Speaker

meetup.com/Azure-Cloud-Security-Meetup-Group/

David Okeyode (MVP)

Cloud Security Consultant

Speaker Bio: Microsoft Azure MVP. Over a decade of experience in Cybersecurity (consultancy, design, implementation). Over 6 years of experience as a trainer. Developed multiple vulnerable by design automation templates that can be used to practice cloud penetration testing techniques. Authored two cloud computing courses for the popular cybersecurity training platform – Cybrary.

@asegunlolu

http://www.youtube.com/c/DavidOkeyode

http://azurehangout.com

Microsoft CERTIFIED AZURE AI ENGINEER ASSOCIATE

Microsoft CERTIFIED AZURE DEVELOPER ASSOCIATE

Microsoft CERTIFIED AZURE ADMINISTRATOR ASSOCIATE

Microsoft CERTIFIED AZURE SECURITY ENGINEER ASSOCIATE

Microsoft CERTIFIED AZURE SOLUTIONS ARCHITECT EXPERT

Microsoft CERTIFIED AZURE DEVOPS ENGINEER EXPERT

aws certified Solutions Architect Professional

aws certified Security Specialty

Agenda

- Different approach to cloud security education
- Containerization primer
- Azure containerization options
- Containerization security models
- Kubernetes attack matrix
- Demo

Why a different approach?

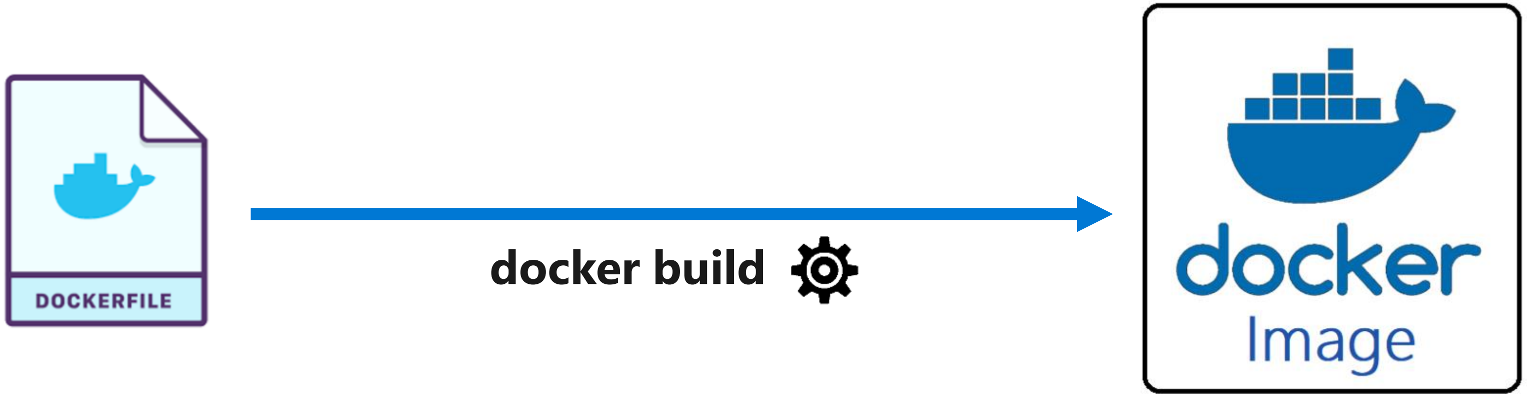
- A lot of presentations, talks and videos already focus on best practices and the use of tools
- Understanding attacker behaviour is an important part of cybersecurity education
- Learn what not to do from the failures of others

Containerization primer

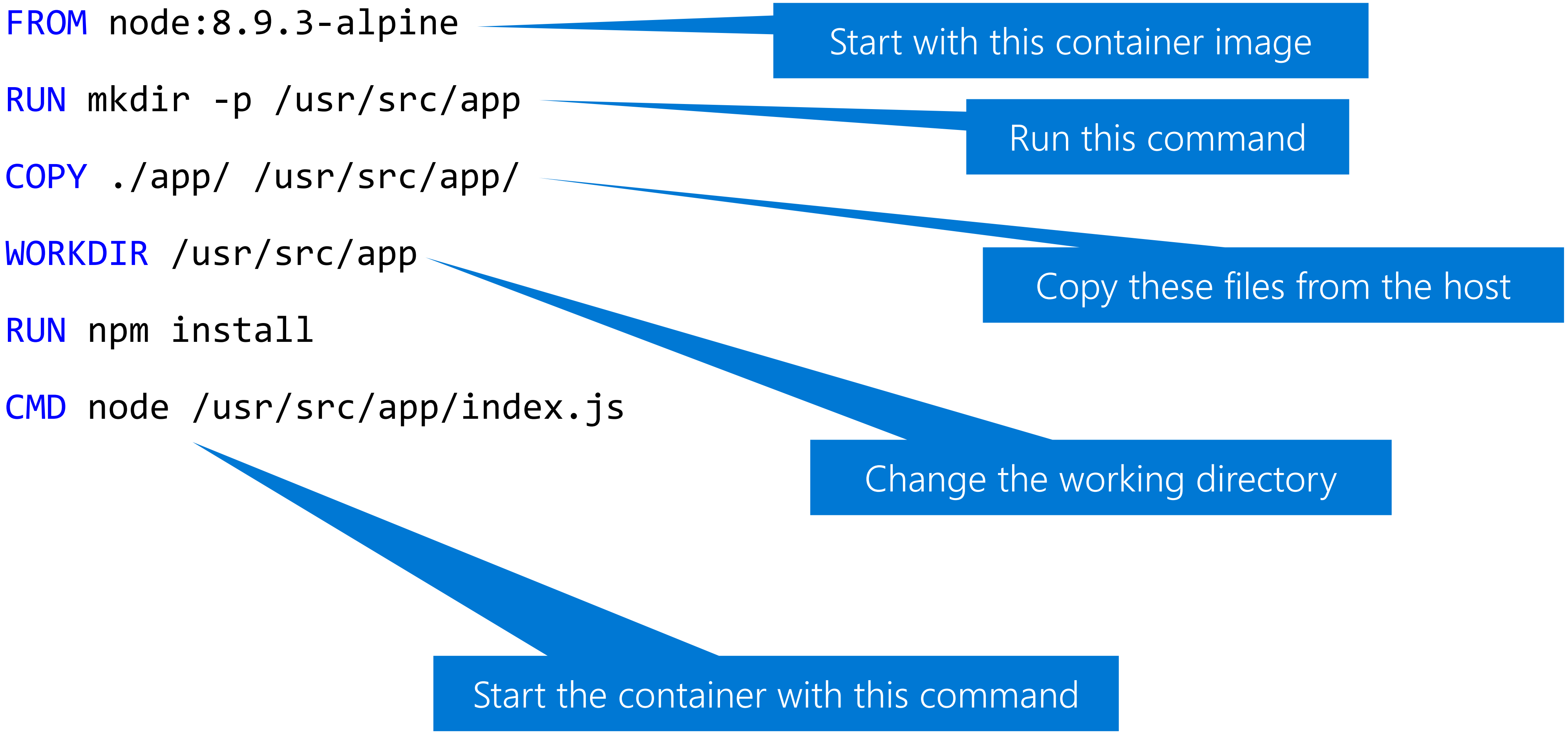
- Development
  - Build container image
  - Publish containerized image to an image registry
- Runtime
  - Run image as a container

