Containerizing .NET Core Applications with Microsoft Azure and AKS

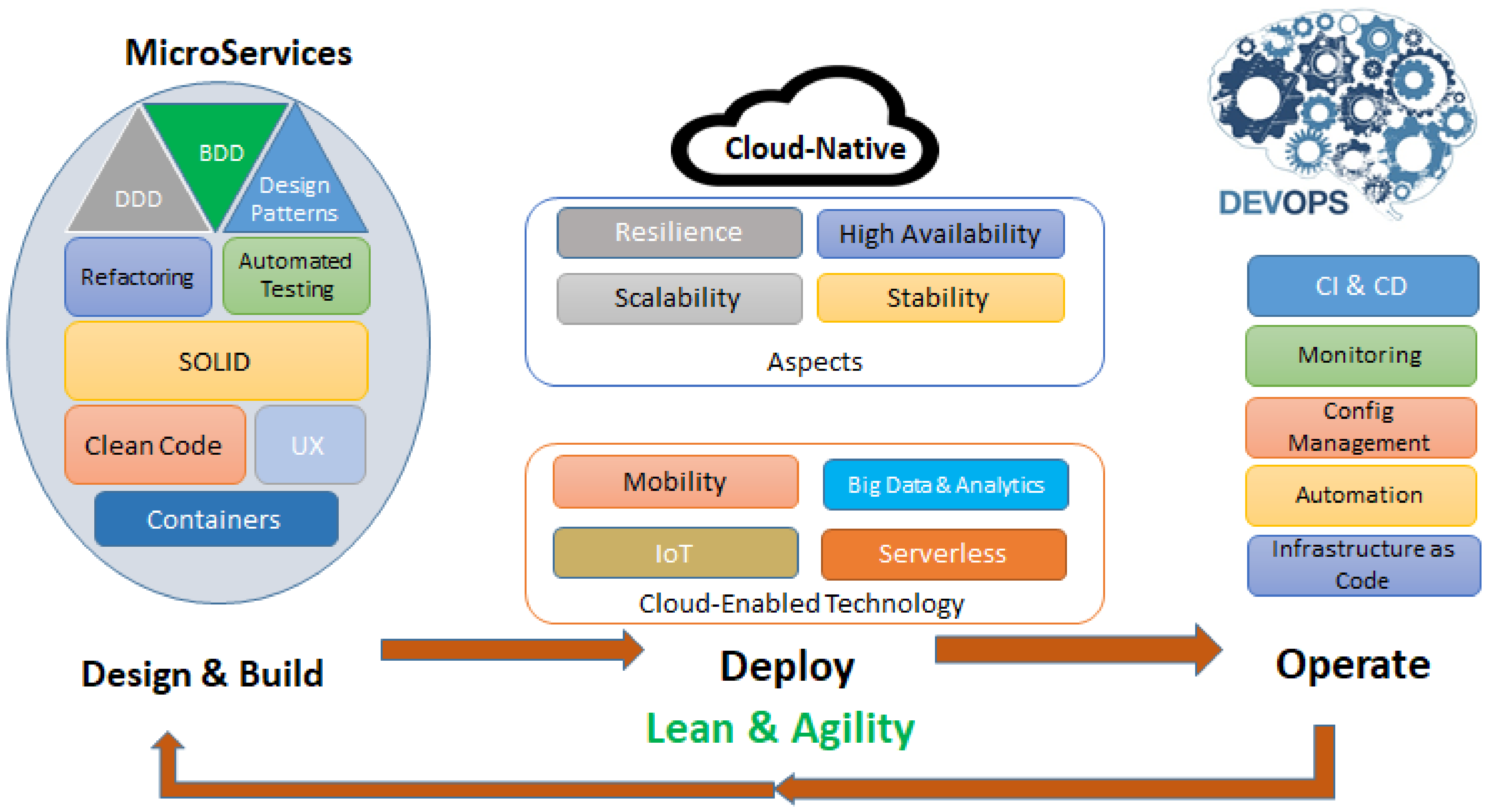
Manoj Ganapathi

Chief Architect, CodeOps

About Me

- Manoj is a seasoned IT professional with more than 20 years of experience. He has extensive experience in enterprise & solution architecture, design and implementation of large & complex enterprise systems. As an architect and technology consultant, he has consulted with several large, fortune 500 enterprises and worked with ISVs and startups. In his career, he has worked in multiple technology-oriented and leadership roles across all phases of software development life cycle. He is experienced in building and running technical communities and has been a speaker in several technology conferences.
- Over the last seven years, he has worked extensively on consulting, architecture and implementation of Cloud-based solutions, specializing on building highly scalable, resilient systems and DevOps practices.
- Currently, he is the Chief Architect at CodeOps Technologies (http://codeops.tech/) and a Digital Technology Consultant.
- LinkedIn profile: https://www.linkedin.com/in/manojg
- @manojgr, manoj@codeops.tech

