Octopus Deploy

Modernisation strategies for a hybrid cloud environment

Rob Pearson | October 2024

Hello from the Octonauts!

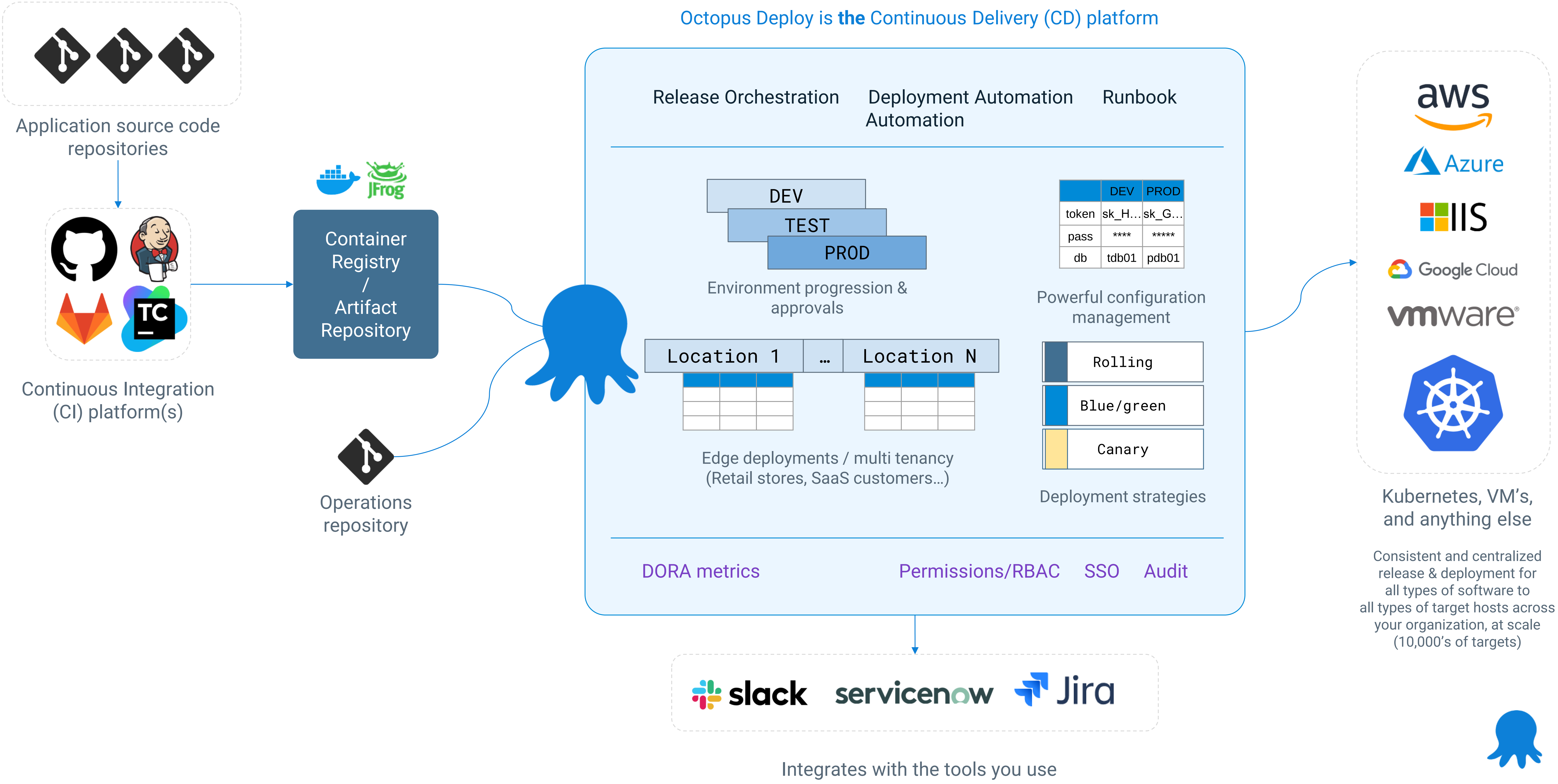
Rob Pearson

Principal DevOps Engineer

Octopus Deploy

- Octonaut since 2015
- LinkedIn: robpearsoncanada
- Email: rob.pearson@octopus.com
- GitHub: @robpearson

About Octopus Deploy



Application source code repositories

Continuous Integration (CI) platform(s)