Creating a container image

- To create a custom container image, we use a dockerfile
- A dockerfile is a simple text file with step-by-step instructions used to build a container image
- The docker build command builds Docker images from a dockerfile



Creating a container image specification with a Dockerfile



```
FROM node:8.9.3-alpine

RUN mkdir -p /usr/src/app

COPY ./app/ /usr/src/app/

WORKDIR /usr/src/app

RUN npm install

CMD node /usr/src/app/index.js
```

Start with this container image

Run this command

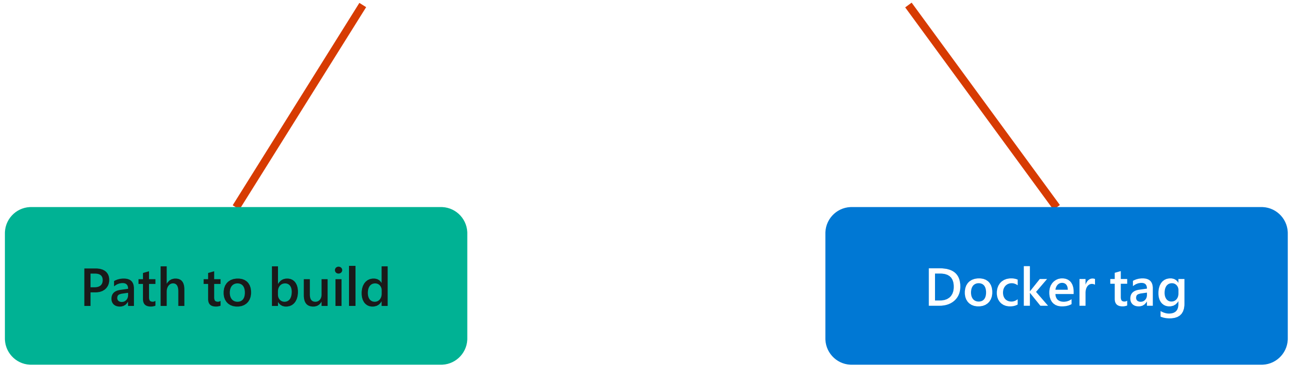
Copy these files from the host

Change the working directory

Start the container with this command

Building the container image

```
# Build your container
docker build ./application -t test-app
```



Path to build

Docker tag

```
# After building, use the following command to view your new container image
docker images
```