Perspective on Modern Software Delivery



Containers v/s VMs

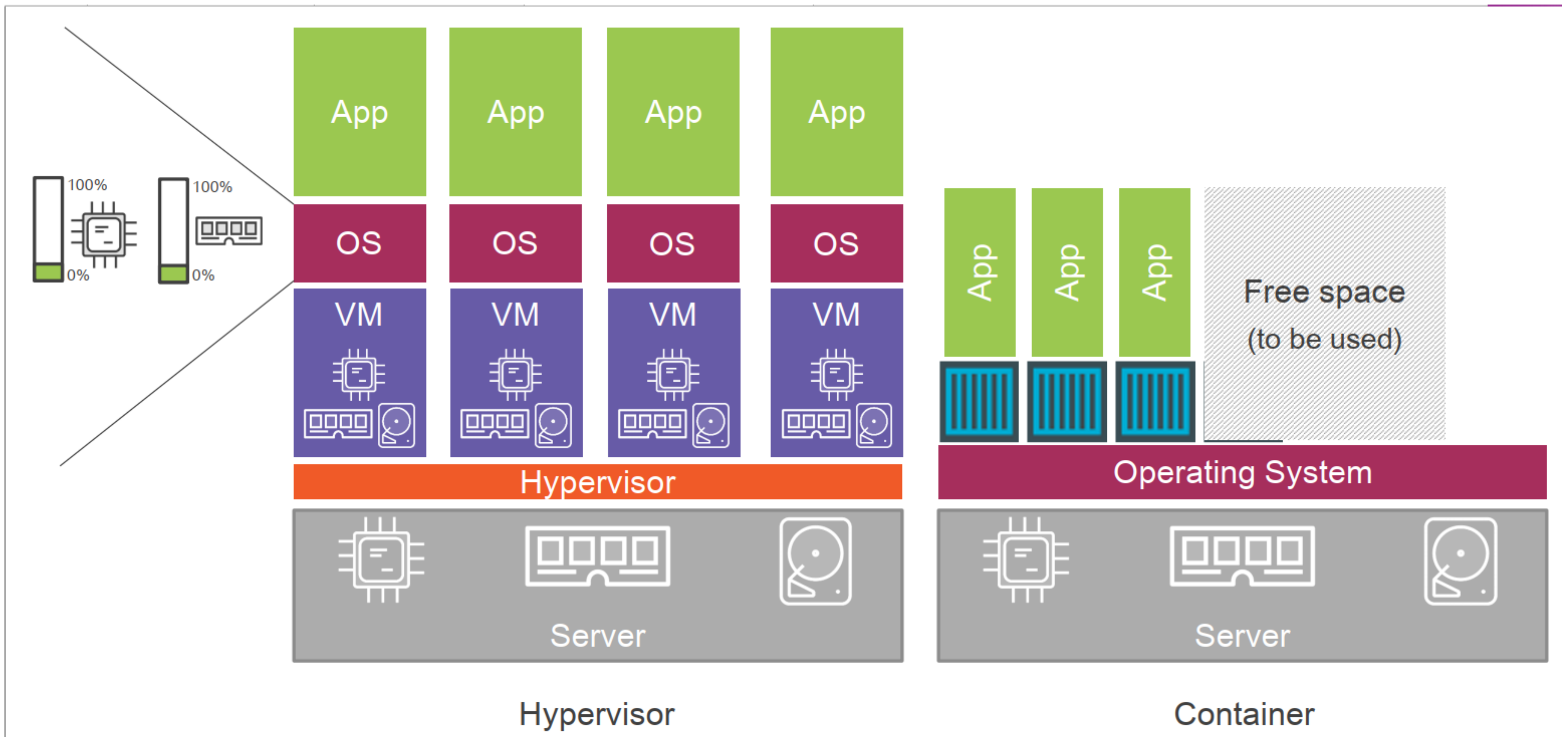
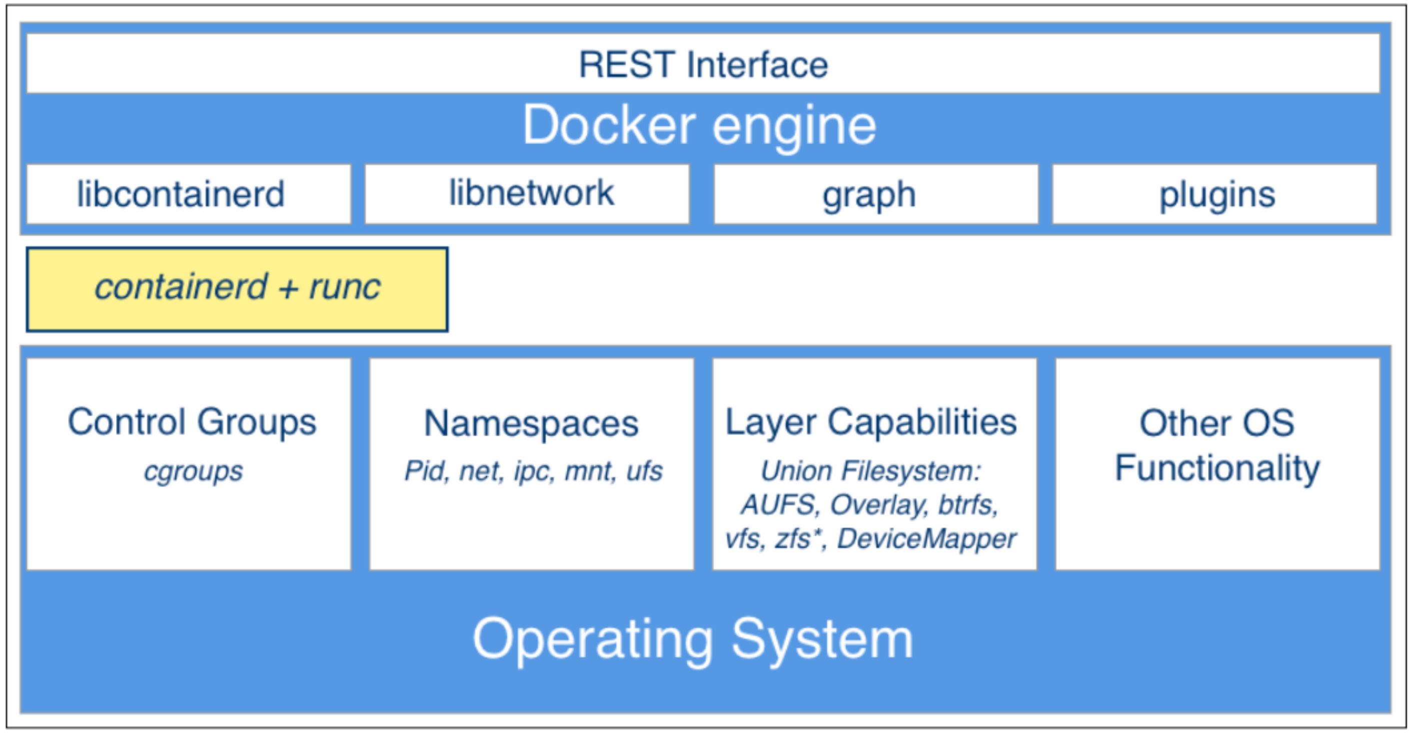


Image Ref: Nigel Poulton

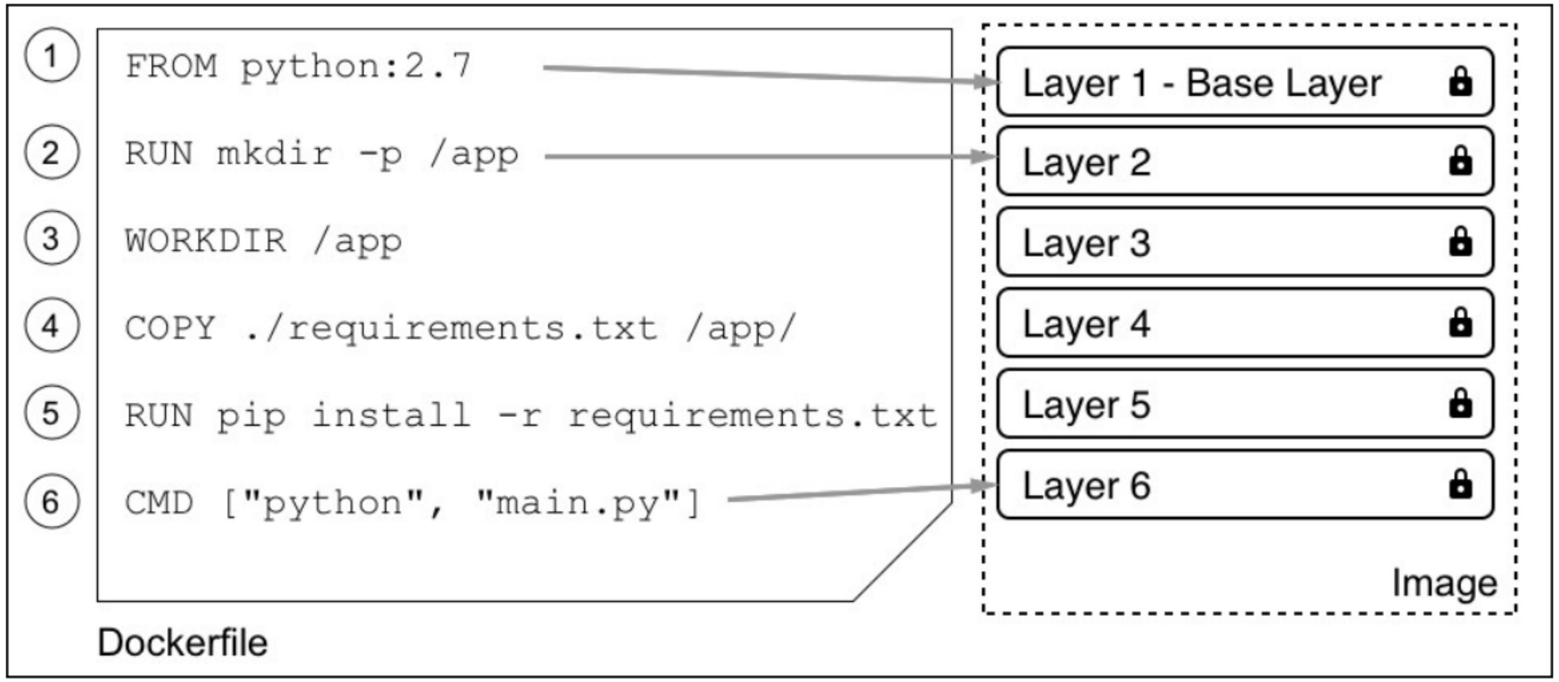
Docker Architecture



- Containers are encapsulated, secure processes running on the host platform
- Benefits
  - Security
  - Isolation
  - Standardized Infra

