

# BUILDING A SCALABLE CLOUD NATIVE TRAINING PLATFORM

with Kubernetes

Or why pods are not VM's





# Whoami

---

**Sofus Albertsen**

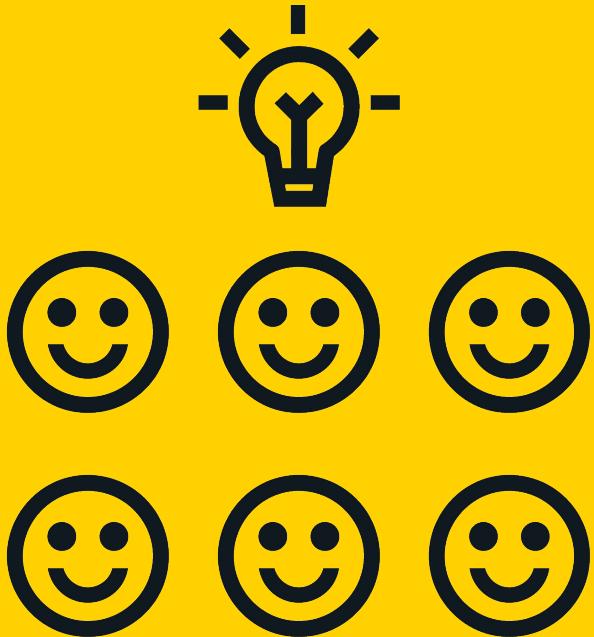
Consultant, Trainer, Speaker

# We don't chop wood

# We sharpen axes

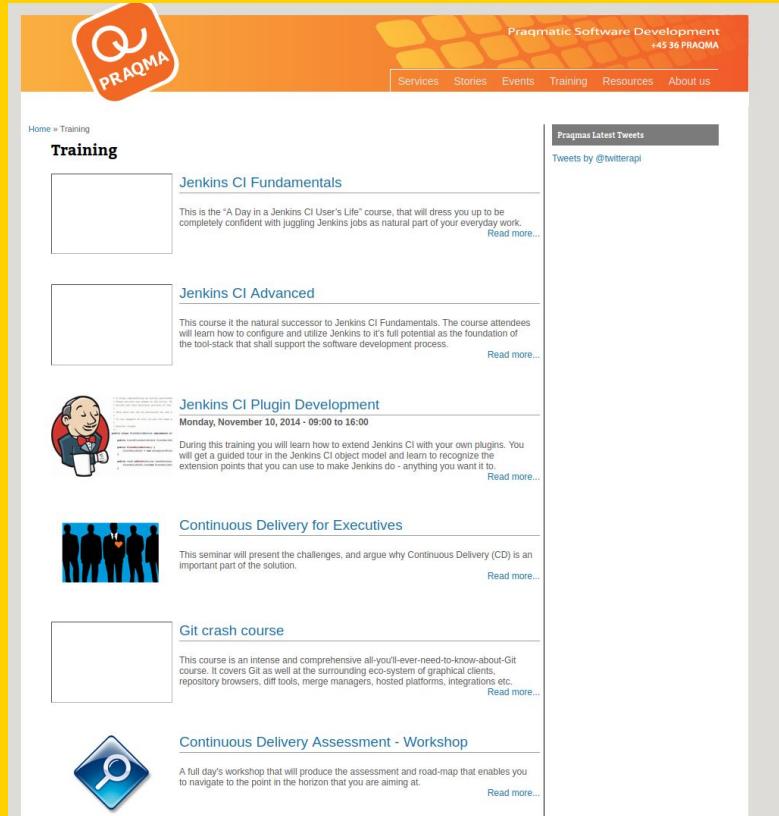


# Setting the scene



# The History

10+ years of training



The screenshot shows the Praagma website's training section. The header features the Praagma logo (a red diamond with a white 'P' and 'RAQMA') and navigation links for Services, Stories, Events, Training, Resources, and About us. A sidebar on the right displays "Praagmas Latest Tweets" with a link to @twitterapi. The main content area lists several training courses:

- Jenkins CI Fundamentals**: Described as "A Day in a Jenkins CI User's Life" course. Includes a small placeholder image and a "Read more..." link.
- Jenkins CI Advanced**: Described as the natural successor to Jenkins CI Fundamentals. Includes a small placeholder image and a "Read more..." link.
- Jenkins CI Plugin Development**: Scheduled for Monday, November 10, 2014, from 09:00 to 16:00. Features an illustration of a person holding a coffee cup. Includes a small placeholder image and a "Read more..." link.
- Continuous Delivery for Executives**: Described as a seminar presenting challenges and arguing why Continuous Delivery (CD) is important. Includes a small placeholder image and a "Read more..." link.
- Git crash course**: Described as an intense and comprehensive course on Git. Includes a small placeholder image and a "Read more..." link.
- Continuous Delivery Assessment - Workshop**: Described as a full-day workshop producing an assessment and road-map. Features an illustration of a magnifying glass. Includes a small placeholder image and a "Read more..." link.