Azure Containerization Options

Build Container Images

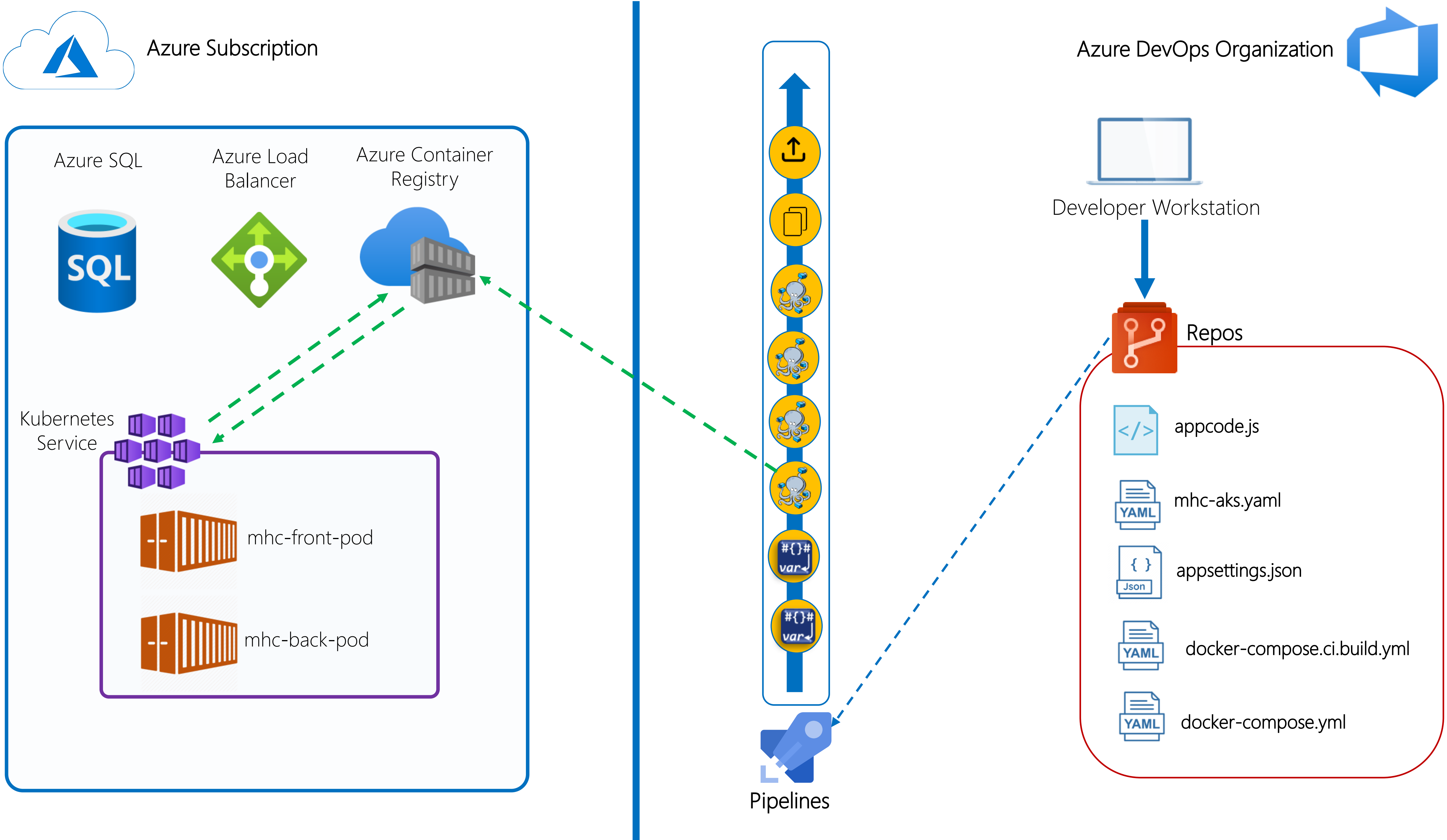
Docker VM

Azure Pipelines

Azure Container Registry Tasks

Store and Distribute Images

Runtime Options



Azure Subscription

Azure SQL

Azure Load Balancer

Azure Container Registry

Kubernetes Service

mhc-front-pod

mhc-back-pod

Pipelines

Azure DevOps Organization

Developer Workstation

Repos

appcode.js

mhc-aks.yaml

appsettings.json

docker-compose.ci.build.yml

docker-compose.yml

Azure Containerization Options

Build Container Images

Docker VM

Azure Pipelines

Azure Container Registry Tasks

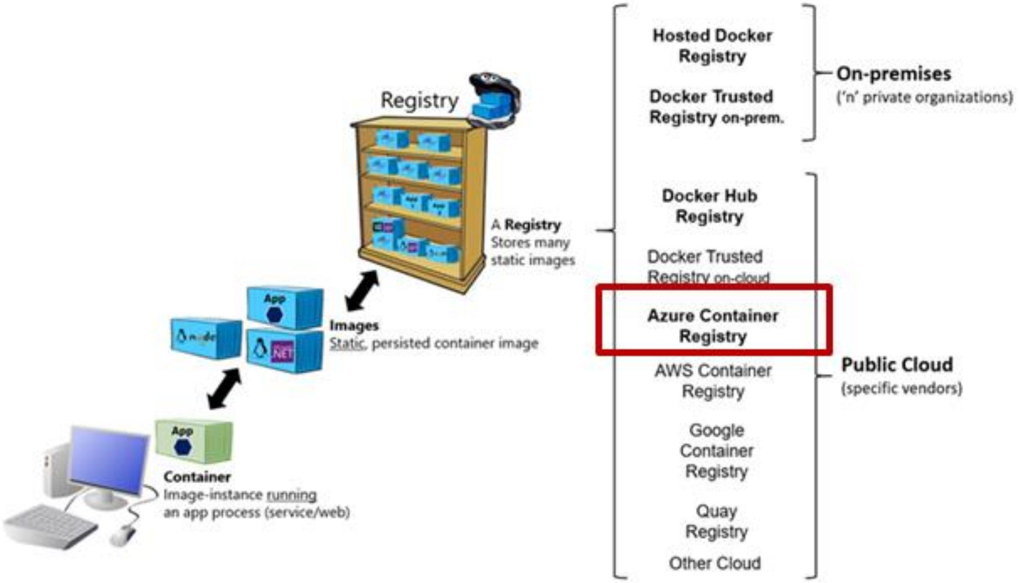
Store and Distribute Images

Azure Container Registry

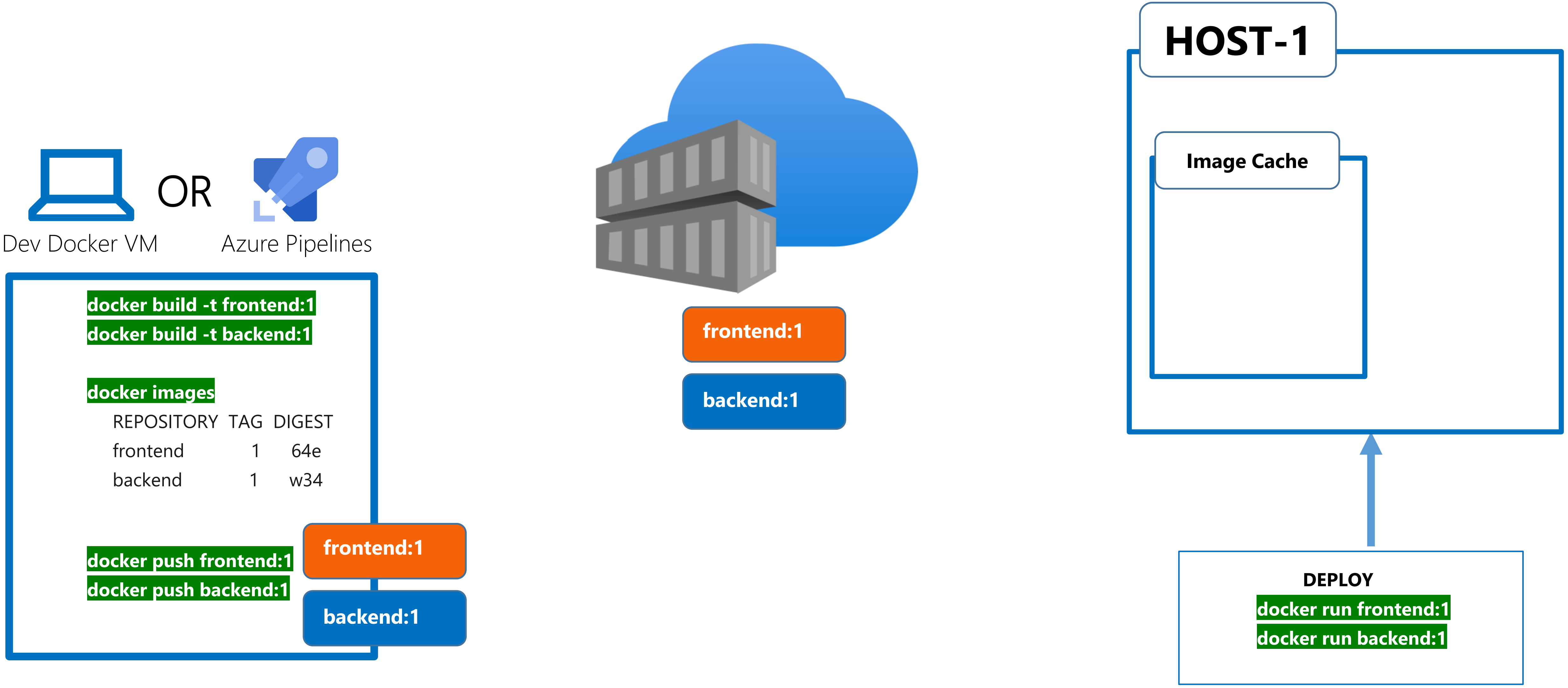
Runtime Options

Azure Container Registry

- A managed docker registry service to STORE and DISTRIBUTE container images and other artifacts
- It is based on the open-source Docker Registry 2.0
- ACR helps us to mitigate the risk of malicious container images
- It has tight integration with multiple Azure services that support these Docker containers


Registry
A Registry
Stores many static images
App
node
App
.NET
Images
Static, persisted container image
App
Container
Image-instance running an app process (service/web)
Hosted Docker Registry
Docker Trusted Registry on-prem.
On-premises
('n' private organizations)
Docker Hub Registry
Docker Trusted Registry on-cloud
Azure Container Registry
AWS Container Registry
Google Container Registry
Quay Registry
Other Cloud