Container Registry / Artifact Repository

Operations repository

Octopus Deploy is **the** Continuous Delivery (CD) platform

Release Orchestration

Deployment Automation

Runbook Automation

DEV

TEST

PROD

Environment progression & approvals

	DEV	PROD
token	sk_H…	sk_G…
pass	****	*****
db	tdb01	pdb01

Powerful configuration management

Location 1 … Location N

Edge deployments / multi tenancy (Retail stores, SaaS customers…)

Rolling

Blue/green

Canary

Deployment strategies

DORA metrics

Permissions/RBAC

SSO

Audit

Kubernetes, VM's, and anything else

Consistent and centralized release & deployment for all types of software to all types of target hosts across your organization, at scale (10,000's of targets)

Integrates with the tools you use

Welcome agenda

1 Introduction: Context and Domain

2 Challenges and Strategies

3 Case Study

4 Industry Trends

5 Questions

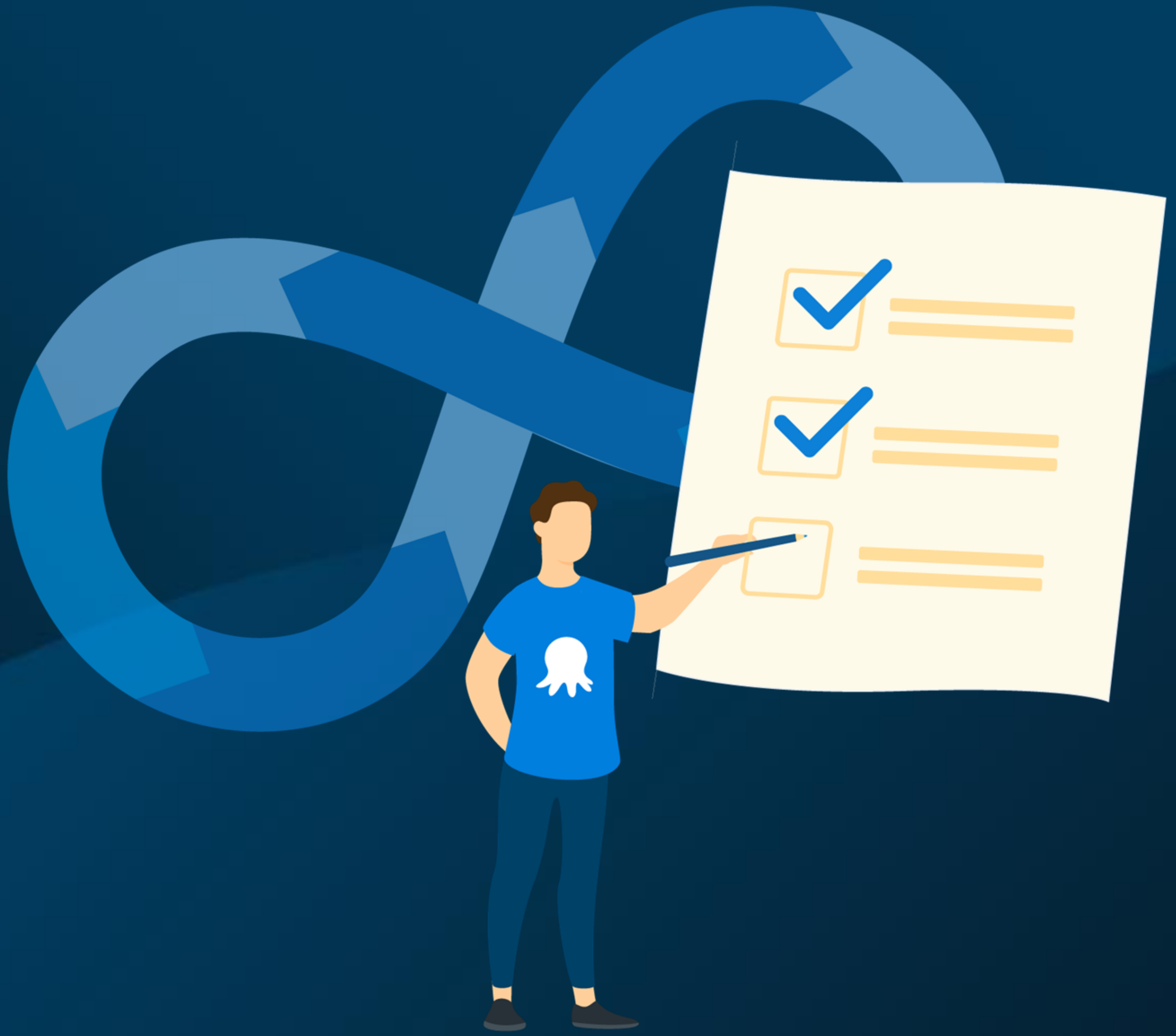
Introduction:

Context and Domain

Context and Domain

Definitions

- What is Hybrid Cloud?
- Differences between Brownfield and Greenfield projects
- IT Transformations and App Modernisation



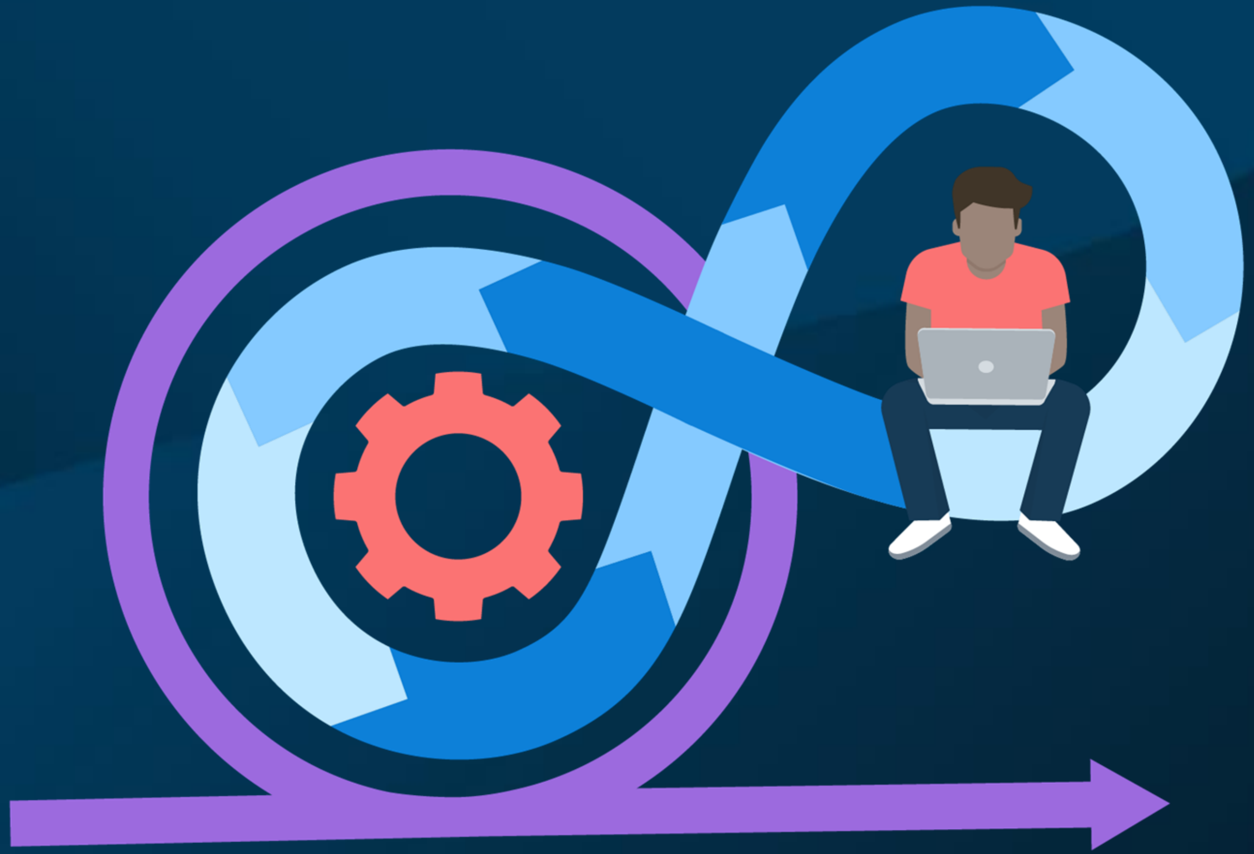
Challenges and Strategies

Challenges of Modernisation projects

Hybrid: Mix of new and old technology

Pain points

- Hard to change & scary to deploy
  - Deploy less often
  - Failures
- Cognitive Load + Solution complexity
  - Diverse technologies
  - Mixed infrastructure
  - Kubernetes



Cloud Smart vs Cloud First

Hybrid: Right mix of new and old technology

How to modernize?