Ref: Containerize your apps with Docker & Kubernetes - book

Container Images & Dockerfile



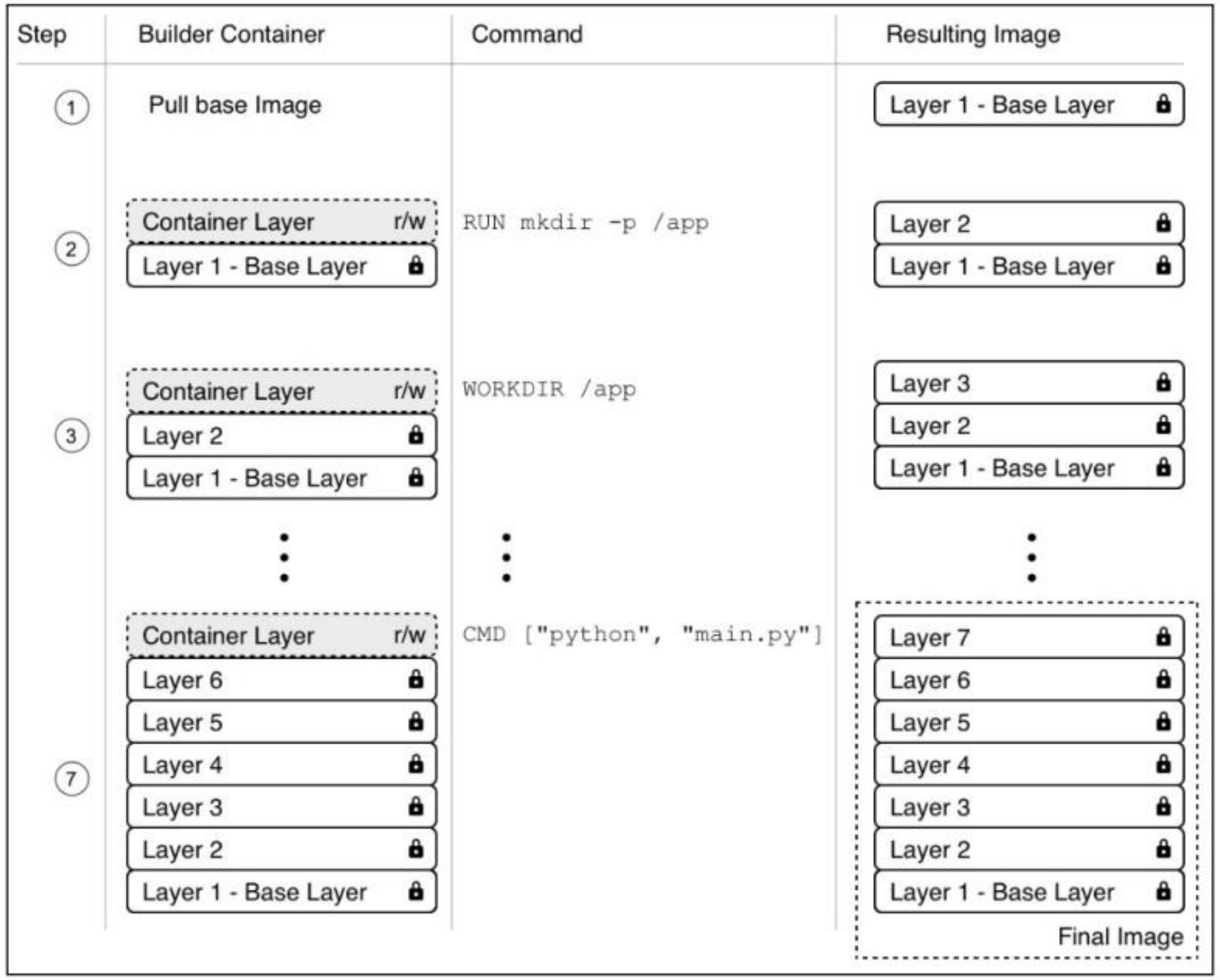
```
FROM python:2.7
RUN mkdir -p /app
WORKDIR /app
COPY ./requirements.txt /app/
RUN pip install -r requirements.txt
CMD ["python", "main.py"]
```

The relation of Dockerfile and layers in an image

- Templates for Containers
- Starts with a base layer, usually the OS
- All layers immutable, only top layer is writable

Ref: Containerize your apps with Docker & Kubernetes - book

Best Practices



The image build process visualized

- Keep Containers ephemeral
- Order commands to leverage caching
- Avoid installing multiple packages
- Use .dockerignore
- Use multi-stage builds