Public Cloud
(specific vendors)

How ACR Works



```
docker build -t frontend:1
docker build -t backend:1

docker images
    REPOSITORY  TAG  DIGEST
    frontend      1    64e
    backend       1    w34

docker push frontend:1
docker push backend:1
```

DEPLOY

```
docker run frontend:1
docker run backend:1
```

Azure Container Registry - Tiers

Basic

- Supports similar functionalities as Standard (webhook integration, registry authentication with Azure Active Directory)
- Has less scale than standard (10 GiB included storage; 1000 ReadOps per minute; 100 WriteOps per minute; 30Mbps download bandwidth ; 10Mbps upload bandwidth)

Standard

- Has more scale than basic (100 GiB included storage; 3000 ReadOps per minute; 500 WriteOps per minute; 60Mbps download bandwidth ; 20Mbps upload bandwidth)

Premium

- Supports security features like geo-replication, content trust, private endpoint, and encryption using customer managed keys
- Has more scale than standard (500 GiB included storage; 10000 ReadOps per minute; 2000 WriteOps per minute; 100Mbps download bandwidth ; 50Mbps upload bandwidth)

Azure Containerization Options

Build Container Images

Docker VM

Azure Pipelines

Azure Container Registry Tasks

Store and Distribute Images

Azure Container Registry

Runtime Options

Without Orchestration

Virtual Machine

Container Instances

Batch

App Service

Function

With Orchestration

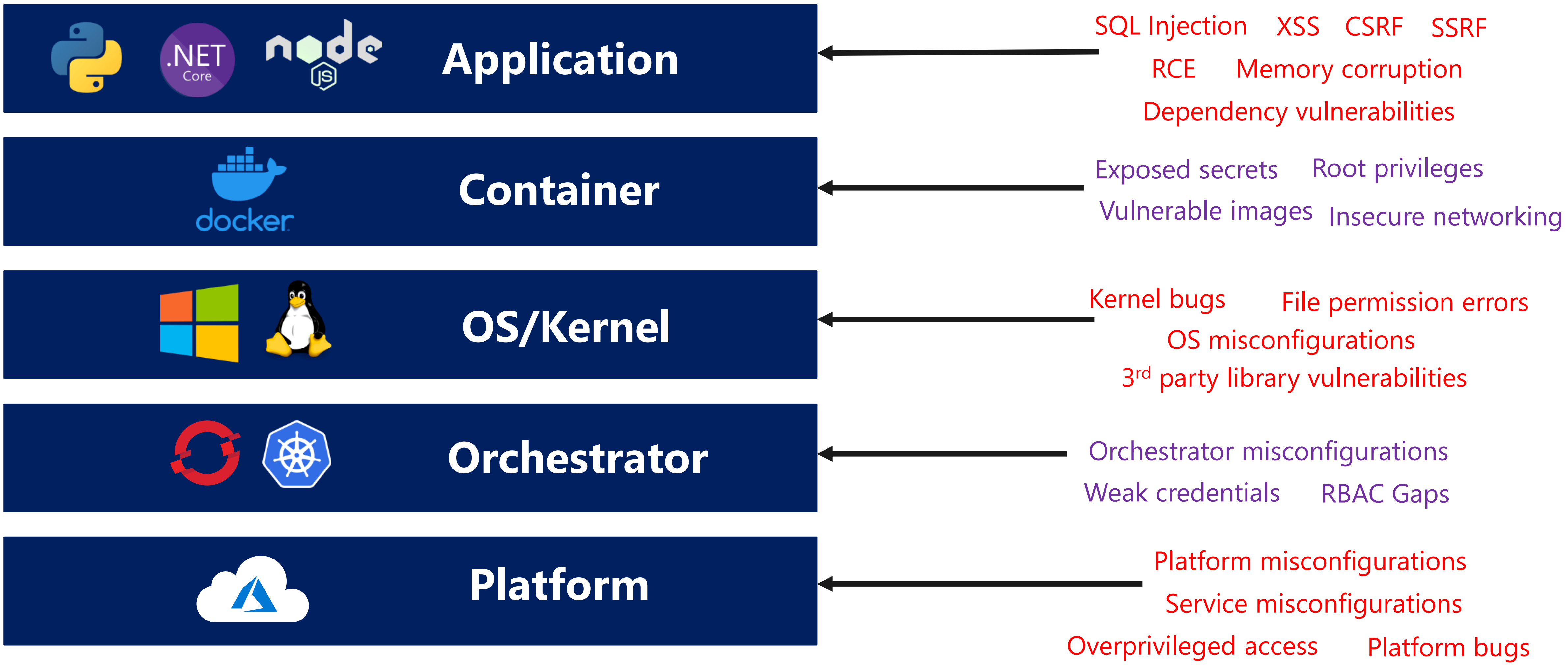
Kubernetes Service

Service Fabric

Azure RedHat OpenShift (ARO)