# **ENTER WESL -**

**Windows**

**Enterprise**

**Standard**

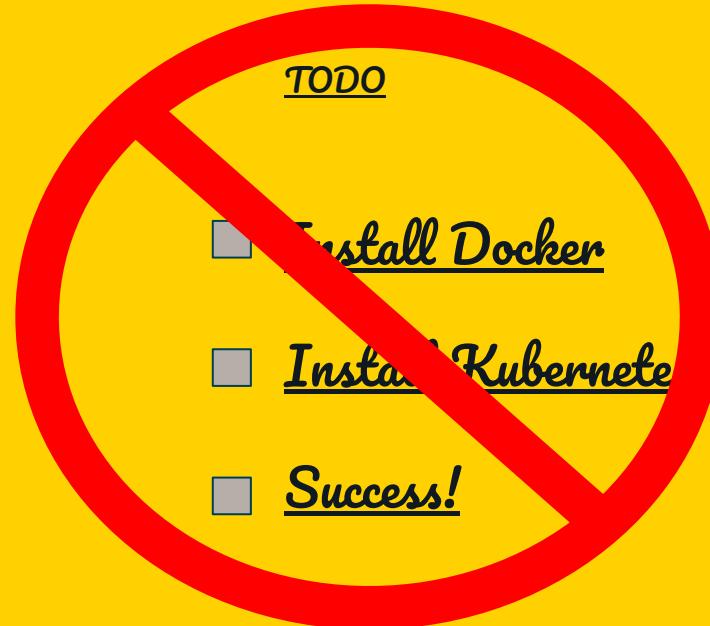
**Laptop**



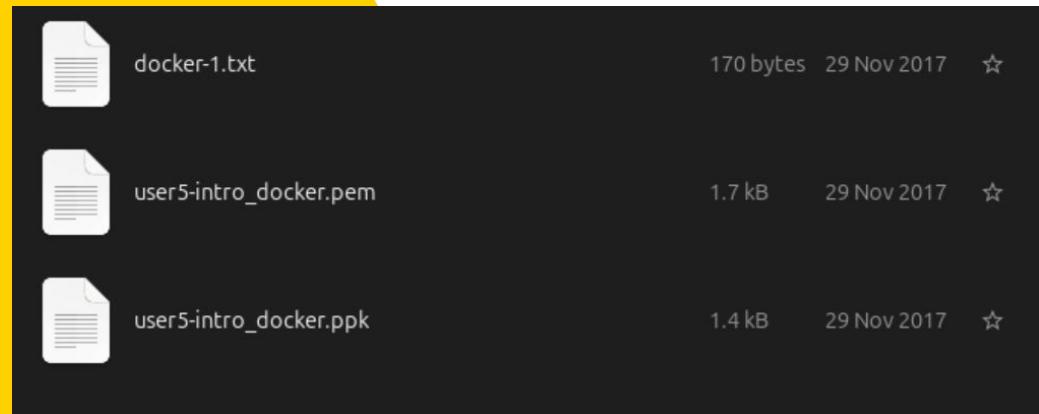
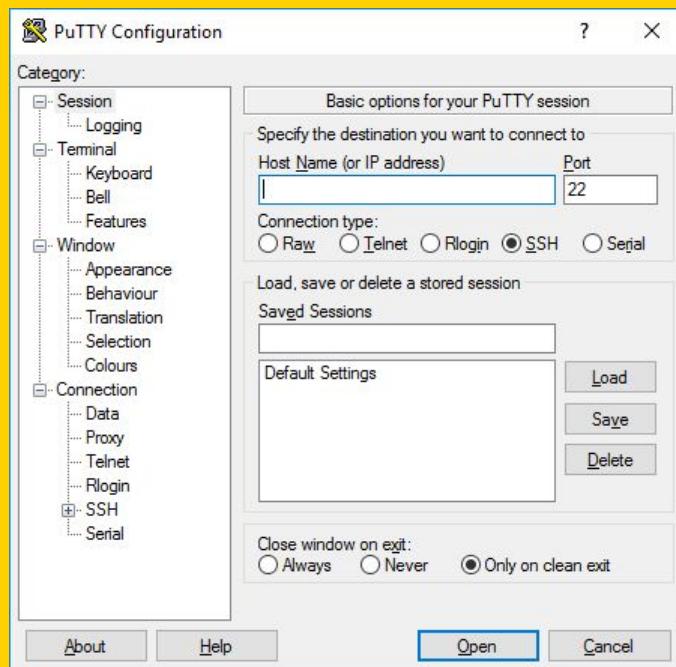
# WESL - the perfect fit

## TODO

- Install Docker
- Install Kubernetes
- Success!