Ref: Containerize your apps with Docker & Kubernetes - book

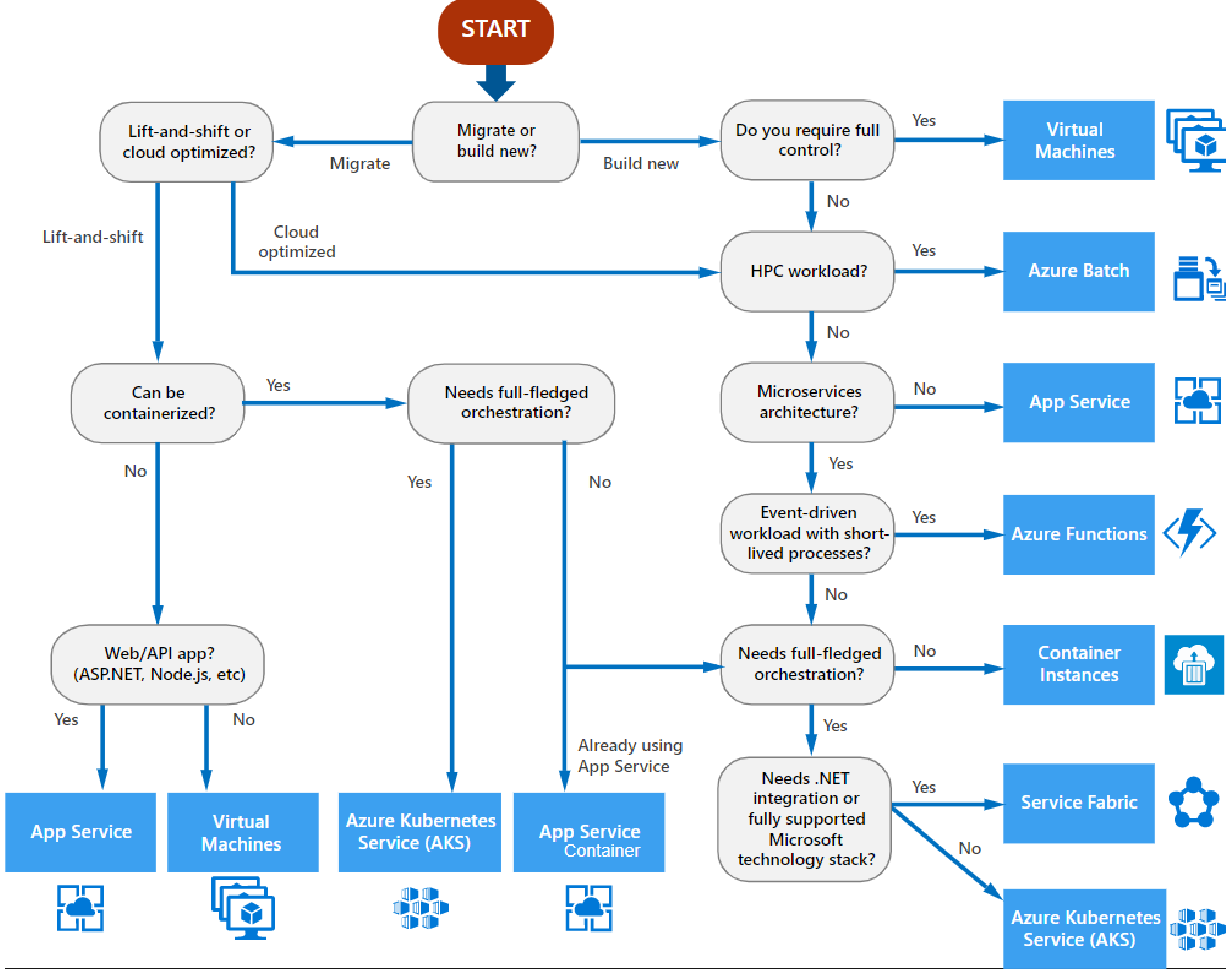
Advantages of Containers

- Very lightweight
- Scales quickly
- Accelerates developer onboarding
- Easy to package apps, dependencies and version
- Eliminate environment inconsistencies
- Easier distribution
- Support CI/CD DevOps more seamlessly
- Better oriented towards Cloud-Native and distributed architecture patterns

Container Registry

- Single place for your team to manage Docker images
- Public and Private registries
  - Azure Container Registry - Private
  - Docker Hub - Public
- Provide features like:
  - Vulnerability analysis
  - Geo-replication
  - Role-based access control.
- Docker Registry API supported by all registry providers (like Azure)
- Important element of CI/CD pipelines

Choosing the right compute option



START
Migrate or build new?
Migrate
Build new
Lift-and-shift or cloud optimized?
Lift-and-shift
Cloud optimized
Do you require full control?
Yes
Virtual Machines
No
HPC workload?
Yes
Azure Batch
No
Microservices architecture?
No
App Service
Yes
Event-driven workload with short-lived processes?
Yes
Azure Functions
No
Needs full-fledged orchestration?
No
Container Instances
Yes
Needs .NET integration or fully supported Microsoft technology stack?
Yes
Service Fabric
No
Azure Kubernetes Service (AKS)
Can be containerized?
Yes
Needs full-fledged orchestration?
Yes
Azure Kubernetes Service (AKS)
No
Already using App Service
App Service Container
No
Web/API app? (ASP.NET, Node.js, etc)
Yes
App Service
No
Virtual Machines

https://docs.microsoft.com/en-us/azure/architecture/guide/technology-choices/compute-decision-tree

Azure Container Instances

- Run containers without managing servers or having to learn new tools
- With the Virtual Kubelet, use ACI to elastically burst from your Azure Kubernetes Service (AKS) cluster when traffic comes in spikes.
- Gain the security of virtual machines for your container workloads, while preserving the efficiency of lightweight containers
- Combine ACI with the ACI Logic Apps connector, Azure queues and Azure Functions to build robust infrastructure which can elastically scale out containers on demand
- Use Azure Container Instances for data processing where source data is ingested, processed and placed in a durable store such as Azure Blob storage

https://azure.microsoft.com/en-in/services/container-instances/

Need for Container Orchestrators

VELOCITY

SCALING (SOFTWARE & TEAMS)