Modern Application Stack

- The modern application stack is distributed and complex



Application

SQL Injection XSS CSRF SSRF
RCE Memory corruption
Dependency vulnerabilities

Container

Exposed secrets Root privileges
Vulnerable images Insecure networking

OS/Kernel

Kernel bugs File permission errors
OS misconfigurations
3rd party library vulnerabilities

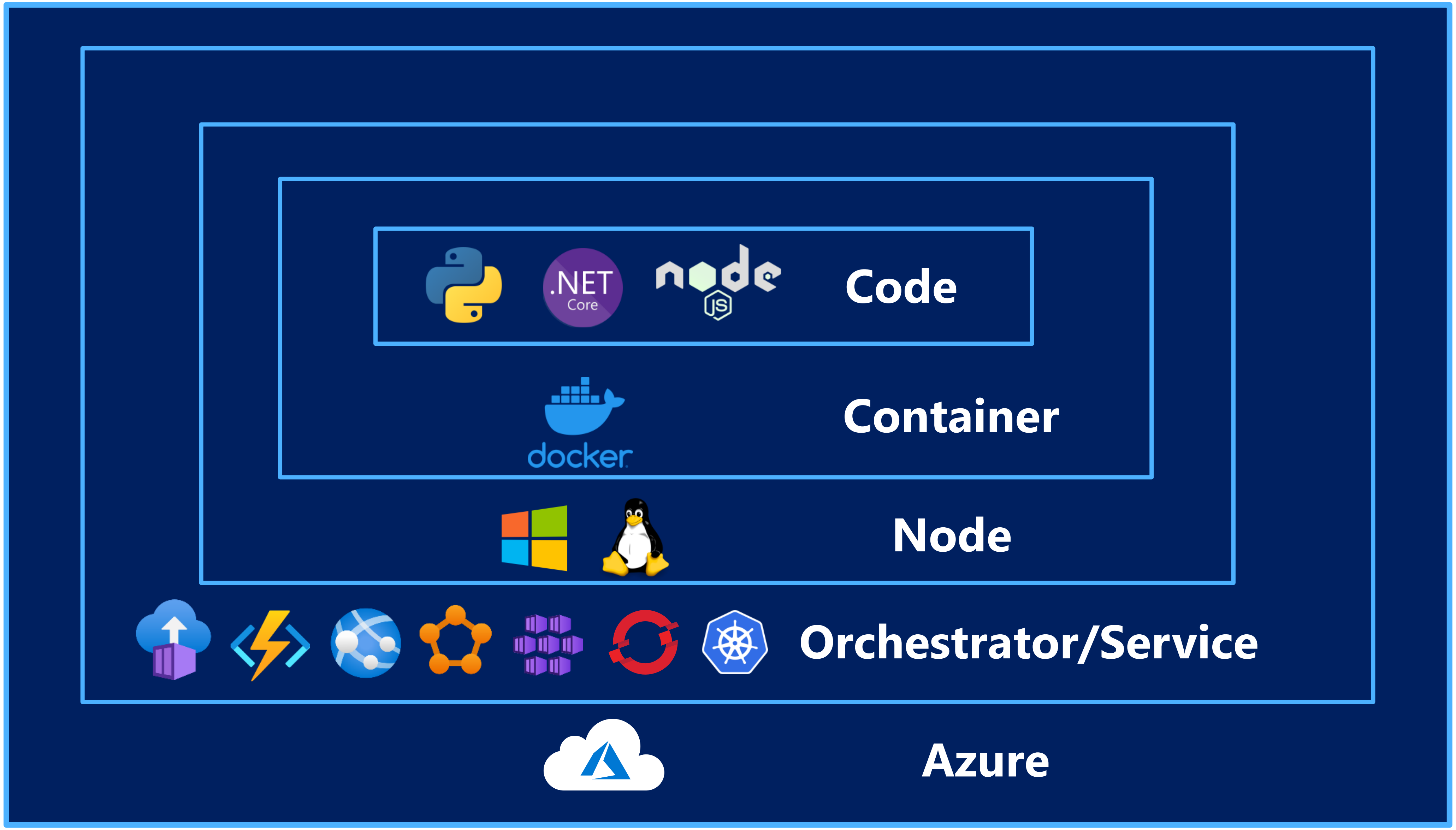
Orchestrator

Orchestrator misconfigurations
Weak credentials RBAC Gaps

Platform

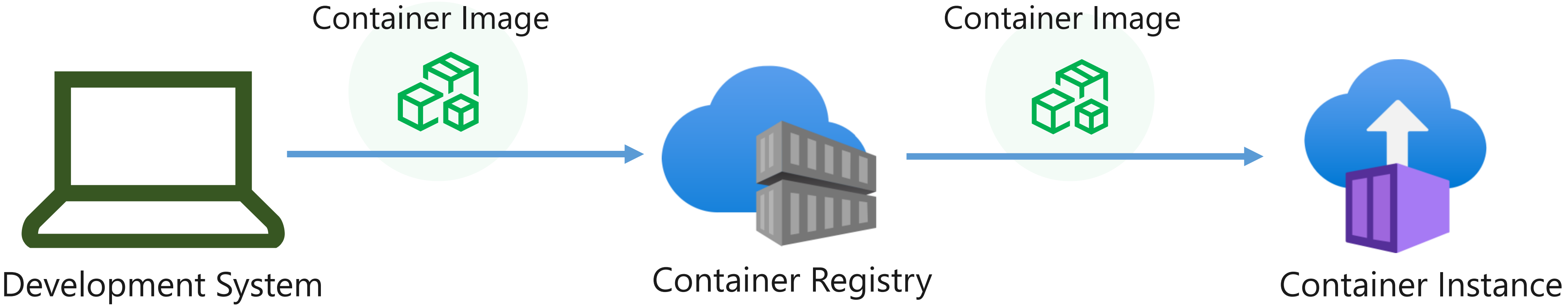
Platform misconfigurations
Service misconfigurations
Overprivileged access Platform bugs