# ... And you get a machine! And a lesson in PuTTY and SSH



# IaC is king!



- Infrastructure as code (!)
  - Layered, modular approach
  - EKS cluster got included
- Introduction to URL!
- Visual studio code in a browser
- 100% connection rate within 10-15 minutes of setup

What could possibly go wrong?

The screenshot shows a VS Code interface with the following details:

- Explorer View:** Shows a folder named "KUBERNETES-KATAS" containing files like ".gitignore", "accessing-your-applicat...", "cheatsheet.md", "configmaps-secrets.md", "CONTRIBUTION.md", "deployments-ingress.md", "desired-state.md", "exercise-template.md", "intro.md" (which is selected), "manifests.md", "persistent-storage.md", "README.md", "rolling-updates.md", "services.md", and "trainer-notes.md".
- Terminal View:** Displays the command `$ kubectl get no` followed by a table of nodes:

NAME	AGE	VERSION	STATUS	ROLES
ip-192-168-75-16.eu-north-1.compute.internal	11h	v1.29.6	Ready	<none>
ip-192-168-88-196.eu-north-1.compute.internal	6h46m	v1.29.6	Ready	<none>
ip-192-168-94-226.eu-north-1.compute.internal	11h	v1.29.6	Ready	<none>
- Status Bar:** Shows the current branch as "master\*", the file count as "0 0", and other status indicators.

# And they lived happily ever after???

- It was magnificent!
- It was secure!
- It was....

Totally unmaintainable!

## Old implementation

Language	files	blank	comment	code
JSON	3	5	0	2925
HCL	44	366	52	1998
YAML	17	43	123	1815
Bourne Shell	21	285	56	1416
Markdown	6	229	0	595
Python	1	65	48	235
make	3	55	7	205
Dockerfile	1	0	1	11
SUM:	96	1048	287	9200

# And they lived happily ever after???



- It was magnificent!
- It was secure!
- It was....

Totally unmaintainable!

**Old implementation**

Language	files	blank	comment	code
JSON	3	5	0	2925
HCL	44	366	52	1998
YAML	17	43	123	1815
Bourne Shell	21	285	56	1416
Markdown	6	229	0	595
Python	1	65	48	235
make	3	55	7	205
Dockerfile	1	0	1	11
SUM:	96	1048	287	9200

**New implementation**

Language	files	blank	comment	code
YAML	62	112	78	1769
Bourne Shell	51	353	219	1177
JSON	7	0	0	313
Python	3	46	47	167
Dockerfile	5	37	35	133
Markdown	2	57	0	122
Bourne Again Shell	3	43	69	93
HTML	1	0	0	10
Text	1	0	0	7
SUM:	135	648	448	3791