ABSTRACTING INFRASTRUCTURE

EFFICIENCY

Achieving Velocity

IMMUTABILITY

DECLARATIVE CONFIGURATION

SELF-HEALING

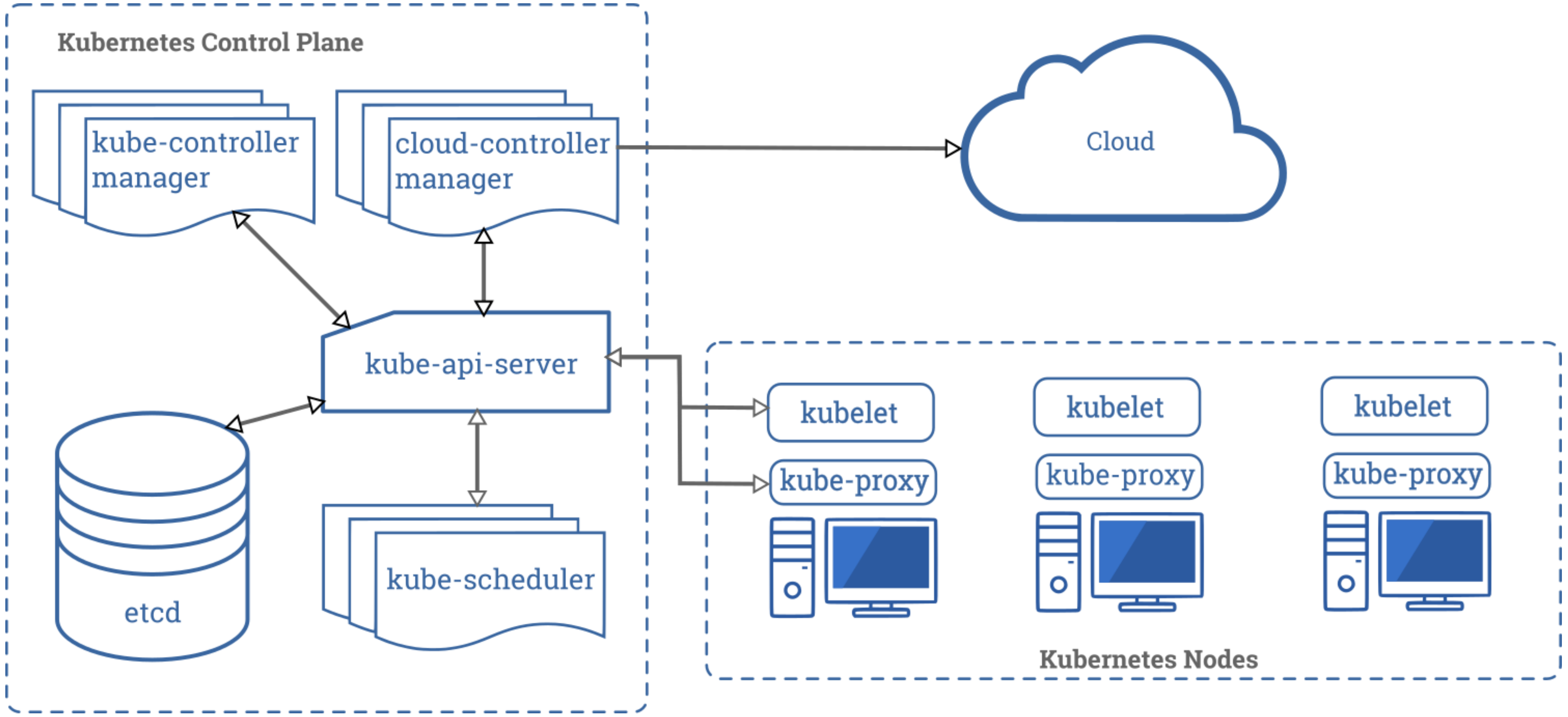
Other benefits

DECOUPLED ARCHITECTURE

HIGH UTILIZATION

EFFICIENCY

K8S Architecture



https://kubernetes.io/docs/concepts/overview/components/

K8S objects

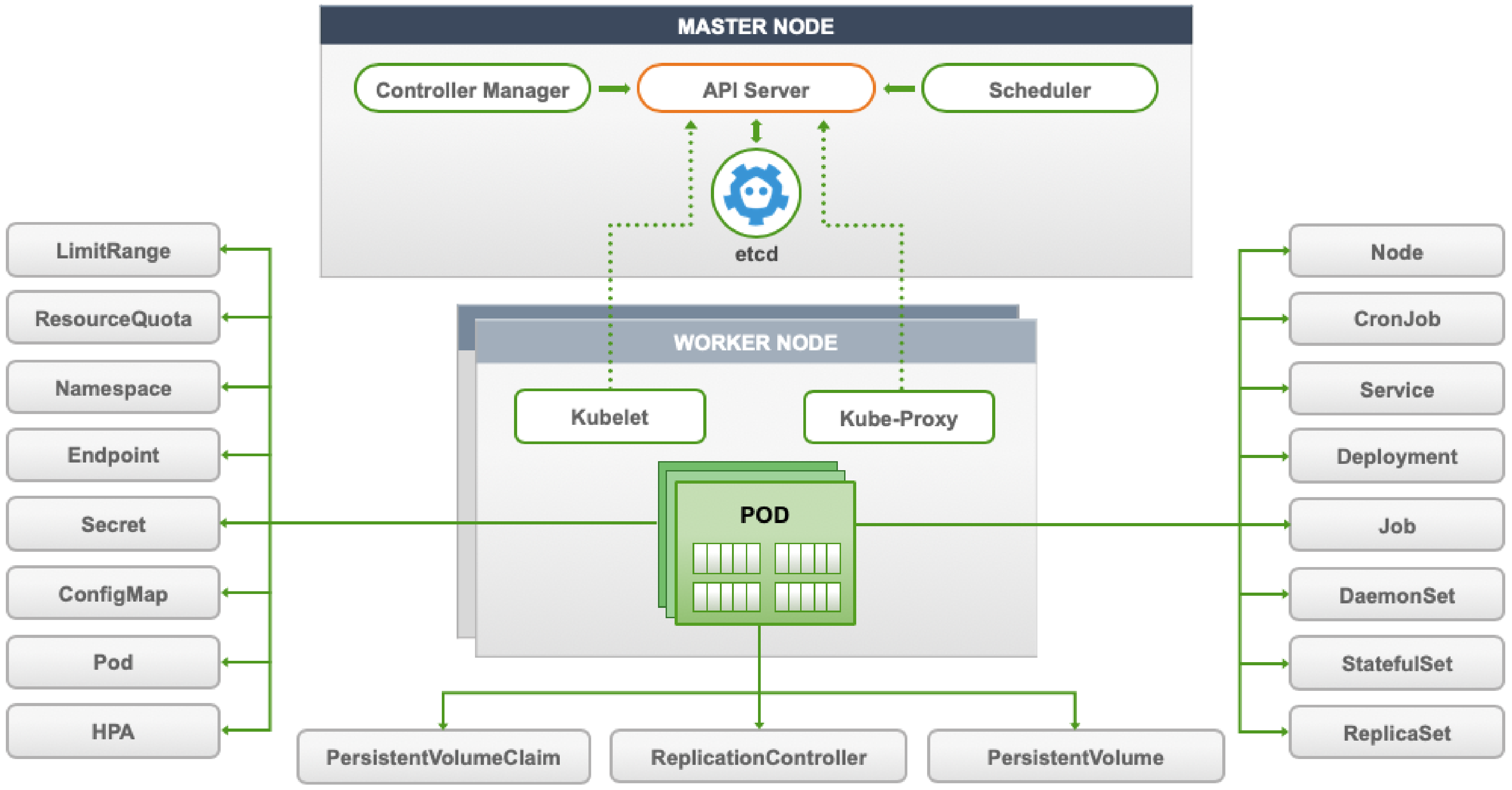
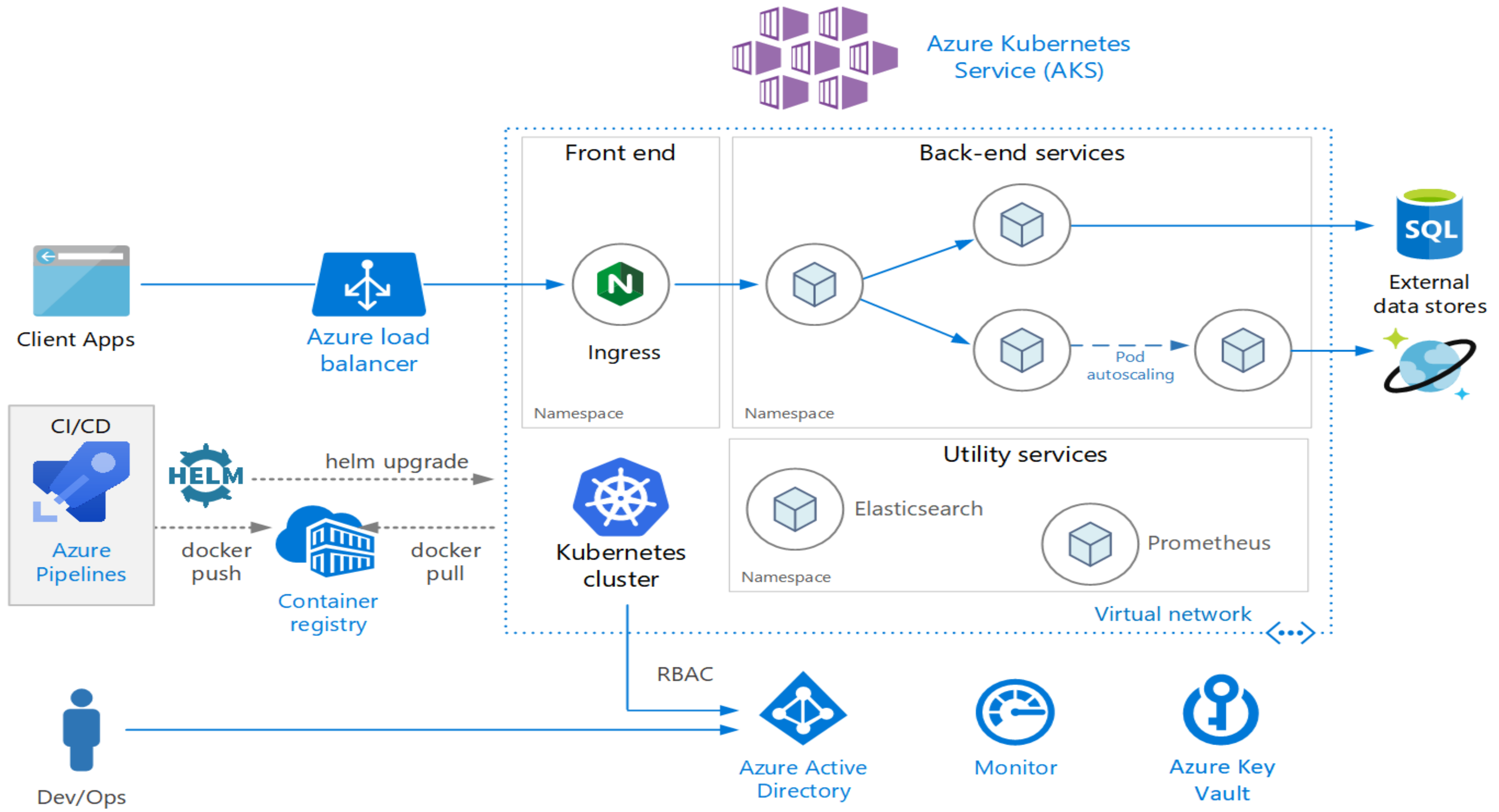


Image Ref: https://www.splunk.com/en_us/blog/it/monitoring-kubernetes.html

Microservices architecture on Azure Kubernetes Service (AKS)



https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/containers/aks-microservices/aks-microservices

Why Azure Kubernetes Service (AKS)?