Azure Containerization – Security Models



Azure Container Instances (ACI) Overview

- Azure Container Instances offers the fastest and simplest way to run a container in Azure
  - Doesn’t require IaaS VM provisioning and ongoing maintenance
  - Faster startup time compared with VMs
- Ideal for isolated containerized workloads that does not require orchestration
  - Simple applications; Task automation; Build jobs
- Supports Linux and Windows containers



Azure Kubernetes Service (AKS)

Fully managed Kubernetes orchestration service

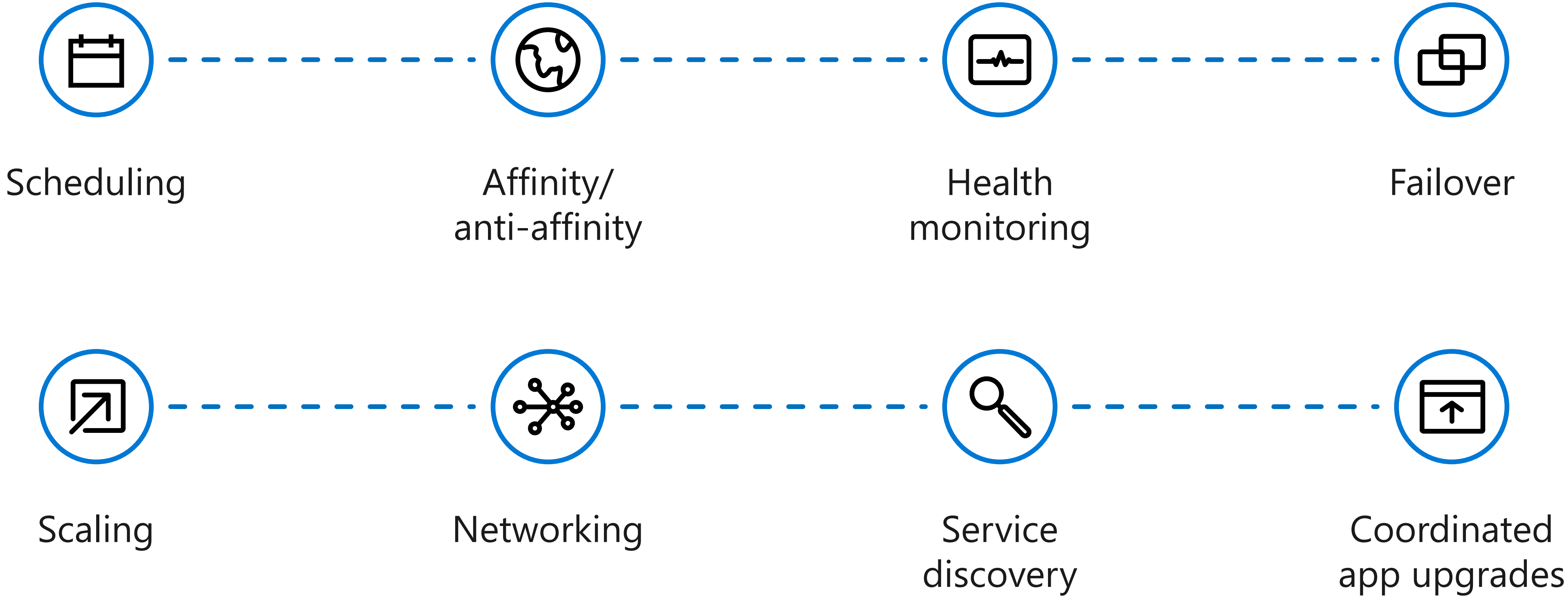
Simplified patching, auto scaling, auto updates

Use the full Kubernetes ecosystem (100% upstream)

Deeply integrated with Azure Dev Tools and services

Azure Kubernetes Service (AKS)

