The 12 factors app

https://12factor.net/

https://en.wikipedia.org/wiki/Twelve-Factor_App_methodology

Immutable infrastructure

https://www.digitalocean.com/community/tutorials/what-is-immutable-infrastructure

“With immutable infrastructure, once an artifact is created in the system it does not change via user modifications.”

Infrastructure as code

https://en.wikipedia.org/wiki/Infrastructure_as_code

Learning Docker & K8S

Plenty of online resources, videos, courses… Google is your friend

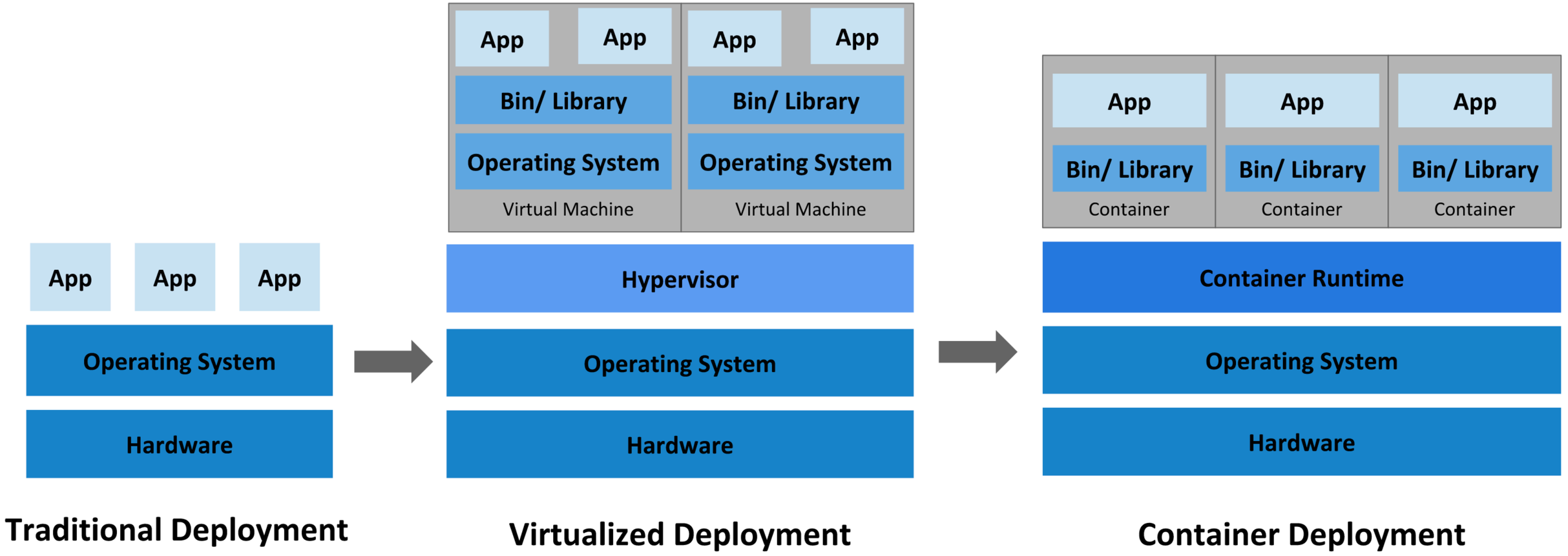
https://kubernetes.io/docs/tutorials/kubernetes-basics/

https://www.digitalocean.com/community/tutorials/an-introduction-to-kubernetes

https://learning.oreilly.com/library/view/kubernetes-up-and/9781492046523/

And more advanced resources:

CERN K8S seminars: https://indico.cern.ch/category/8202/



https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

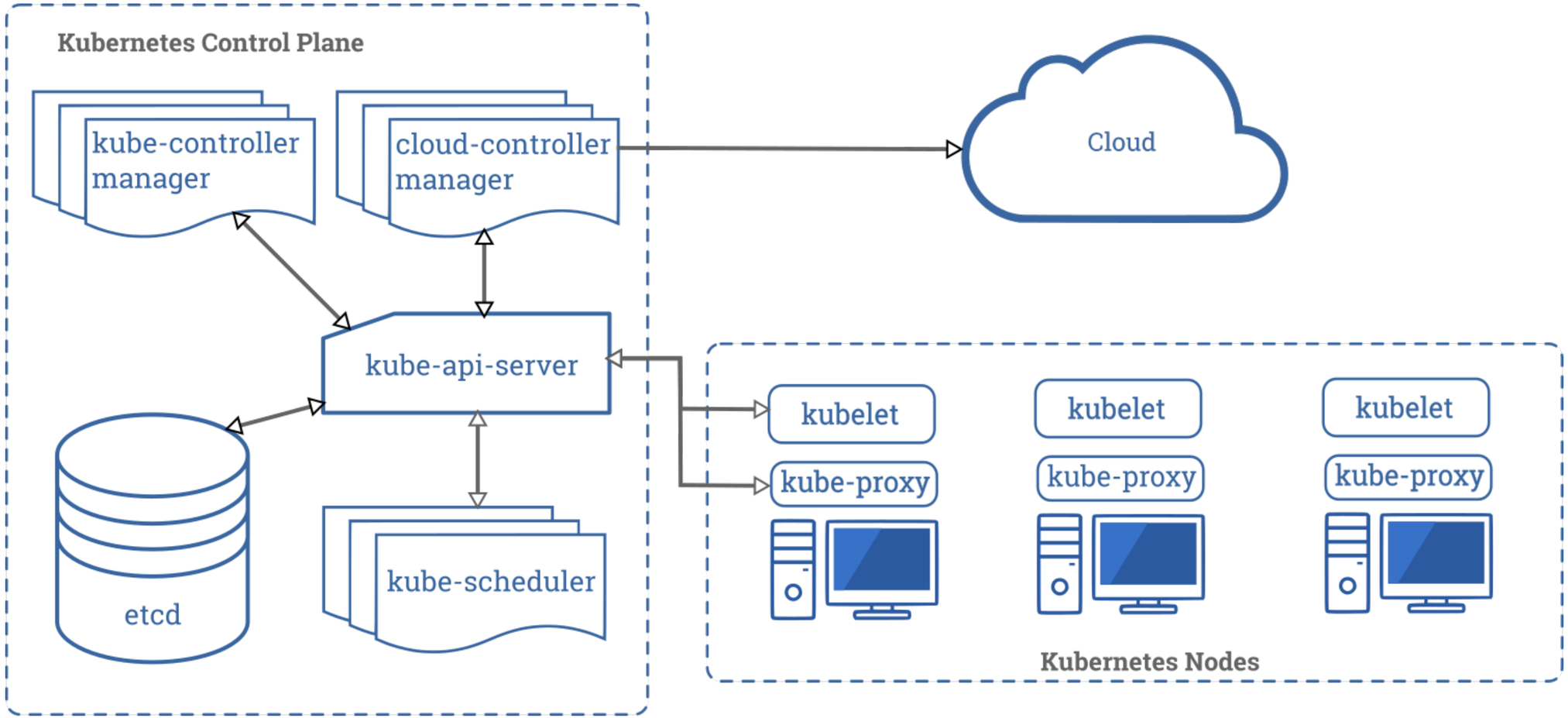
Declarative vs imperative

https://thenewstack.io/kubernetes-design-and-development-explained/

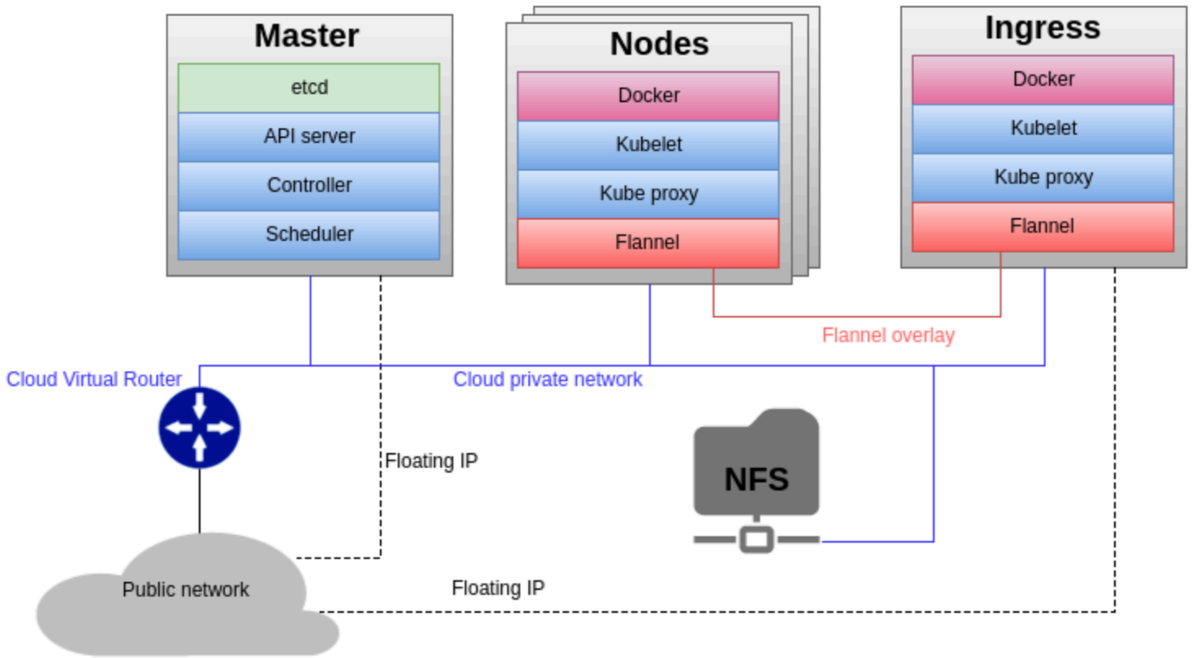
In an imperative API, you directly issue the commands that the server will carry out, e.g. “run container,” “stop container,” and so on.

In a declarative API, you declare what you want the system to do, and the system will constantly drive towards that state.

K8S components



Our little K8S cluster




```
$ kubectl get nodes
NAME           STATUS   ROLES    AGE    VERSION
kube-lb01      Ready    <none>   288d   v1.10.2
kube-node01    Ready    <none>   288d   v1.10.2
kube-node02    Ready    <none>   288d   v1.10.2
kube-node03    Ready    <none>   288d   v1.10.2
kube-node04    Ready    <none>   288d   v1.10.2
kube-node05    Ready    <none>   288d   v1.10.2
kube-node06    Ready    <none>   288d   v1.10.2
kube-node07    Ready    <none>   288d   v1.10.2
kube-node08    Ready    <none>   288d   v1.10.2
kube-node09    Ready    <none>   288d   v1.10.2
kube-node10    Ready    <none>   288d   v1.10.2
kube-node11    Ready    <none>   288d   v1.10.2
kube-node12    Ready    <none>   288d   v1.10.2
kube-node13    Ready    <none>   273d   v1.10.2
kube-node14    Ready    <none>   72d    v1.10.2
```

K8S in action: small demo

K8S for CNAF storage services

GPFS and containers

https://developer.ibm.com/storage/2018/12/20/spectrum-scale-and-containers/

What is Storage Enabler for Containers (SEC)?

Simply put, Storage Enabler for Containers (SEC) is an interface that is delivered via IBM Spectrum Connect that enables IBM Storage products (e.g. Spectrum Scale, Spectrum Virtualize, Spectrum Accelerate, and IBM FlashSystem) to connect to Kubernetes clusters. Once the interface is configured, SEC allows containerized applications to make use of IBM Storage to dynamically provision Persistent Volumes (PVs) through Persistent Volume Claims (PVCs).

K8S for data access and management services

Data access & management services (StoRM, StoRM WebDAV, XRootD, GridFTP, …) are deployed on K8S on T1 resources

GPFS (but Ceph could also be used where appropriate) provides persistent storage

Pros:

- Immutable infrastructure, Self-healing, autoscaling, continuous delivery, …

Cons:

- It’s a sea change