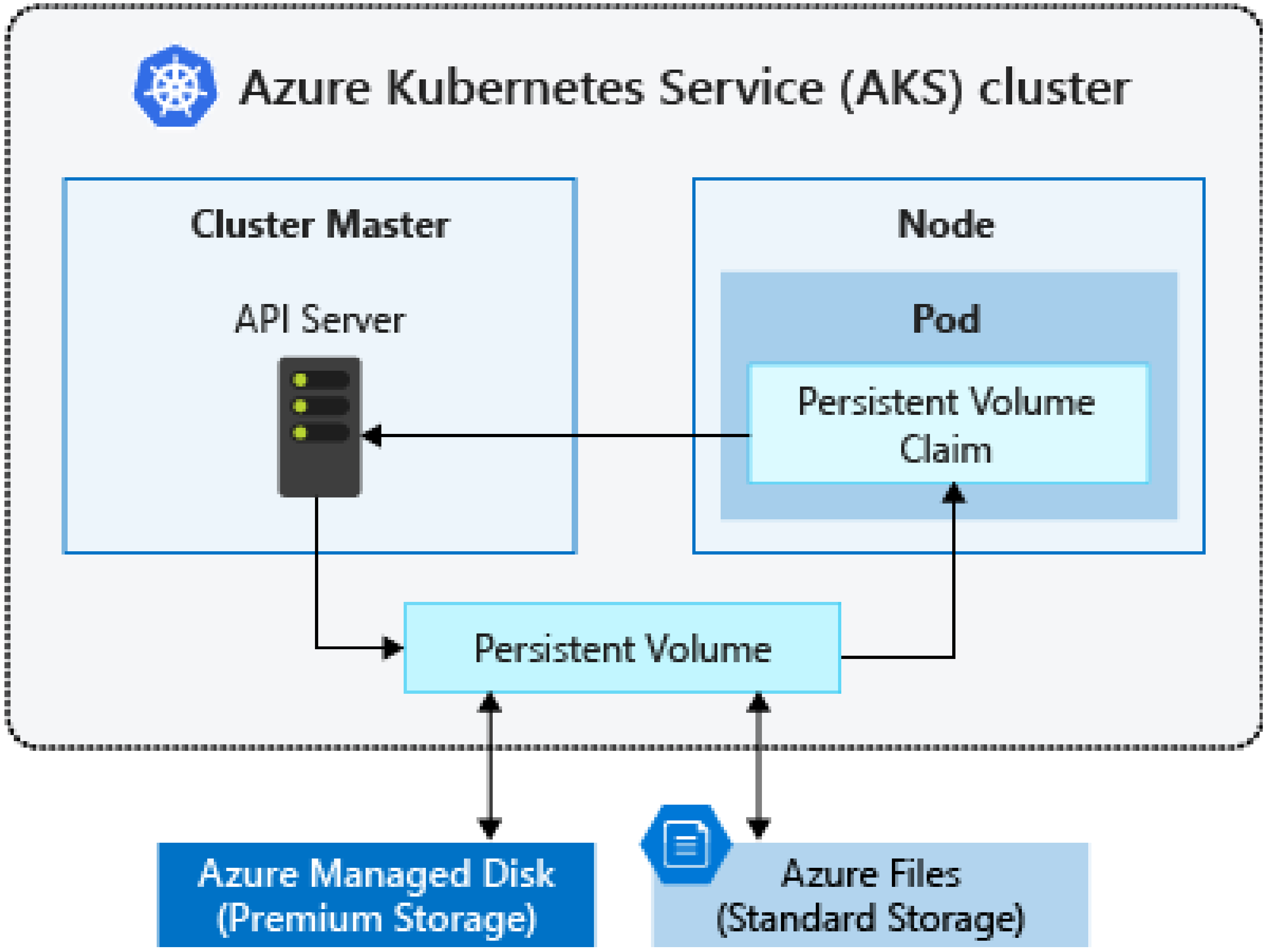
Highly available, secure and fully managed Kubernetes service

Fully managed control plane – you focus on your applications/workloads

Elastic provisioning of capacity without the need to manage the infrastructure

Faster end-to-end development experience with seamless integration with **Visual Studio**/VS Code, Dev Spaces, **Azure DevOps and Azure Monitor**

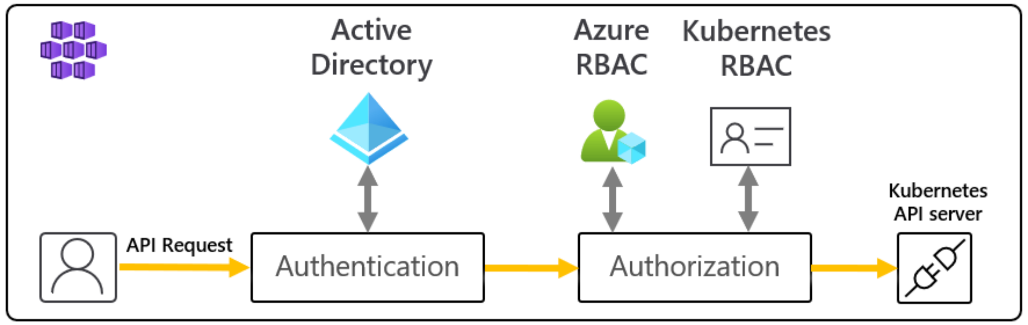
Advanced identity and access management using Azure Active Directory and dynamic rules enforcement across multiple clusters with Azure Policy