The elements of orchestration



Scheduling

Affinity/ anti-affinity

Health monitoring

Failover

Scaling

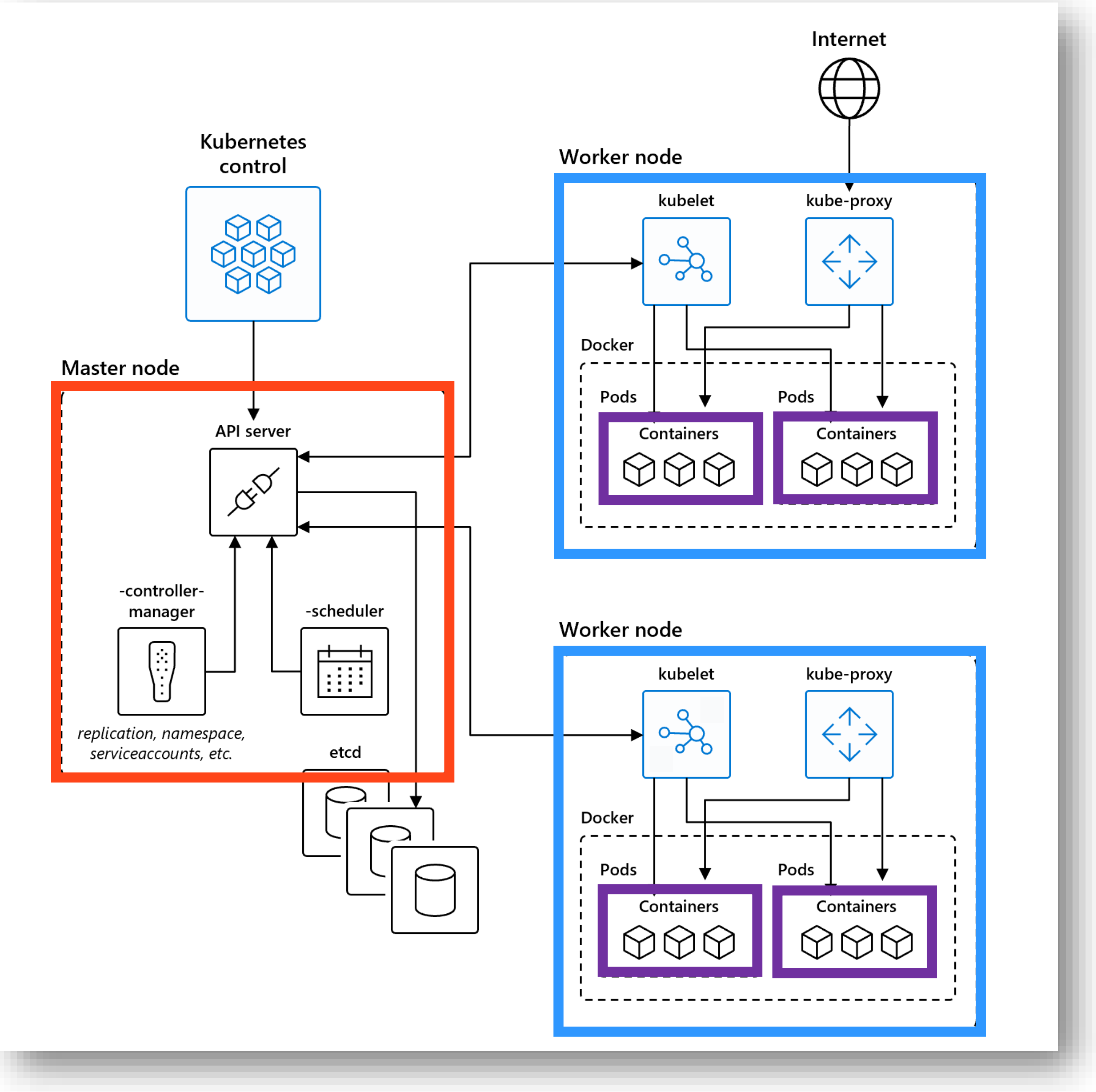
Networking

Service discovery

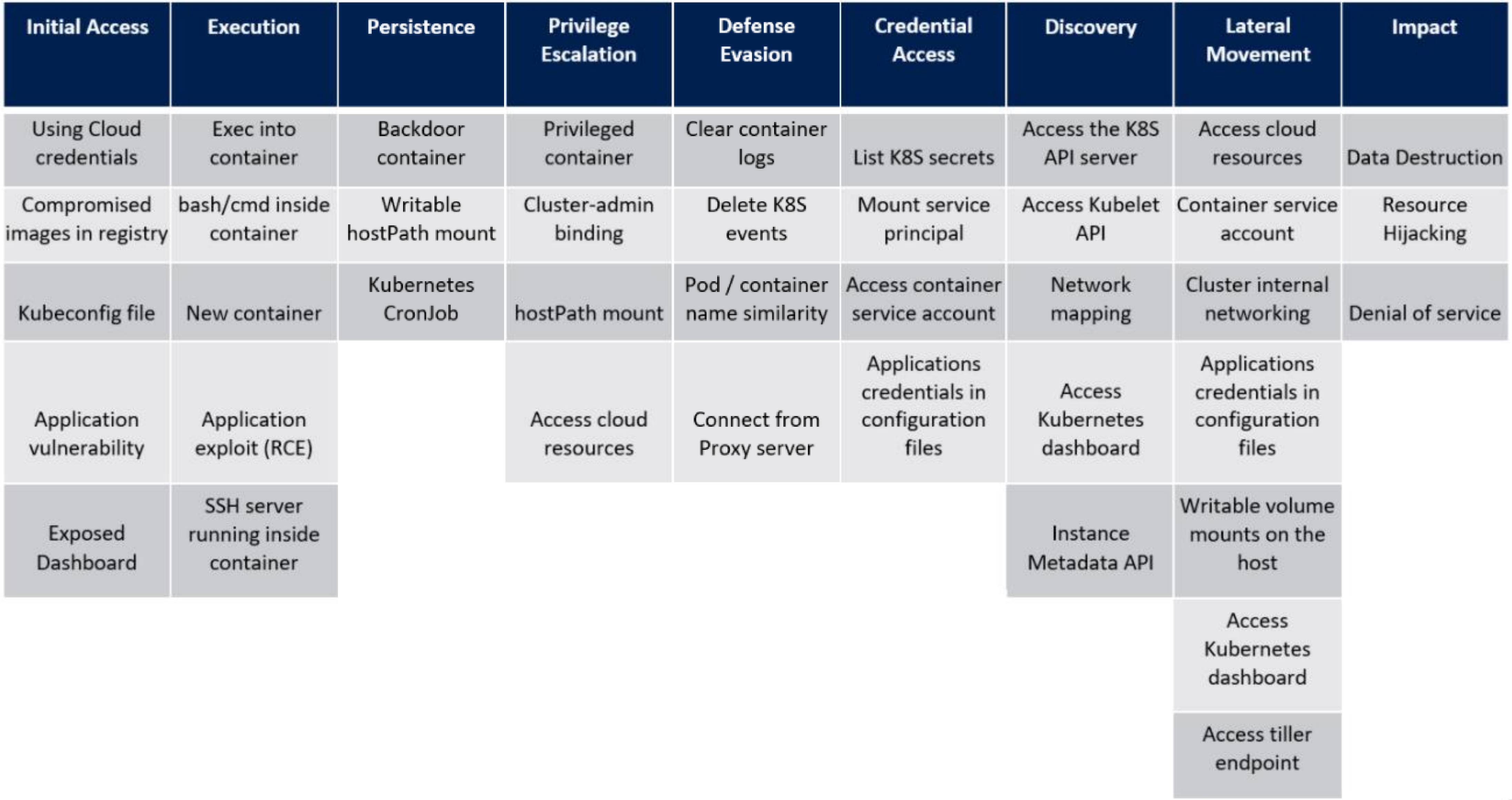
Coordinated app upgrades

AKS - Architecture

- The master which is responsible for the coordination and orchestration of the cluster
- The nodes which are VMs that runs the containerized workloads and supporting services
  - Linux (Optimized Ubuntu)
  - Windows (Optimized Windows Server 2019)
  - Mixed environment is supported
- A Pod represents a single instance of an application