- Team’s can’t rewrite everything
- Need to prioritise updates
- Evolutionary architecture
  - Set yourself up for success
  - Do not repeat past mistakes

Cloud Smart credit: https://medium.com/@dynotes/a-strategic-choice-cloud-smart-vs-cloud-first-39cc4f590a45

Modernisation Strategies in a Hybrid World

Assess and Analyse

- Prioritize changes (Cloud Smart)
- Iterative approach

Rehost or Replatform

- Consistent infrastructure
- Containers, K8s etc.

Automate everything

- Developer experience (productivity)
- Testing
- Pipelines as Code

Measure what matters

- DORA Metrics
- Fitness functions

Modernisation

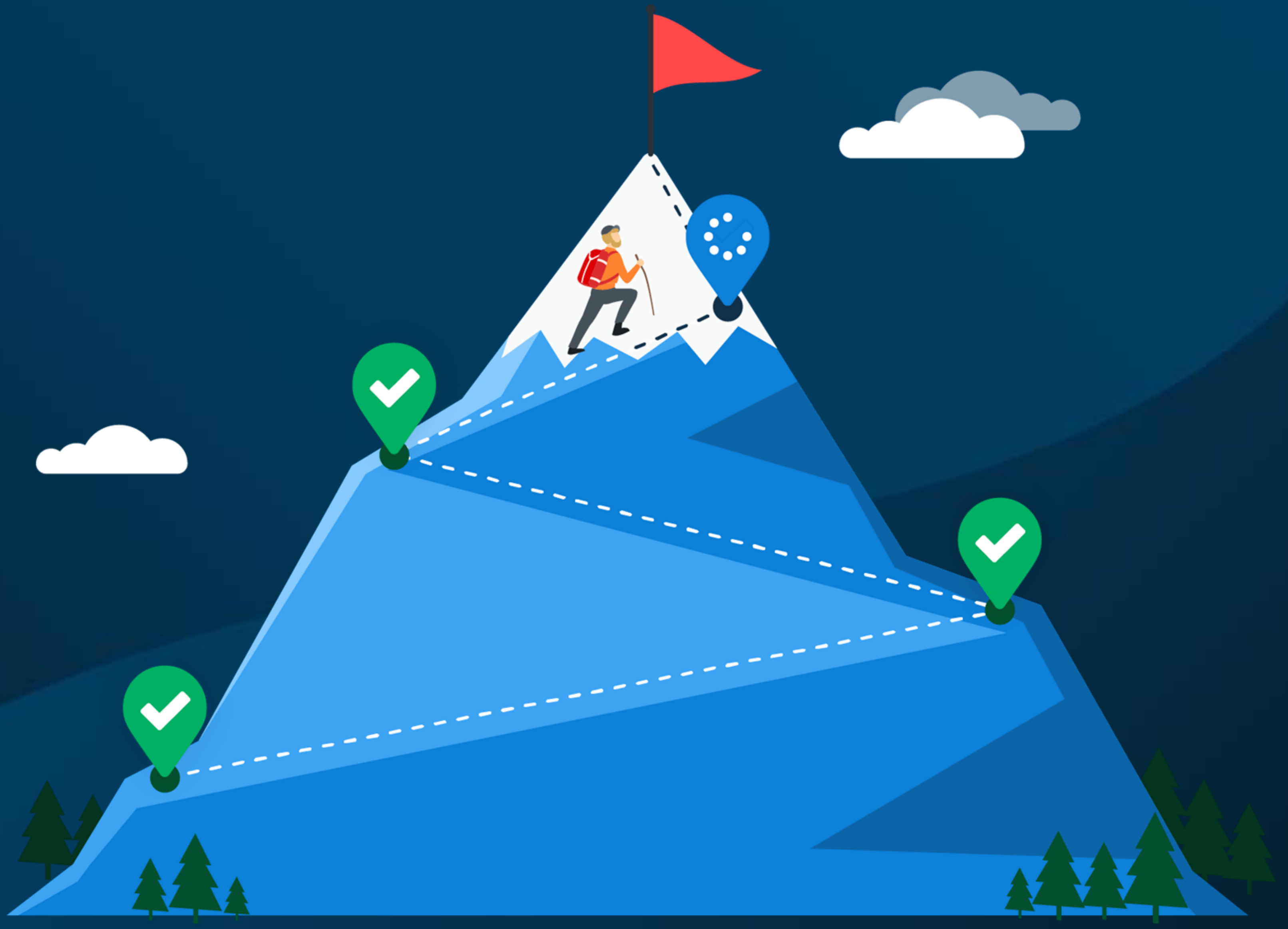
Case Study

Modernisation Walk Through

Goal: Build confidence to modernise and shift to the cloud (Cloud smart)

- Start with the CI process
- Include testing
- Include DevSecOps processes
- (Security scanning)
- Automate with your build tool. Scripts.