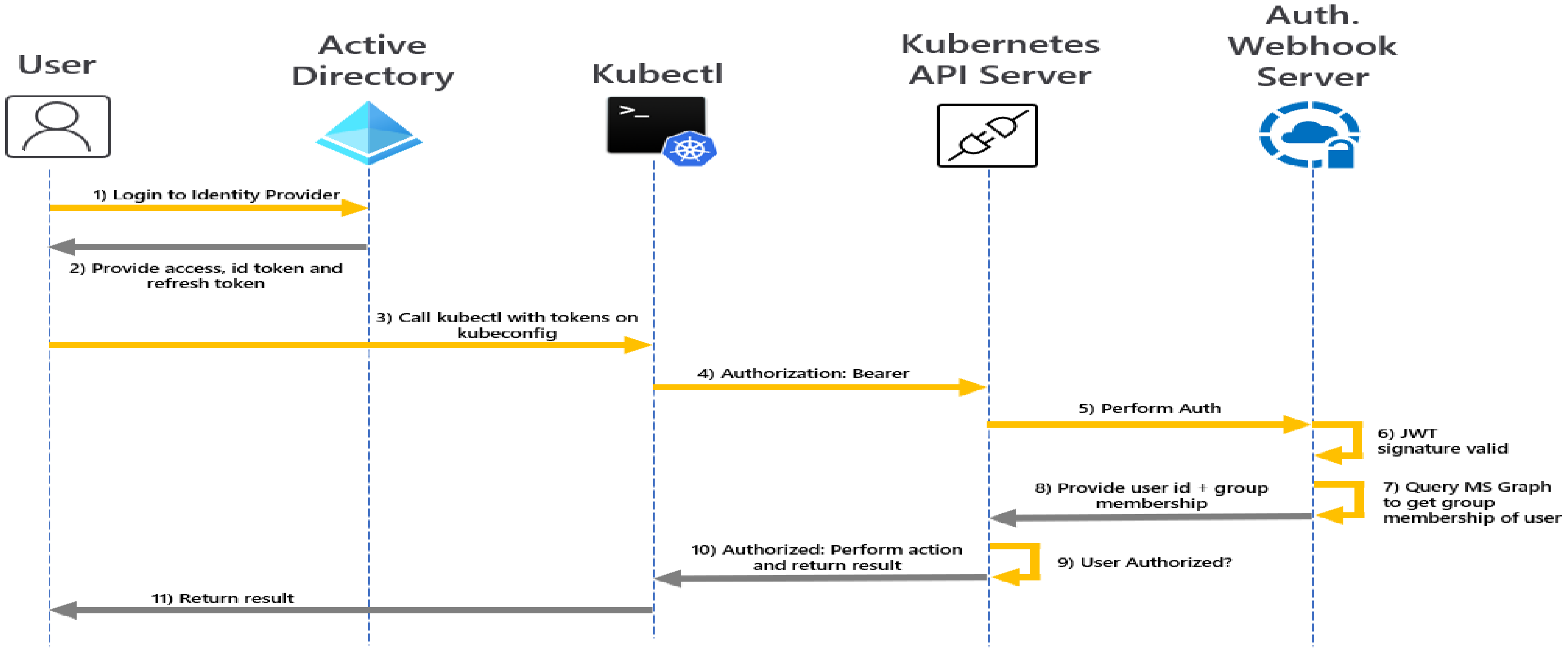

Azure Kubernetes Service (AKS) cluster
Cluster Master
API Server
Node
Pod
Persistent Volume Claim
Persistent Volume
Azure Managed Disk (Premium Storage)
Azure Files (Standard Storage)

Storage in K8S/AKS

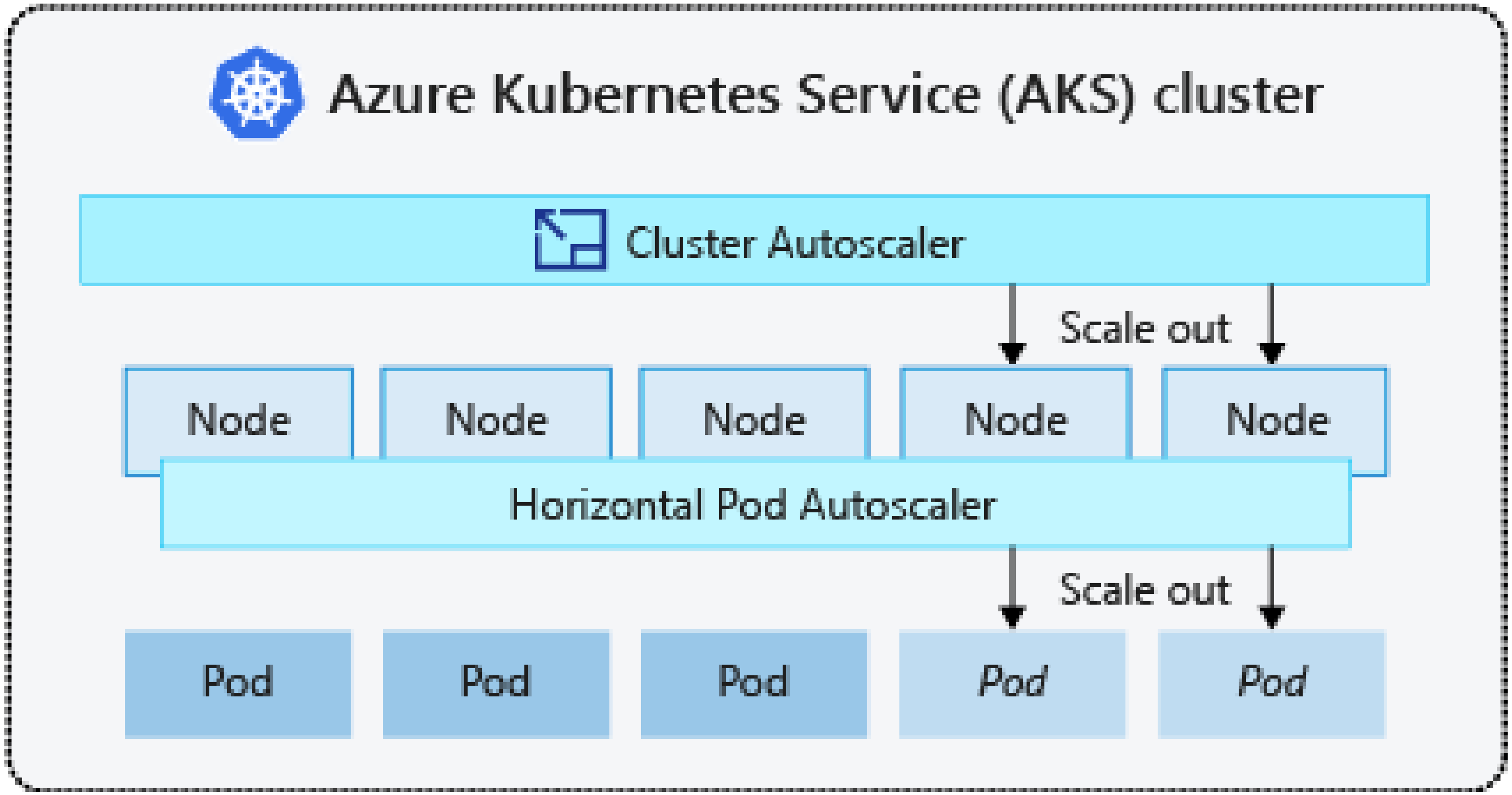
https://docs.microsoft.com/en-us/azure/aks/concepts-storage

Authentication & Authorization




User
Active Directory
Kubectl
Kubernetes API Server
Auth. Webhook Server
1) Login to Identity Provider
2) Provide access, id token and refresh token
3) Call kubectl with tokens on kubeconfig
4) Authorization: Bearer
5) Perform Auth
6) JWT signature valid
7) Query MS Graph to get group membership of user
8) Provide user id + group membership
9) User Authorized?
10) Authorized: Perform action and return result
11) Return result