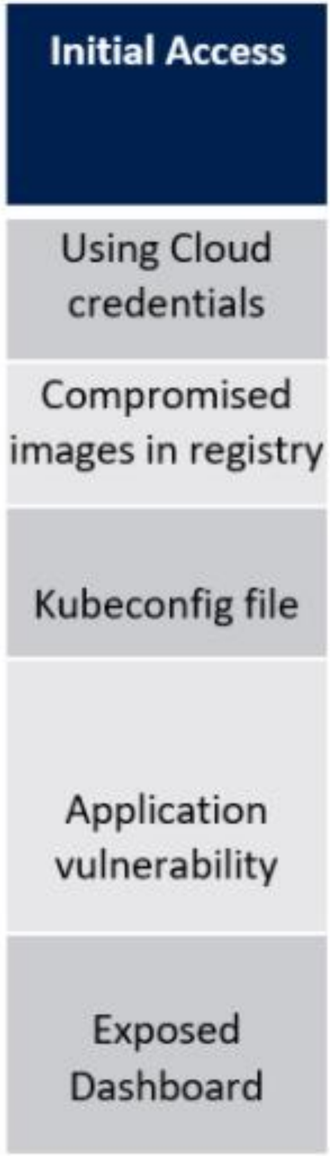


Attack Matrix



Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Impact
Using Cloud credentials	Exec into container	Backdoor container	Privileged container	Clear container logs	List K8S secrets	Access the K8S API server	Access cloud resources	Data Destruction
Compromised images in registry	bash/cmd inside container	Writable hostPath mount	Cluster-admin binding	Delete K8S events	Mount service principal	Access Kubelet API	Container service account	Resource Hijacking
Kubeconfig file	New container	Kubernetes CronJob	hostPath mount	Pod / container name similarity	Access container service account	Network mapping	Cluster internal networking	Denial of service
Application vulnerability	Application exploit (RCE)		Access cloud resources	Connect from Proxy server	Applications credentials in configuration files	Access Kubernetes dashboard	Applications credentials in configuration files	
Exposed Dashboard	SSH server running inside container					Instance Metadata API	Writable volume mounts on the host	
							Access Kubernetes dashboard	
							Access tiller endpoint	

Attack Matrix - Initial Access



Initial Access

Using Cloud credentials

Compromised images in registry

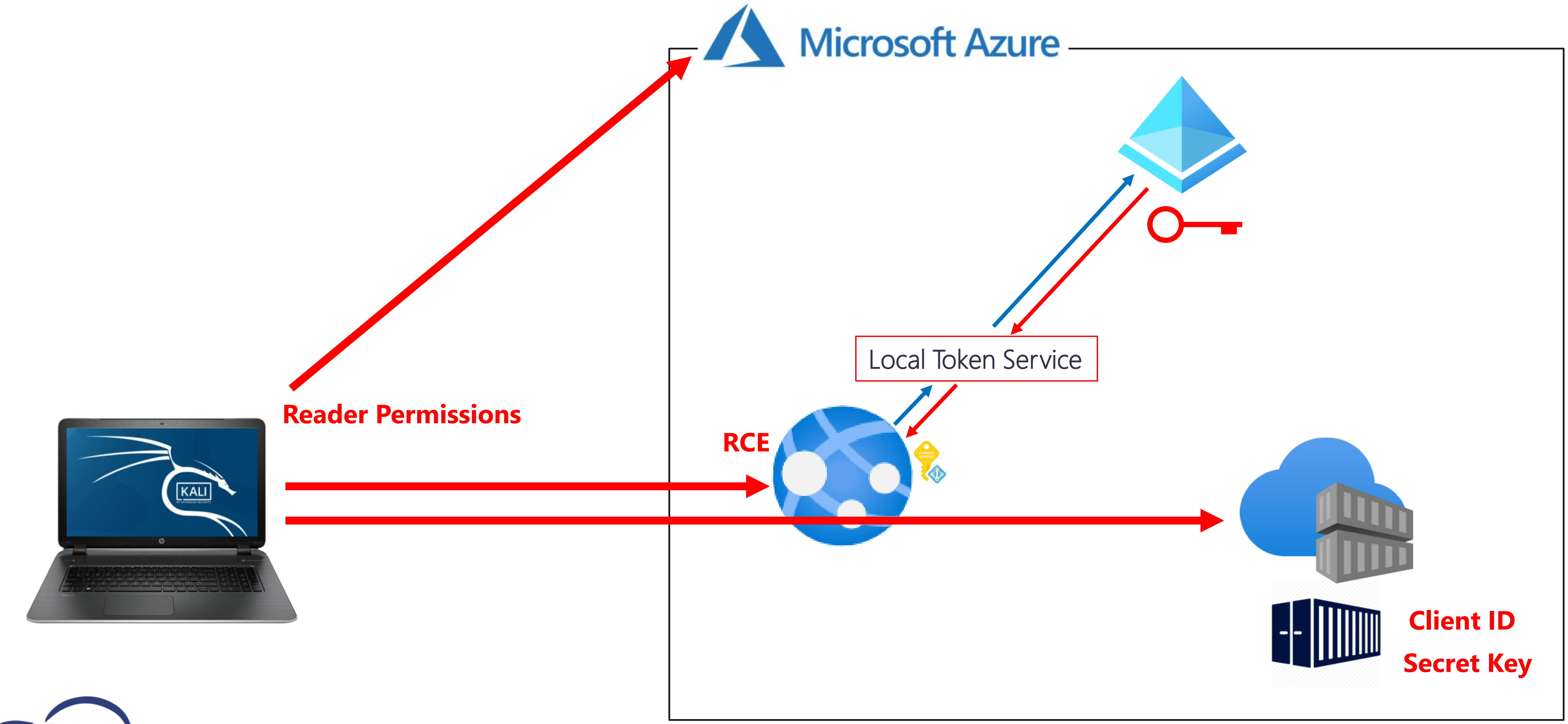
Kubeconfig file

Application vulnerability

Exposed Dashboard

- Exposed or stolen cloud credential used to compromise containerization services
- Unsafe or compromised images created by developers OR downloaded from a public image registry and used to launch containers
- The file could be stolen from a compromised admin client to gain unauthorized access to the Kubernetes cluster
- Containerized applications that are public-facing can allow initial attacker access, and if these containers have vulnerabilities
- The Kubernetes Dashboard is a web-based user interface that can be used for managing cluster resources. Public exposure could lead to compromise

ACR Privilege Escalation Scenario - DEMO



Microsoft Azure
Reader Permissions
RCE
Local Token Service
Client ID
Secret Key