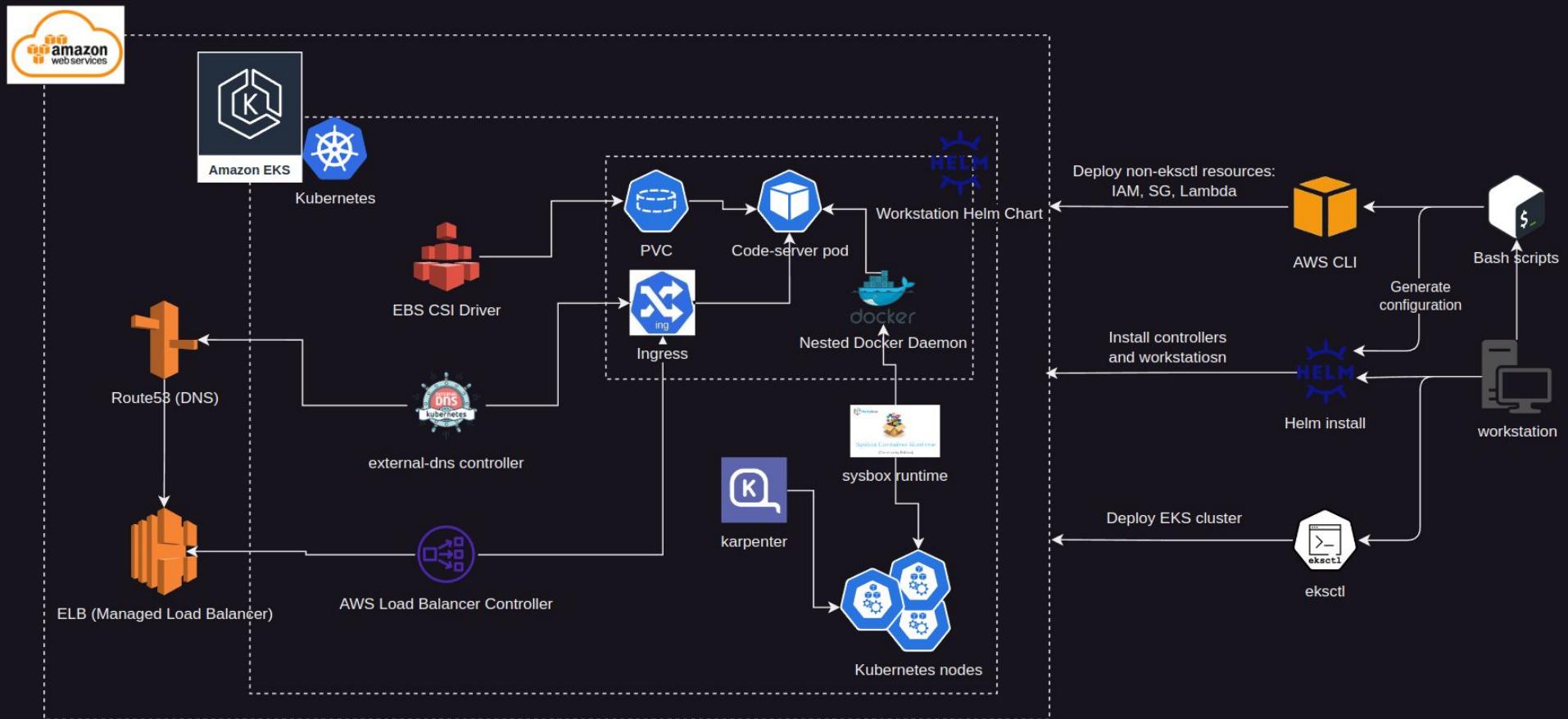


# The task

- User:
  - Look and feel as a VM to the students.
- Infrastructure:
  - We wanted a platform build on top of Kubernetes.
  - Must work with all existing courses.
  - All additions as cloud native implemented as possible.
  - Everything should be declarative --> avoid running scripts to configure things.

# The landscape

eficode



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

metadata:
  name: "soal-cluster"
  region: "eu-north-1"
  version: "1.29"

iam:
...
addons:
- name: coredns
  version: "latest"
- name: vpc-cni
  version: "latest"
- name: kube-proxy
  version: "latest"
- name: aws-ebs-csi-driver
  version: "latest"

nodeGroups:
- name: "soal-cluster-default-workers"
  amiFamily: Ubuntu2004
  instanceType: "t3.xlarge"
  desiredCapacity: 2
  availabilityZones: ["eu-north-1a"]
  labels:
    owner: "sofus.albertsen"
    role: "workers"
    sysbox-install: "yes"
```



# eksctl

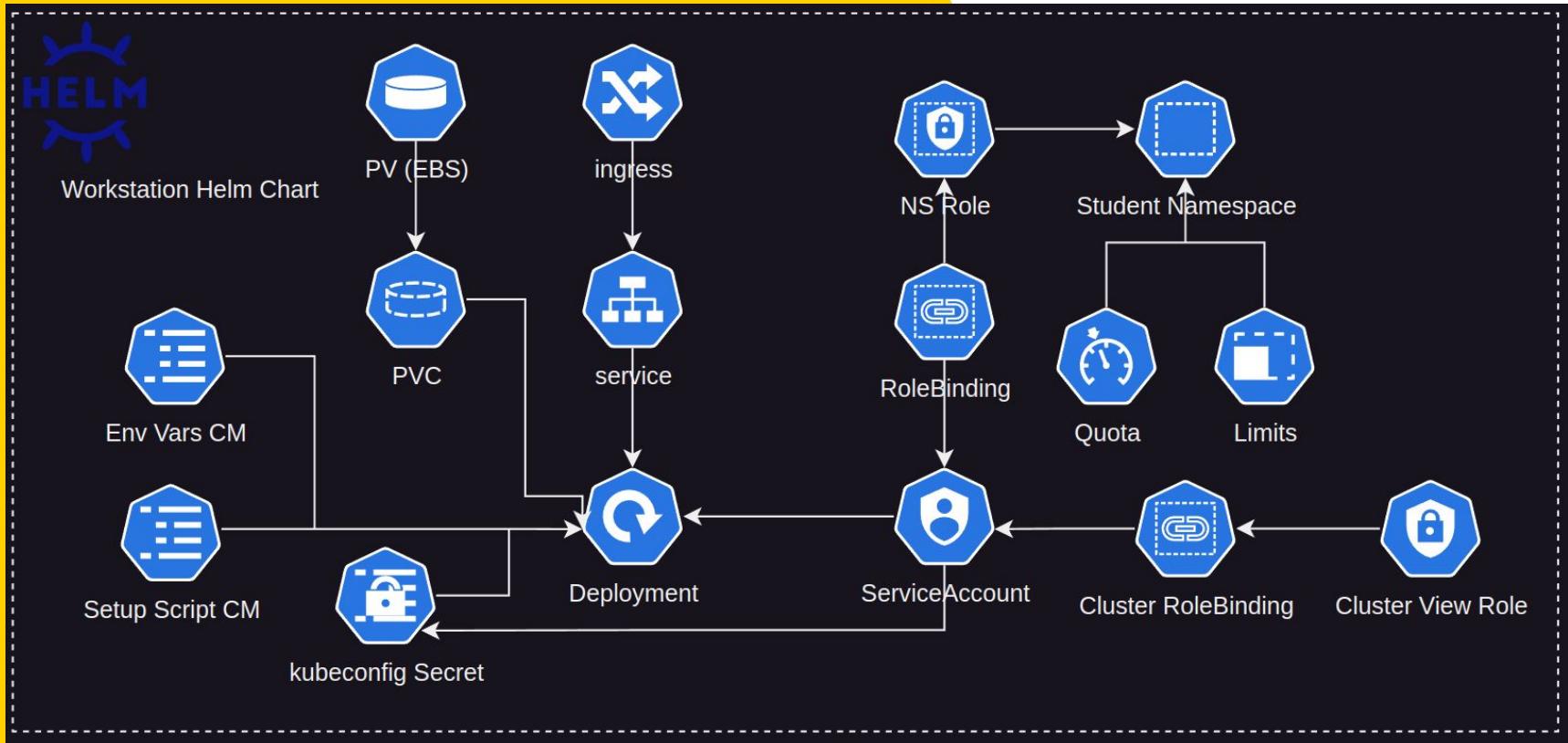
- Simplified EKS Cluster Management
- Infrastructure as Code
- Opinionated (read: less config than Terraform)

# Helm charts

