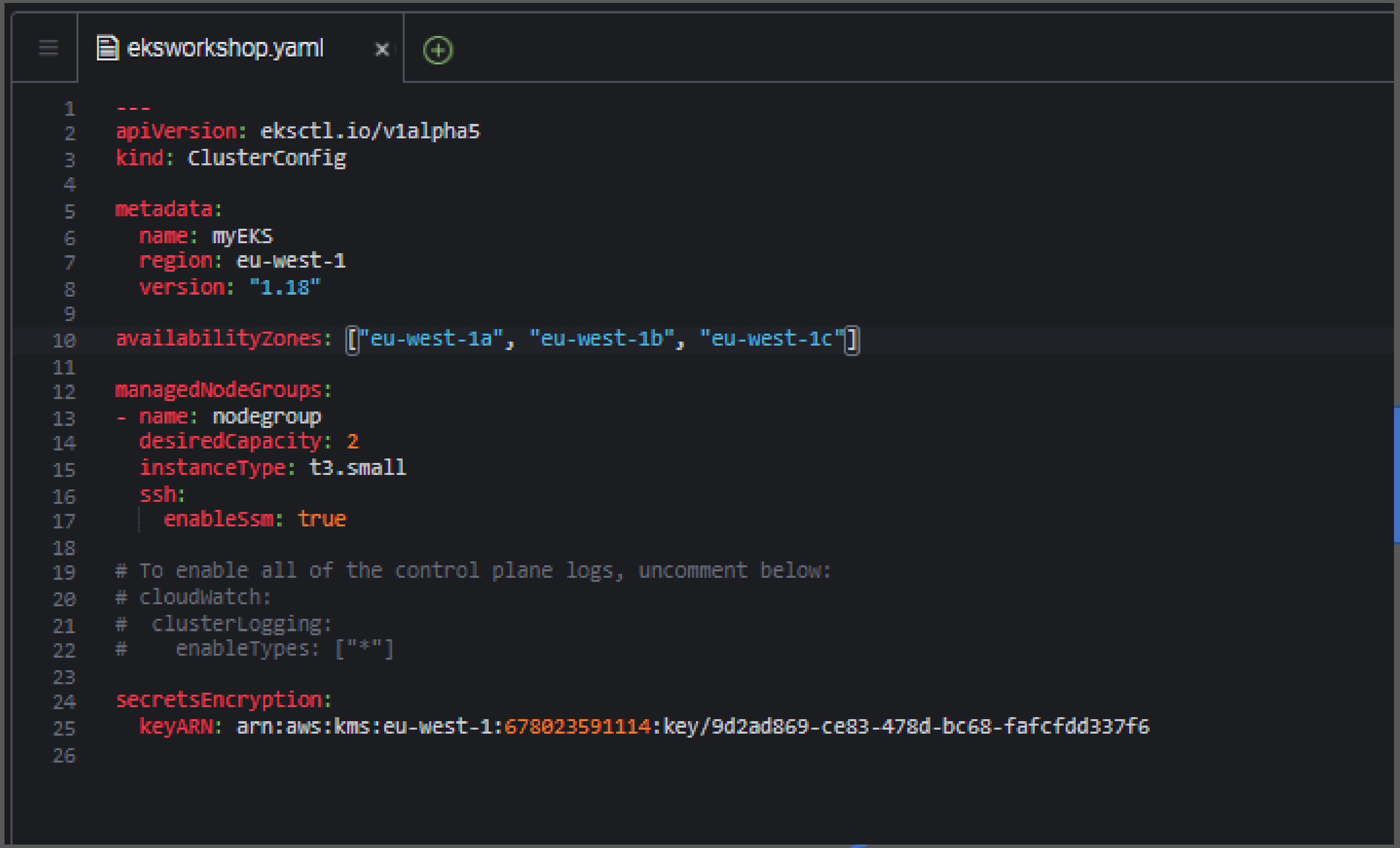


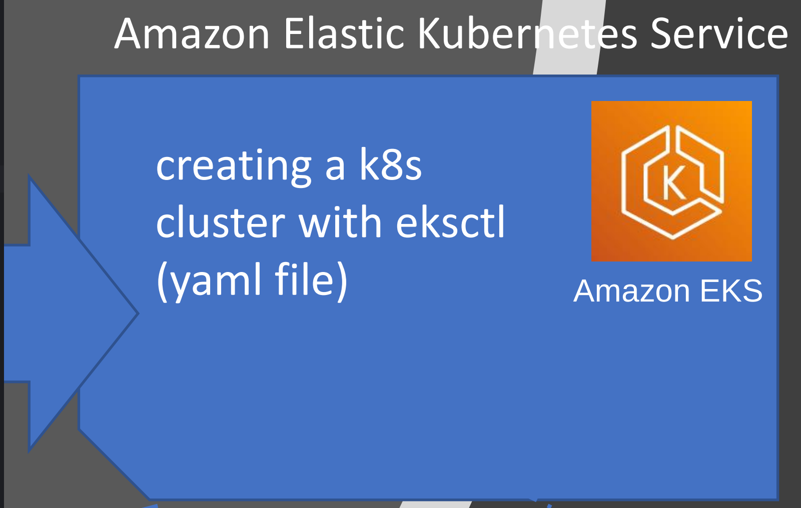
Image 3: EKS Control and Data Plane



3.3 Workflow

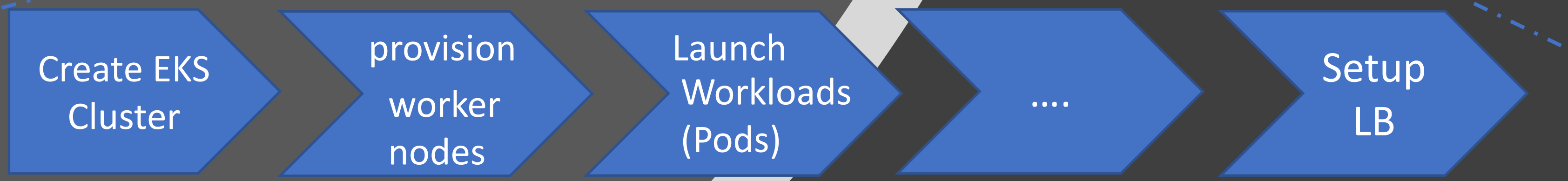


```yaml
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: myEKS
  region: eu-west-1
  version: "1.18"

availabilityZones: ["eu-west-1a", "eu-west-1b", "eu-west-1c"]

managedNodeGroups:
- name: nodegroup
  desiredCapacity: 2
  instanceType: t3.small
  ssh:
    enableSsm: true

# To enable all of the control plane logs, uncomment below:
# cloudWatch:
#  clusterLogging:
#    enableTypes: ["*"]

secretsEncryption:
  keyARN: arn:aws:kms:eu-west-1:678023591114:key/9d2ad869-ce83-478d-bc68-fafcfdd337f6

```



Amazon Elastic Kubernetes Service

creating a k8s cluster with eksctl (yaml file)



3.4 Deployments / Pods (+ ECR)

Amazon ECR

Image

- **ECR** = **E**lastic **C**ontainer **R**egistry
- easy to store, manage, share, and deploy container images
- highly available and high performance architecture
- privately <----> publicly   (management possible by IAM)
- works with ECS, EKS and AWS Lambda
- further: with Fargate one-click deployments

