CORD Development

How the New Platform Will Make Your Life Better

December 6, 2018

Topics

- Docker, Kubernetes, Helm - Let's clear some concepts
- Development workflow
  - Find an issue in the code
  - Fix it and test it
  - Deploy it on a running POD

Do you *really* know

what **Docker** is?

Docker

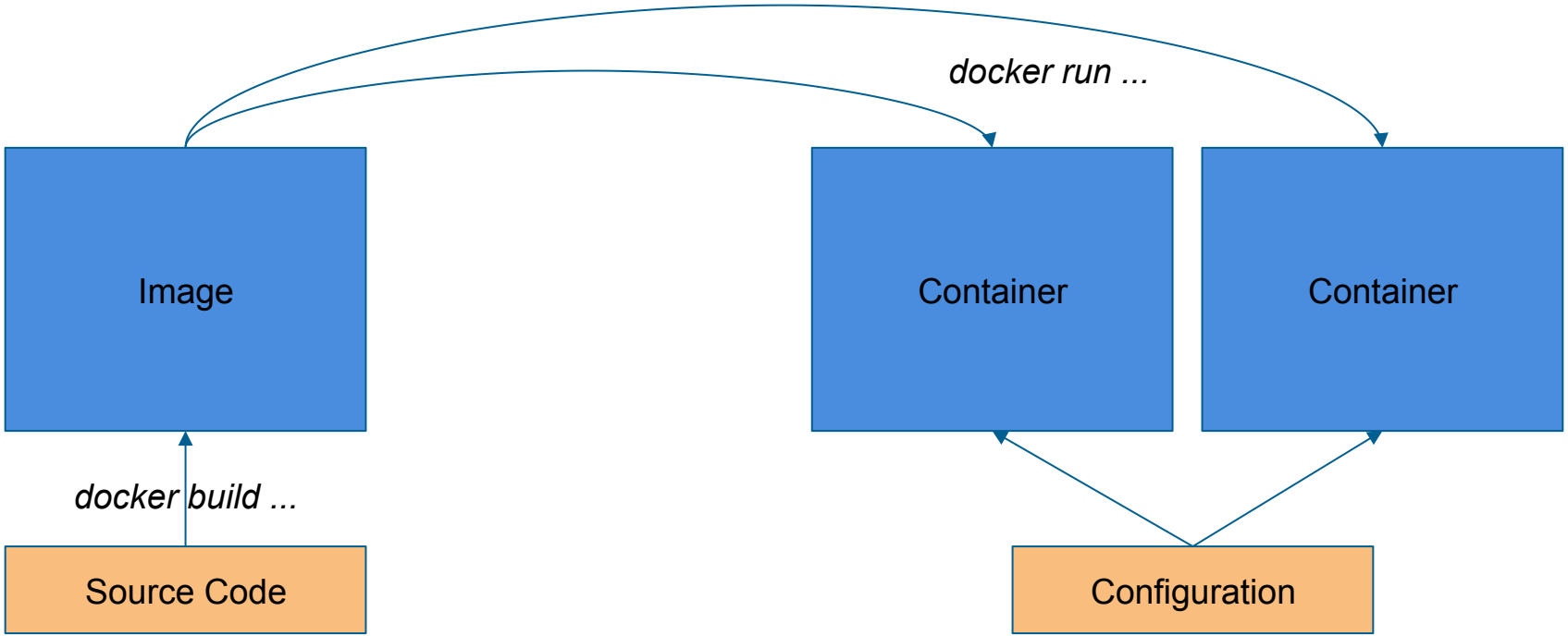
... performs *operating-system-level virtualization*,

also known as "containerization" ...