```
# flags for tools
INSTALL_ARGO_CD="true"
INSTALL_ISTIO="false"
INSTALL_ARTIFACTORY="false"
FIREFOX_IN_DOCKER="false"
```



# The workstation



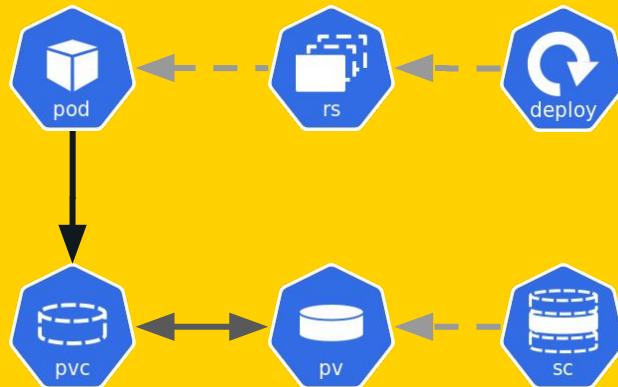
# Demo

# The struggles

Or why pods are not VM's



# Persistency



```
setup.sh: |
#!/usr/bin/env bash

FLAG_FILE="/home/coder/.setup_done"

if [ -f "${FLAG_FILE}" ]; then
    echo "Setup has already been performed. Skipping..."
    exit 0
fi

echo =====
echo "Running setup script"
echo =====

echo "--> copying initial /home/coder/ files back from
backup"
sudo cp -r /coder-initial-home/coder /home/
...