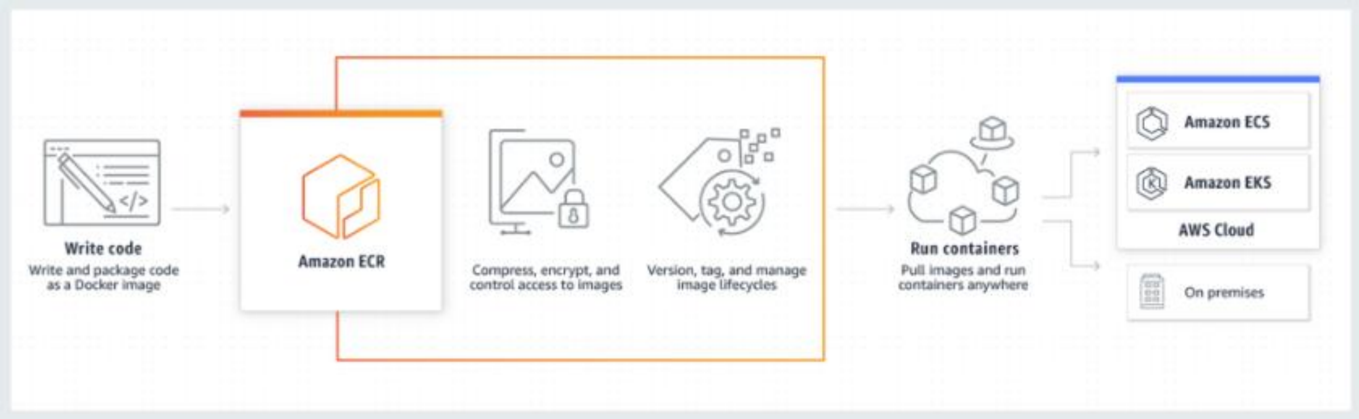
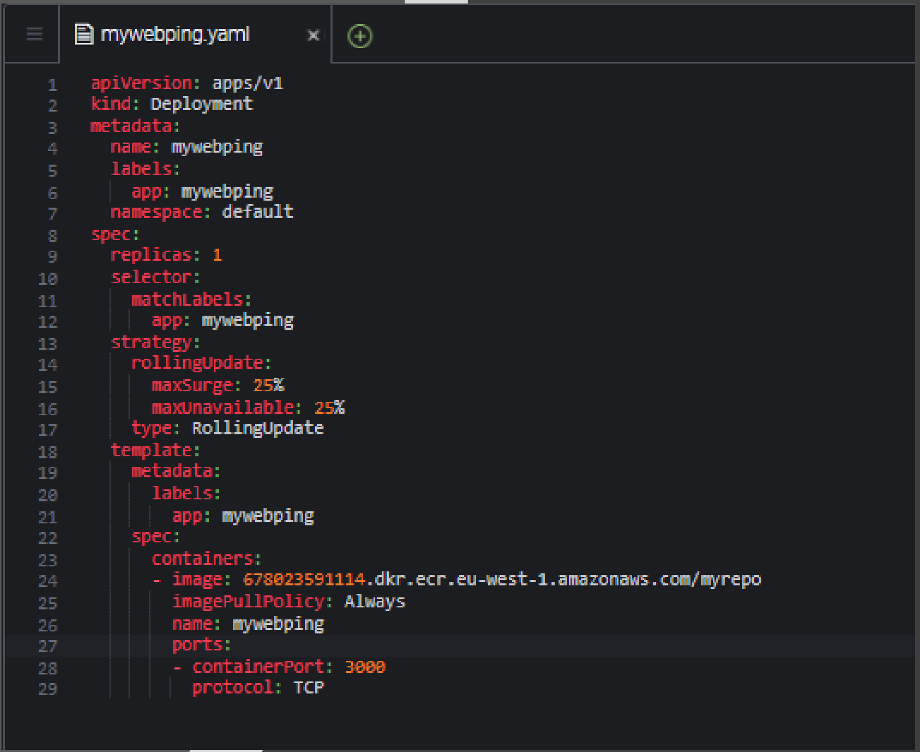

Write code
Write and package code as a Docker image
Amazon ECR
Compress, encrypt, and control access to images
Version, tag, and manage image lifecycles
Run containers
Pull images and run containers anywhere
Amazon ECS
Amazon EKS
AWS Cloud
On premises