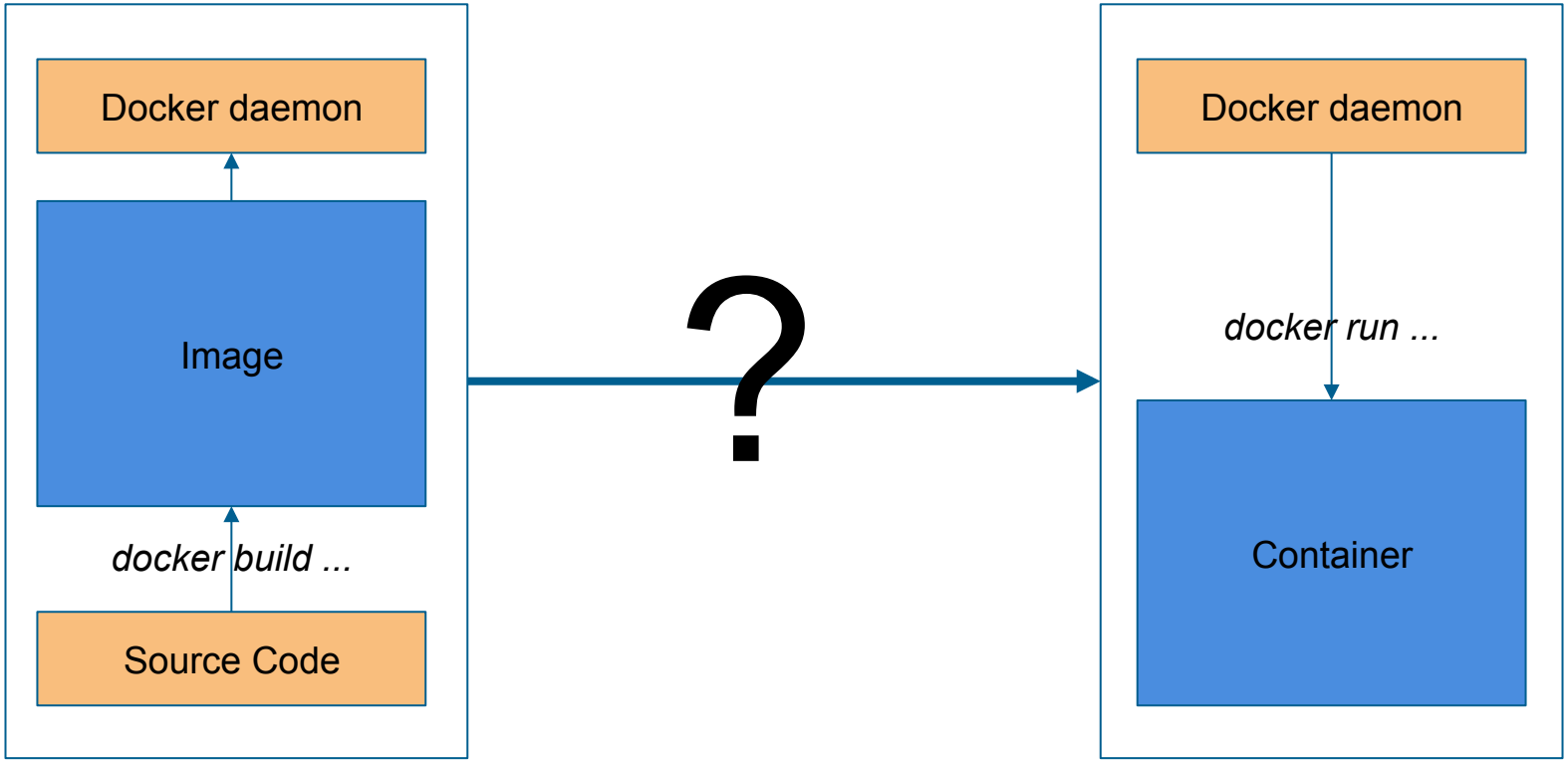
Applications are bundled in containers, together with their own tools, libraries and configuration files.