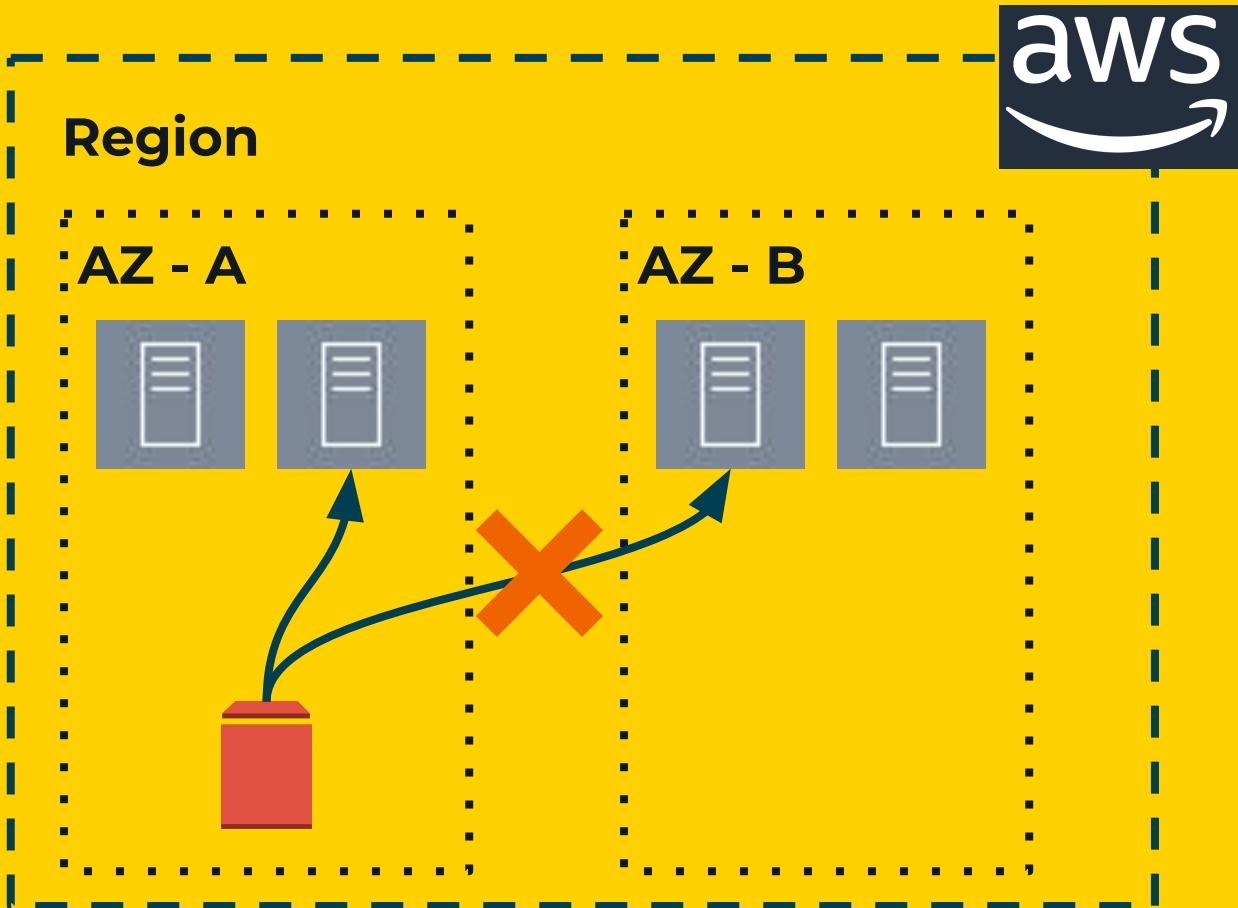
# Touch the flag file to indicate that the setup has been
performed
touch ${FLAG_FILE}

echo =====
echo "Done running setup script"
echo =====
```

# Persistency

- PVC != Named Volumes
- What is in container gets deleted
- Needs trickeries to function

# EBS != Region



eficode

Solution: All nodes in one AZ

# Ingress

**100 workstations x 100 ports**

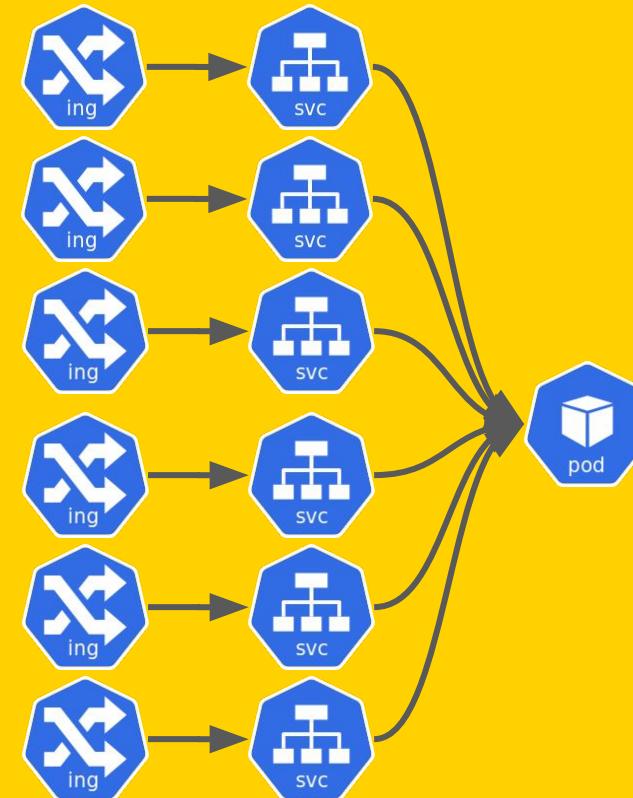