Image 4: Amazon ECR



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mywebping
  labels:
    app: mywebping
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mywebping
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: mywebping
    spec:
      containers:
      - image: 678023591114.dkr.ecr.eu-west-1.amazonaws.com/myrepo
        imagePullPolicy: Always
        name: mywebping
        ports:
        - containerPort: 3000
          protocol: TCP
```

$kubectl apply –f mywebping.yaml
-> deployed

annotation:
assigning deployments / pods to a certain node possible by specificating this in deployment yaml file

3.5 RBAC Permission for EKS Cluster

I have a k8s joke, however i am not authorized

- Security Topic
- Role-based access control (RBAC): method of regulating access to compute/network ressources
- EKS uses IAM to provide authentication to EKS Cluster, but still **relies on native Kubernetes RBAC**
- Rbac is managed kubernetes api server, api server know all ressources out there in k8s environment,
- = set of permissions
- Two different types:
    - Role: dedicated to a specific namespaces
    - ClusterRole: clusterwide

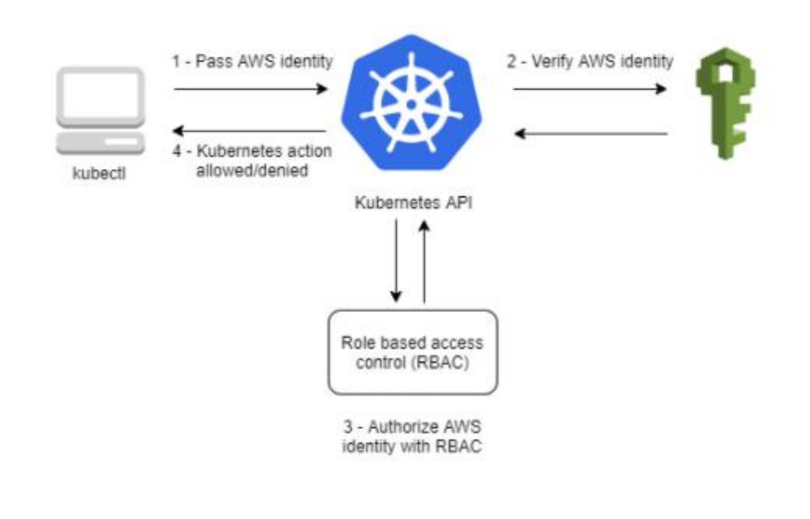

kubectl
1 - Pass AWS identity
4 - Kubernetes action allowed/denied
Kubernetes API
2 - Verify AWS identity
Role based access control (RBAC)
3 - Authorize AWS identity with RBAC

Image 5: Cluster Authentication

3.6 Monitoring (K8s, Grafana, Prometheus)

Definition: Monitoring = „surveillance process on measurable events and outputs of a system“

- Helm: package manager and appl. Management tool for K8s -> packages ressources into a **Chart**

Kubernetes-Dashboard:

- start proxy
- listen on port 8080
- access dashboard with aws cluster token

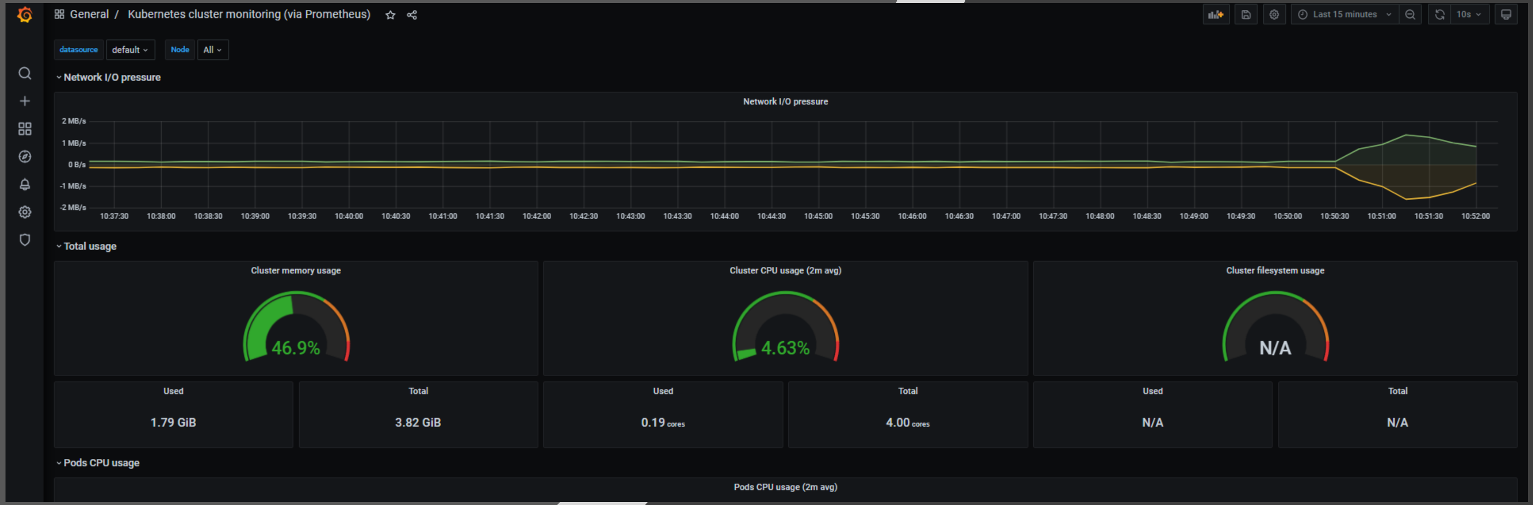
Prometheus:

- open-source sytems monitoring and alerting toolkit