They can communicate with each other through well-defined channels

Docker Image vs Docker Container



Docker Images

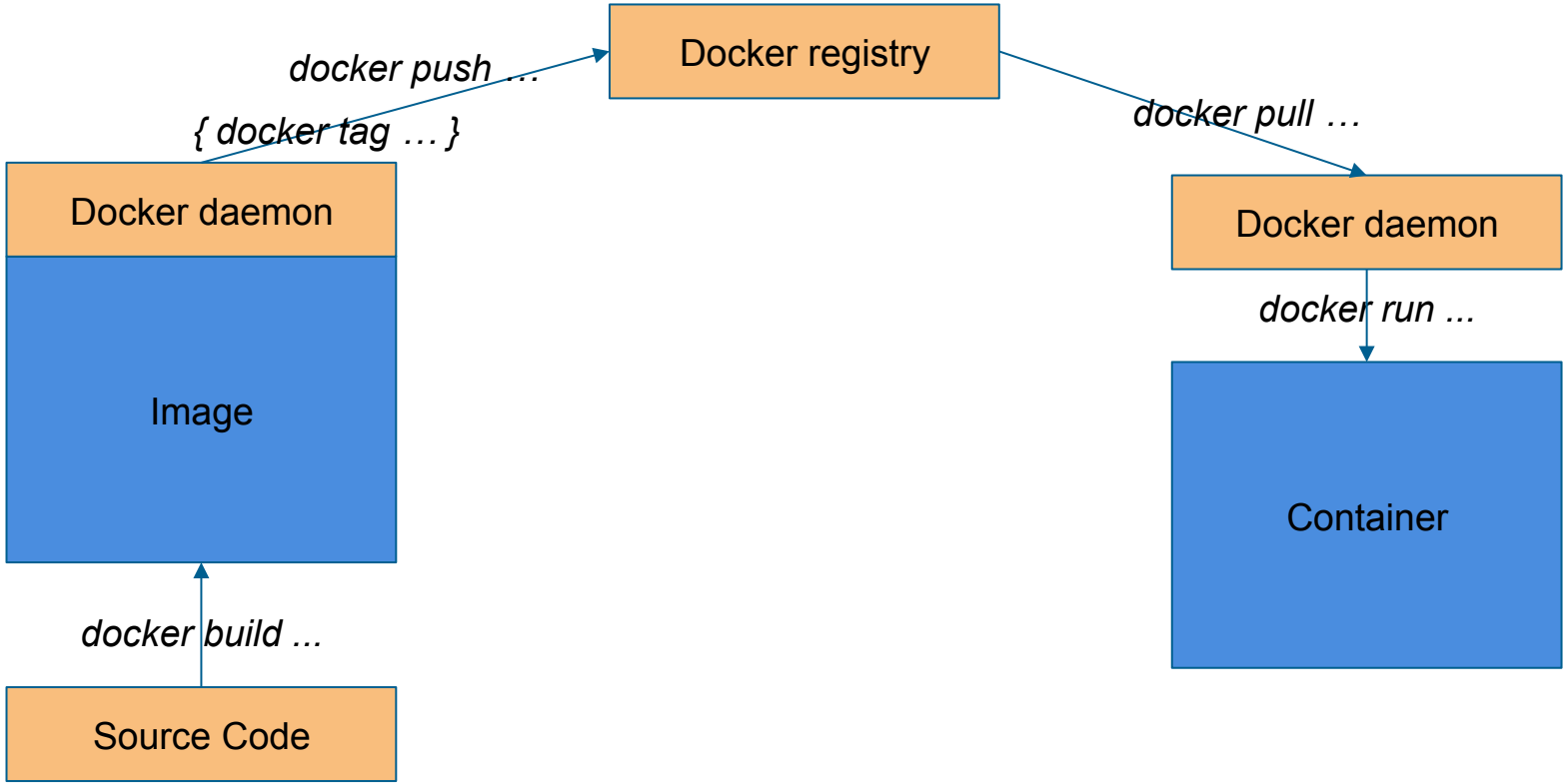


Docker Registry

A catalog of Docker Images

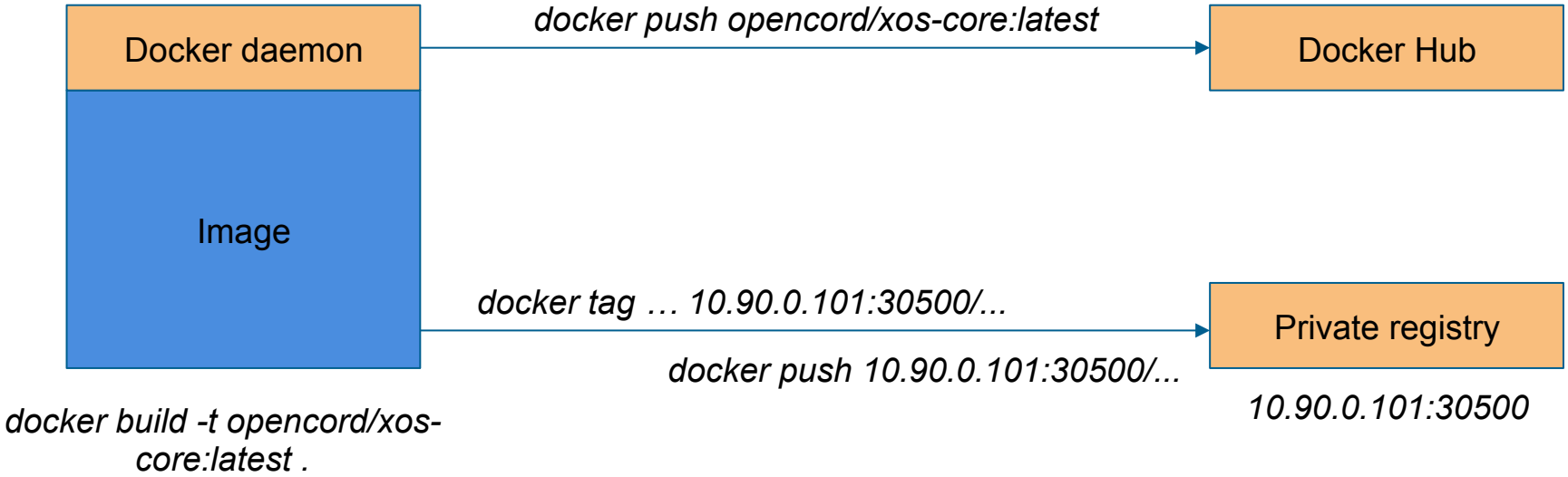
- public (available over the internet)
- private (running on premises)