=

**10.000 Service +  
10.000 Ingress**

=

**10.000 security groups in AWS  
...And how many load balancers?**

**Solution: Create only a few ports**



# Docker in docker



Docker in docker



Docker outside of docker

Dind?

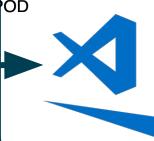
DouD?

**Douh?**

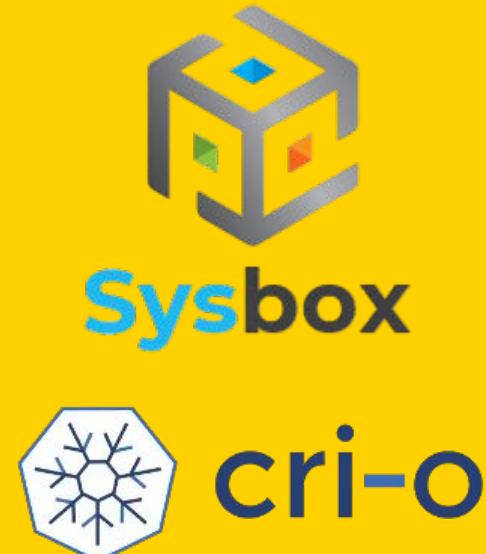
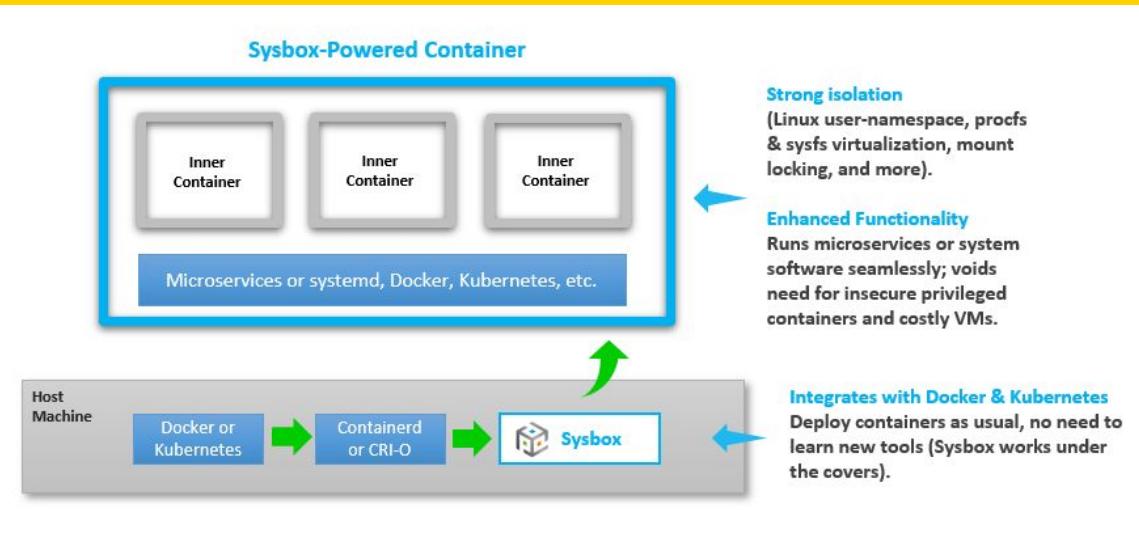
We need help.....