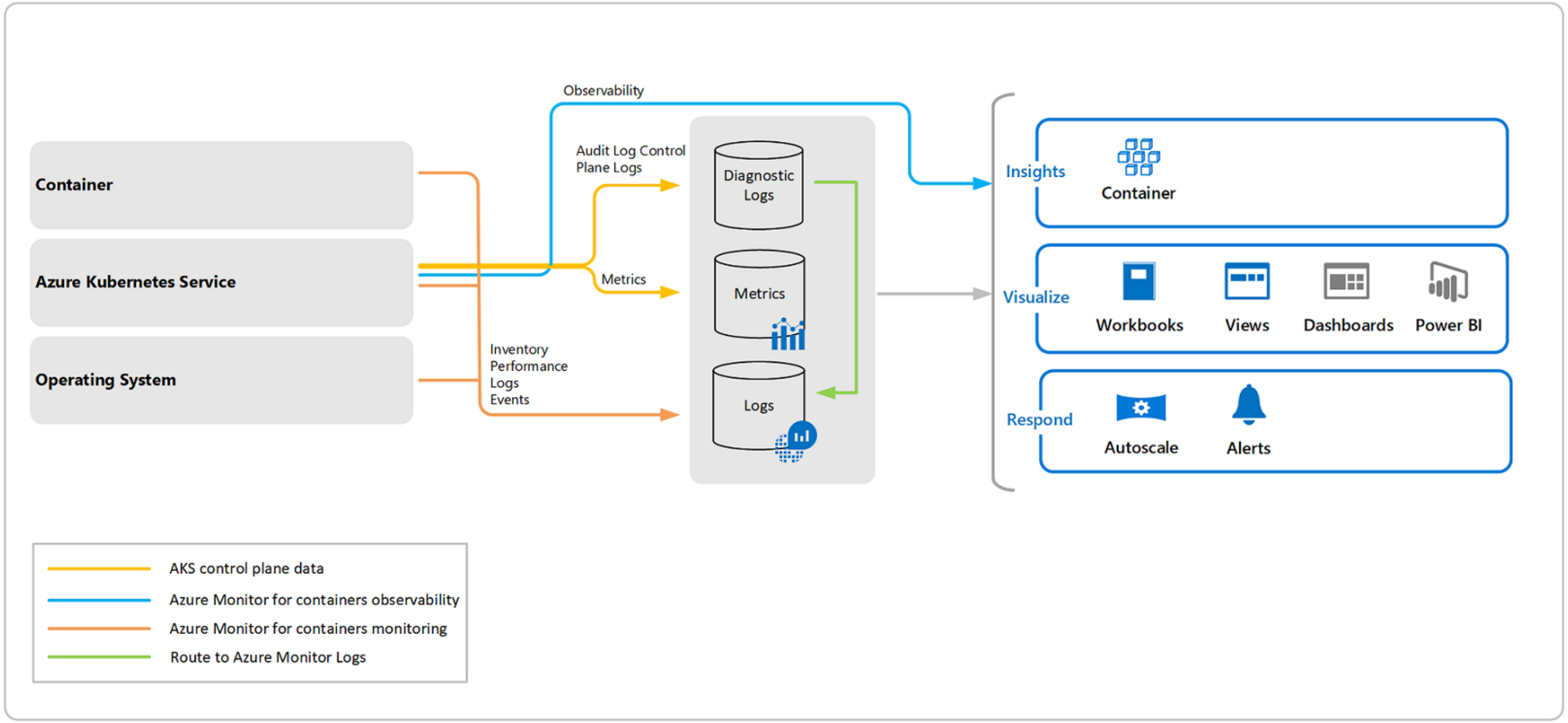
https://docs.microsoft.com/en-us/azure/aks/concepts-identity



Scaling options for applications in Azure Kubernetes Service (AKS)

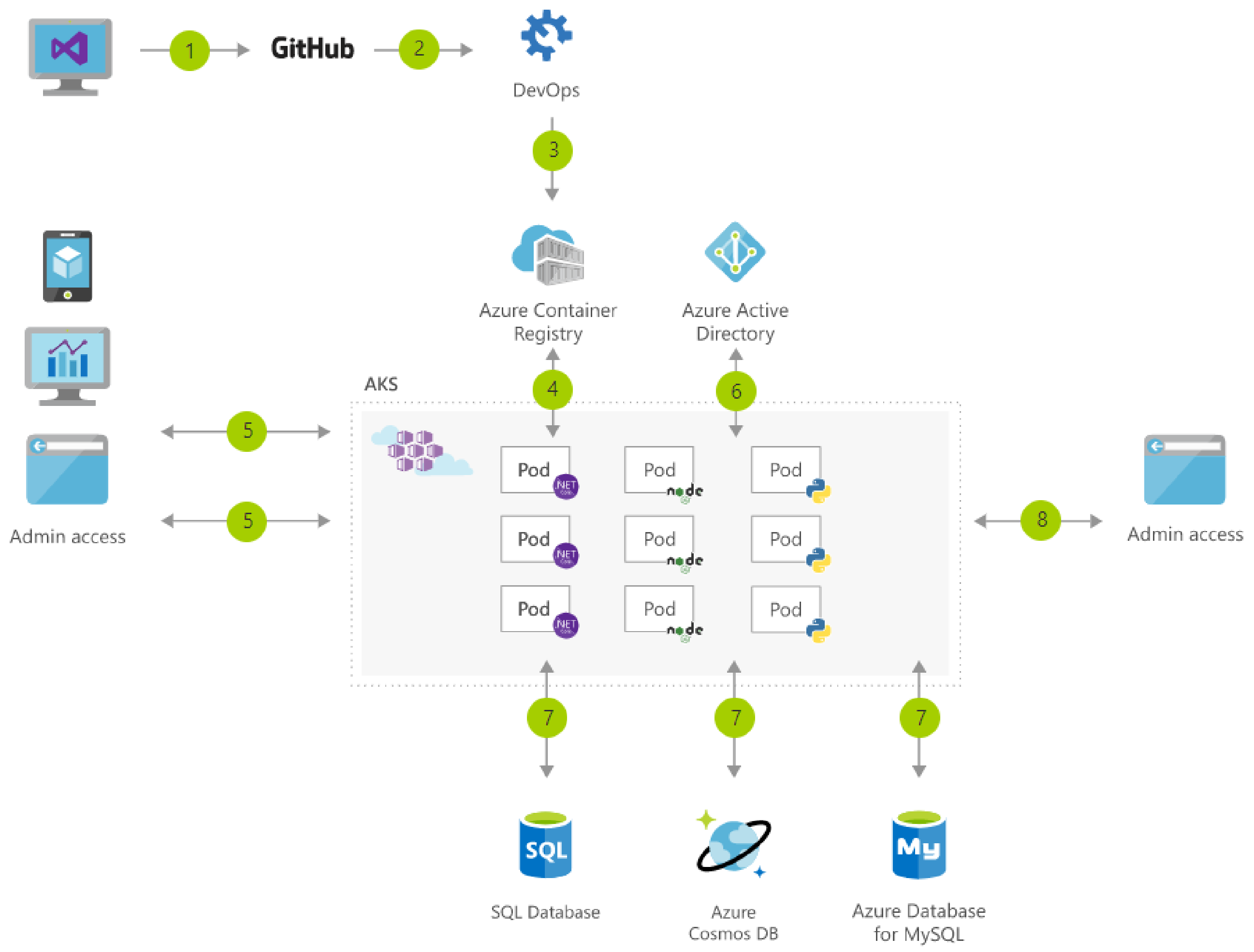
https://docs.microsoft.com/en-us/azure/aks/concepts-scale

Monitoring in AKS (Azure Monitor)



https://docs.microsoft.com/en-us/azure/azure-monitor/insights/container-insights-overview

Deploying with AKS (DevOps)



1. Use an IDE, such as Visual Studio, to commit changes to GitHub.
2. GitHub triggers a new build on Azure DevOps
3. Azure DevOps packages microservices as containers and pushes them to the Azure Container Registry
4. Containers are deployed to AKS cluster

5. Azure Active Directory is used to secure access to the resources
6. Users access services via apps and websites
7. Administrators access the apps via a separate admin portal
8. Microservices use databases to store and retrieve information

https://azure.microsoft.com/en-in/solutions/architecture/microservices-with-aks/

Resources

- K8s Learning Path
- Docker on Azure -
- Containerize your apps with Docker & Kubernetes
- Learning path for Containers & Kubernetes on MS Learn
- Git Hub Repo (Labs)
- K8S Cheat Sheet
- AKS Best Practices
- Kubernetes Learning Path
- Kubernetes Up and Running Book