Docker Registry



Docker image names

{registry}/organization/image:{tag|latest}



Images tagging

Git	Docker
master	latest
branch	eg: 1.6 by project
tag	1.6.0

semver is your friend!

Kubernetes

… automating deployment, scaling, and management of containerized applications …

Kubernetes

- Runs in cluster
  - don’t have to worry on where your containers are running
- Manages mounted files
  - don’t have to worry about deployment specific configurations
- Handles the networking (L3 and above)
  - don’t have to worry about service to service communication

Helm

The package manager for kubernetes

Helm

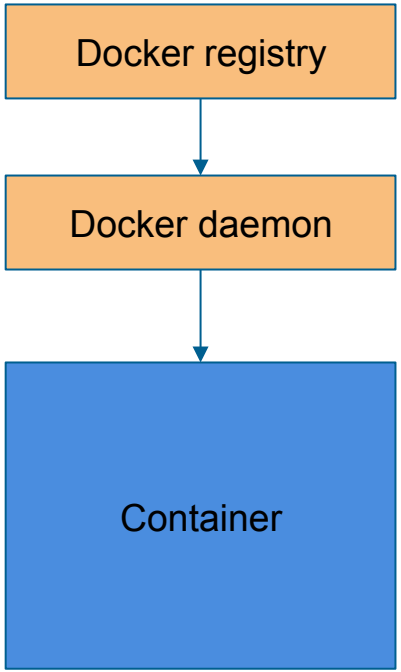
- Define applications as charts
- Templatize data and configuration files
- Manage container lifecycle
  - upgrades
  - scale

Helm and Kubernetes - Images and Containers

```
containers:
      - name: onos
        image:  "{{ .Values.images.onos.repository }}:{{ tpl .Values.images.onos.tag . }}"
        imagePullPolicy: {{ .Values.images.onos.pullPolicy }}
```

```
images:
  onos:
    repository: 'onosproject/onos'
    tag: 'latest'
    pullPolicy: 'Always'
```

ImagePullPolicy: Always



- helm install …
- helm upgrade … ***
- kubectl delete pod …

Anytime the container restart the image is pulled from the registry

ImagePullPolicy: Always

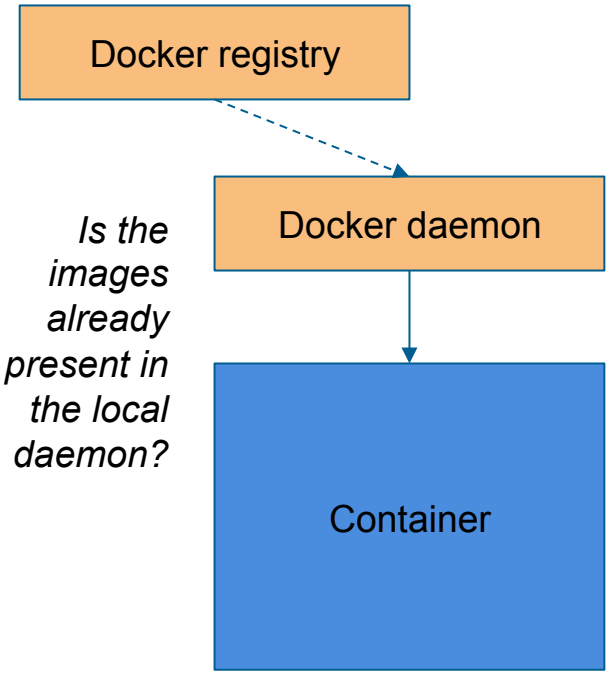
Pros:

- you are sure it is always getting the latest available image

Cons:

- the image can change if the upstream one changes (eg: latest)

ImagePullPolicy: IfNotPresent



Anytime the container restart the daemon checks if the images is present:

- if yes it uses it
- if not it downloads it

ImagePullPolicy: IfNotPresent

Pros:

- faster
- doesn't download newer image
- easier to update an image for dev *

Cons:

- it won't pull a new version of the image (eg: latest)
- can end up with different images in different nodes **

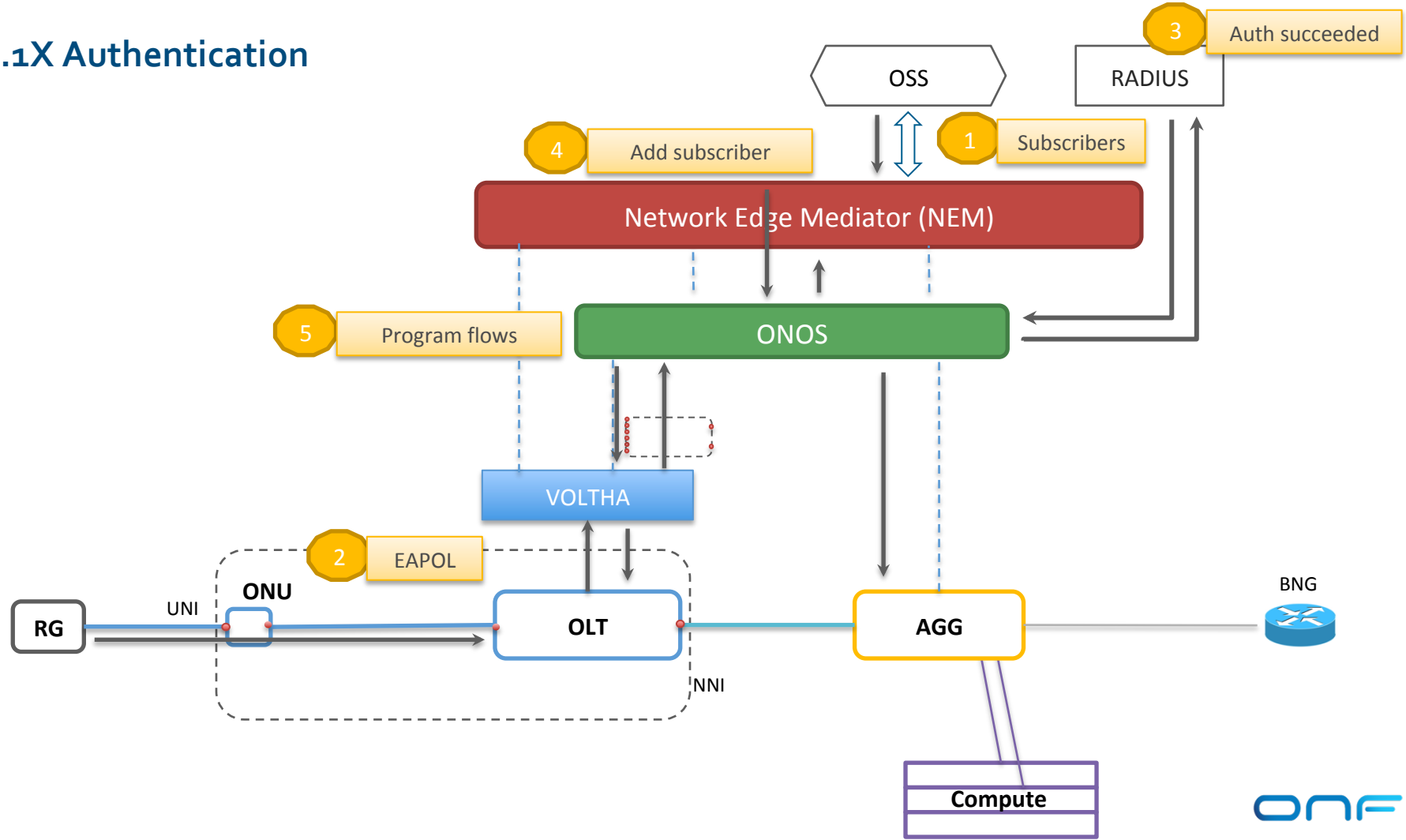
Always VS IfNotPresent: When to use what