```
docker run --name  
hej alpine
```

```
docker run --name  
hej alpine
```



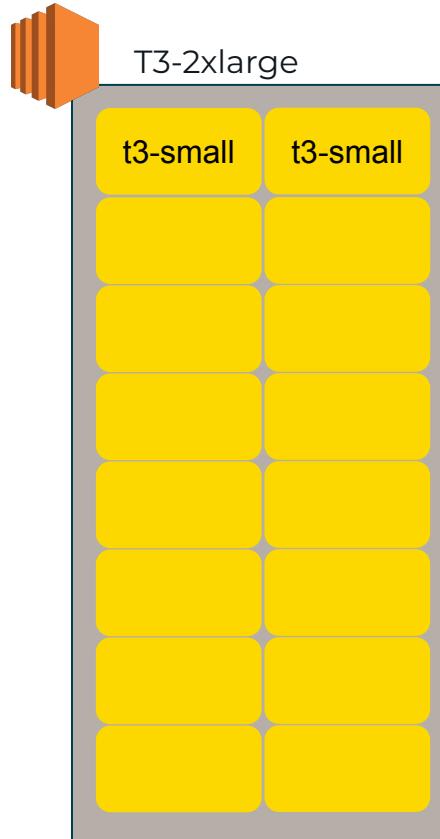
# Docker in docker



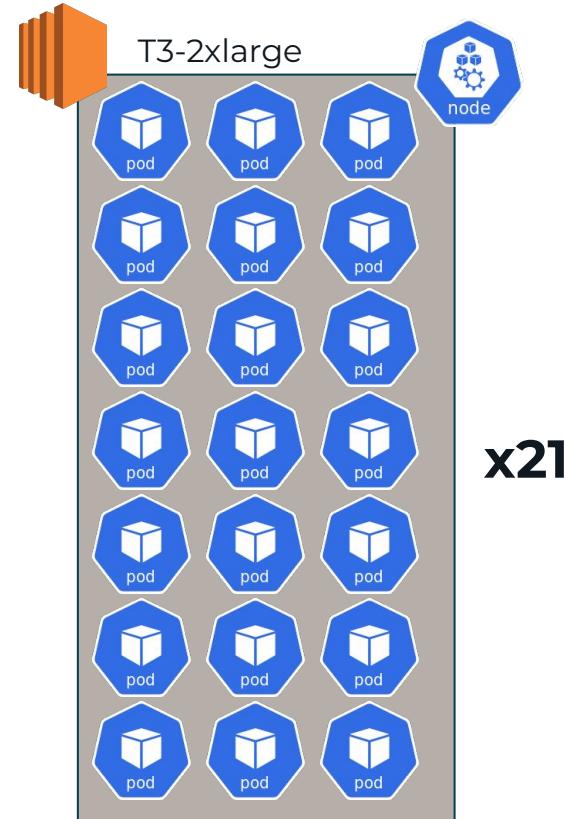
# The result

- Looks and feels like a VM!
- Less unstable
- Easier to use
- Easier to maintain\*





User= 2 Gigs mem, 2 CPU MAX



User= 3 Gigs mem, 6 CPU MAX

# Todo



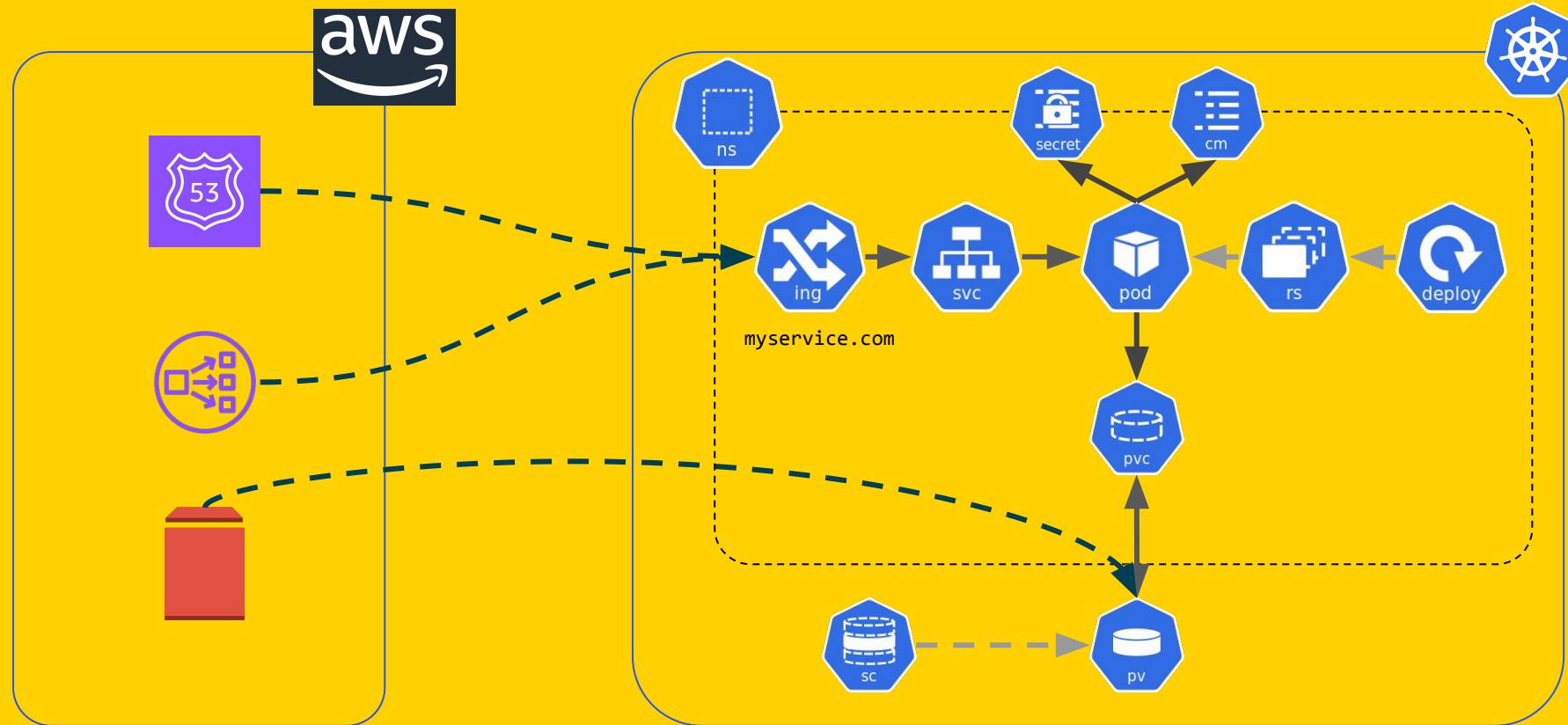
Who run the world?

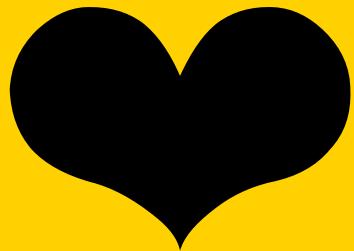
# Todo



# Todo

eficode





Reach out:

<https://www.linkedin.com/in/sofusaibertsen/>

Thank you ❤