- monitoring service for metrics
- is running in a dedicated namespaces as several pods

Grafana:

- open source visualization and analytics SW
- query, visualize, alert on and explore metrics
- -> beautiful graphs and visualization

all three monitoring services are running as pods on „myEKS“ cluster



General / Kubernetes cluster monitoring (via Prometheus)

datasource: default

Node: All

Last 15 minutes

10s

Network I/O pressure

Total usage

Cluster memory usage		Cluster CPU usage (2m avg)		Cluster filesystem usage	
46.9%		4.63%		N/A	
Used	Total	Used	Total	Used	Total
1.79 GiB	3.82 GiB	0.19 cores	4.00 cores	N/A	N/A

Pods CPU usage

Pods CPU usage (2m avg)

Screenshot1: Grafana

3.7 Logging with CloudWatch

Amazon CloudWatch

Definition: Logging = automatic creation of a protocol (=log) of software processes

1. Logging of „myEKS“ Cluster

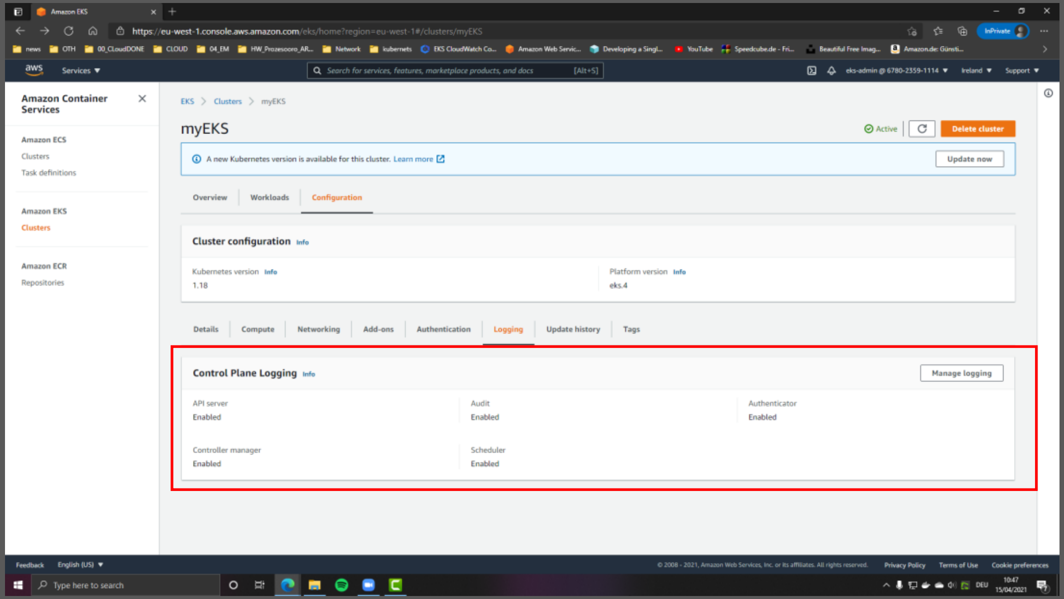
- API Server
- Audit
- Authenticator
- Controller Manager
- Scheduler

Logs

2. CloudWatch Container Insights

- required: Security Account, ClusterRole, RoleBinding
- CloudWatch-Agent: responsible for sending metrics to CloudWatch
- Fluentd: responsible for sending logs to CloudWatch

- Terms:
  - Logs: only „log“ events stream
  - Metrics: get a view of the state of applications



myEKS

A new Kubernetes version is available for this cluster. Learn more

Cluster configuration Info

Kubernetes version: 1.18

Platform version: eks.4

Control Plane Logging Info

API server: Enabled

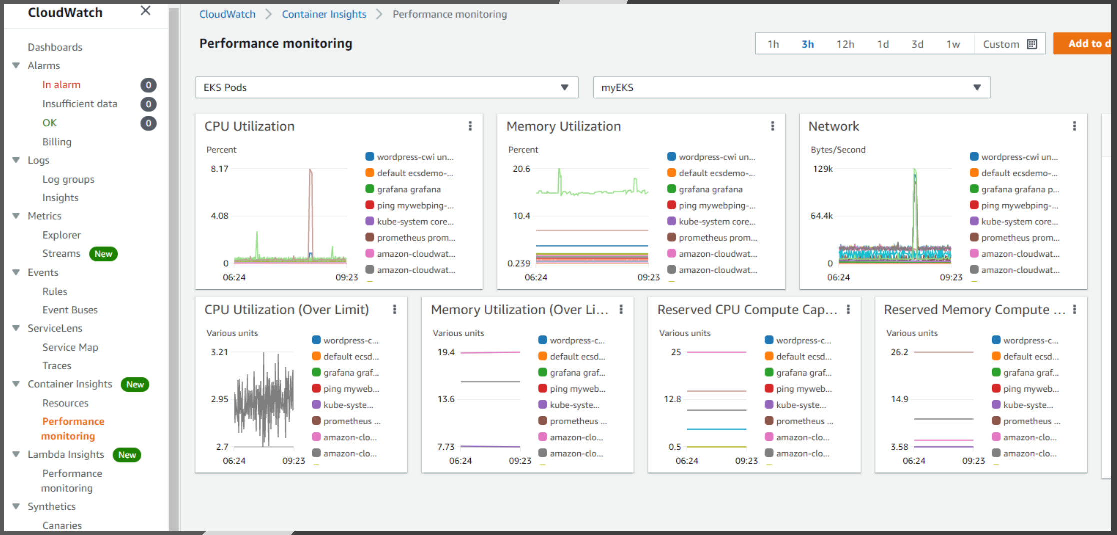
Audit: Enabled

Authenticator: Enabled

Controller manager: Enabled

Scheduler: Enabled

Screenshot2: Logging EKS Cluster



Screenshot3: Logging Container Insights

4. Summary & Outlook

Summary:

- Big challenge to deal with all different AWS Services, espially IAM at first