	Development	Production / Trial
Single Node	**IfNotPresent**	**Al**
Cluster	**Always**	**ys**

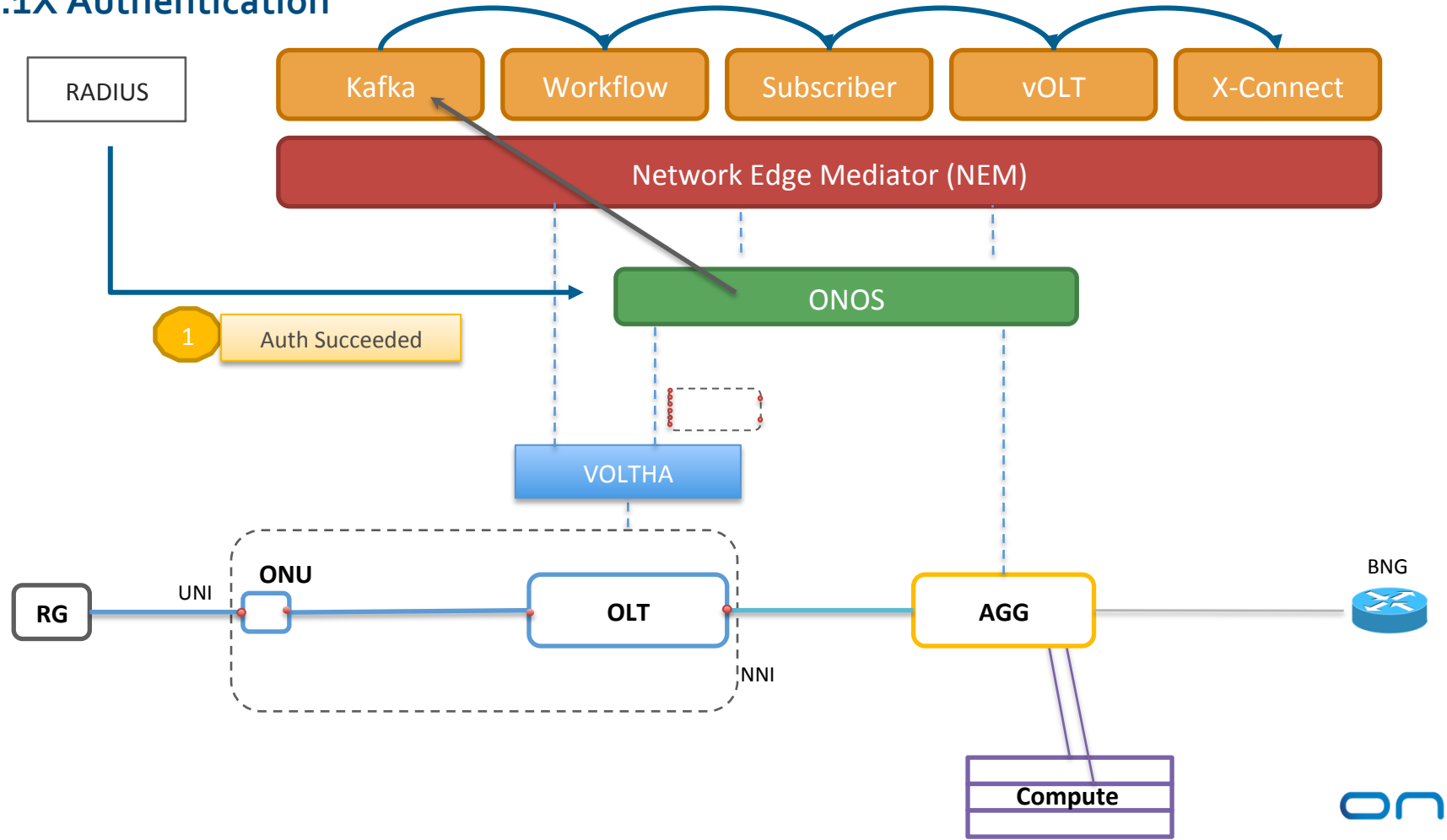
Let’s practice

- running multi-node development environment
  - private docker registry
  - imagePullPolicy: Always
- authenticate a subscriber
  - find an error
  - fix the error
  - update the environment

802.1X Authentication



802.1X Authentication



Thanks!

Any question?

References

- https://en.wikipedia.org/wiki/Docker_(software)
- https://kubernetes.io
- https://helm.sh
- https://guide.opencord.org