- Kubernetes was a very complex topic, which I needed many hours to get an idea how it works (and Docker Containers)
- Lots of documentation available, but only one is easy to understand for newcomer: https://www.eksworkshop.com
- All in all very challenging but worth spending time

Outlook:

- instead of Prometheus and Grafana running as pods on the cluster -> using AMP and AMG
- using Fargate (serverless, as replacement for managed node worker groups)
- AWS CloudTrail, AWS X-Ray

AWS Fargate

AWS X-Ray

AWS CloudTrail

Amazon Managed Service for Grafana

Amazon Managed Service for Prometheus

Sources

1. Logos
   1. https://aws.amazon.com/architecture/icons/
   2. https_//github.com/kubernetes/community/tree/master/icons
   3. https://helm.sh/

2. AWS
   1. https://www.eksworkshop.com
   2. https://docs.aws.amazon.com/eks/latest/userguide/getting-started.html
3. Kubernetes
   1. https://kubernetes.io/docs/tasks/access-application-cluster/web-ui-dashboard/
   2. https://kubernetes.io/docs/reference/access-authn-authz/rbac/
4. Others
   1. https://www.igi-global.com/dictionary/system-monitoring/29068
5. Images
   1. Image 1: https://www.eksworkshop.com/010_introduction/architecture/architecture_control_and_data_overview/
   2. Image 2: https://aws.amazon.com/de/eks/?whats-new-cards.sort-by=item.additionalFields.postDateTime&whats-new-cards.sort-order=desc&eks-blogs.sort-by=item.additionalFields.createdDate&eks-blogs.sort-order=desc
   3. Image 3: https://www.eksworkshop.com/010_introduction/eks/eks_high_architecture
   4. Image 4: https://aws.amazon.com/de/ecr/
   5. Image 5: https://docs.aws.amazon.com/eks/latest/userguide/managing-auth.html
6. Screenshots: refers to my implemented EKS Cluster
7. Books
   1. Learn Docker in a Month of Lunches – Elton Stoneman, publisher: Manning
   2. Learn Kubernetes in a Month of Lunches – Elton Stoneman, publisher: Manning