Just get started



Hybrid System

- Understand your goal
- Pick a starting point.

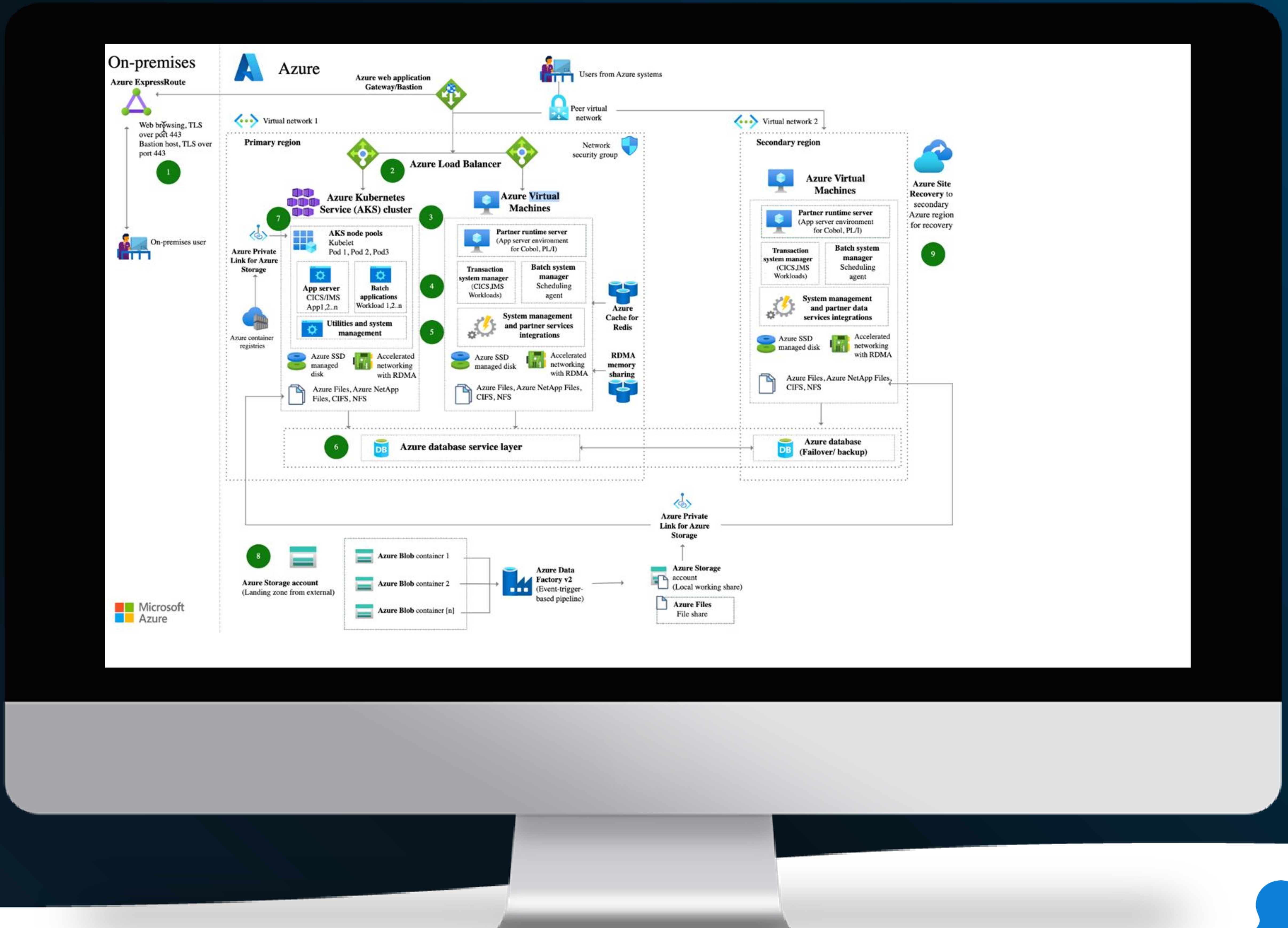
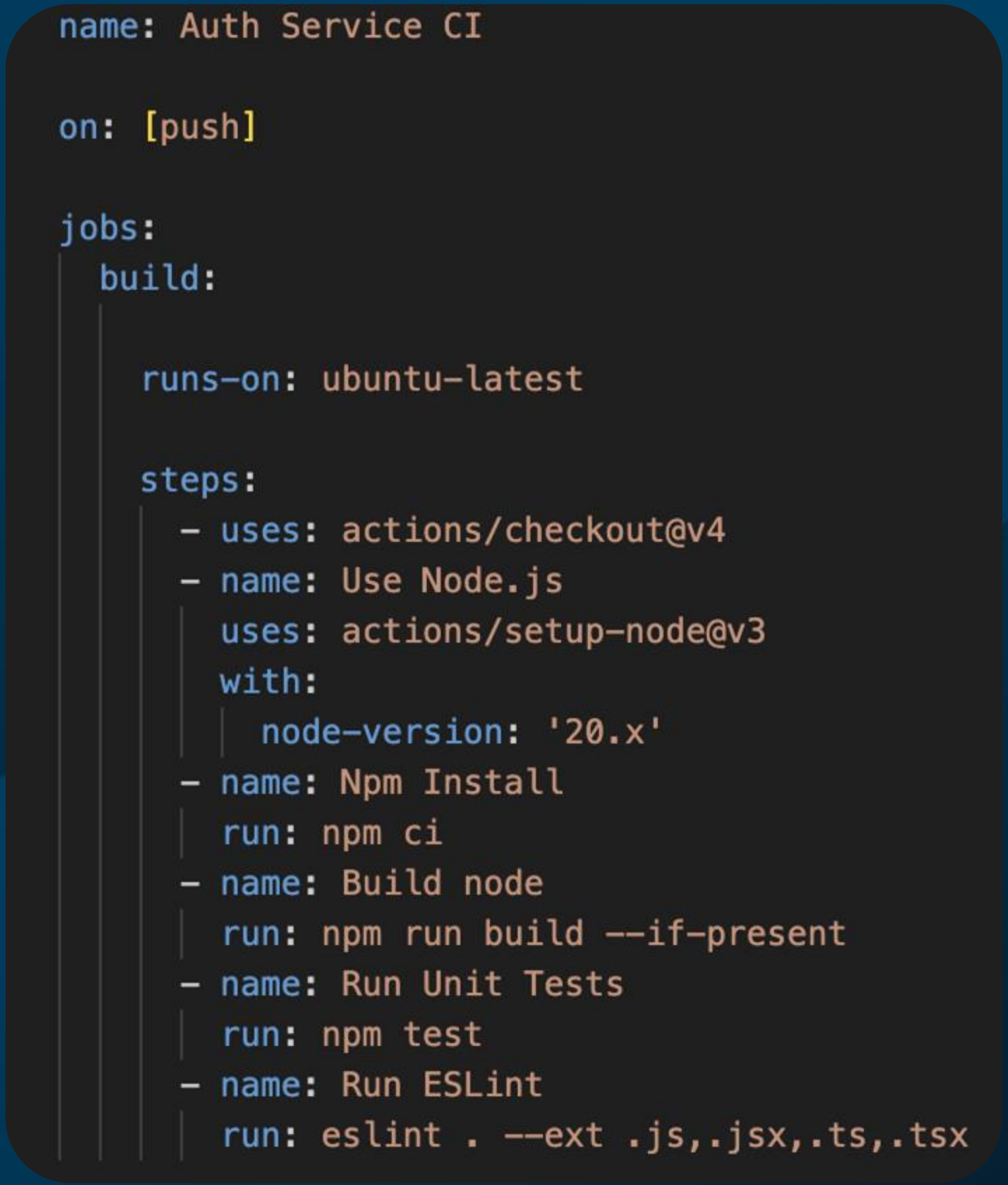

On-premises
Azure ExpressRoute
Web browsing, TLS over port 443
Bastion host, TLS over port 443
1
On-premises user
Azure
Azure web application Gateway/Bastion
Users from Azure systems
Peer virtual network
Virtual network 1
Primary region
Azure Load Balancer
2
Network security group
Azure Kubernetes Service (AKS) cluster
7
Azure Private Link for Azure Storage
Azure container registries
AKS node pools
Kubelet
Pod 1, Pod 2, Pod3
App server
CICS/IMS
App1,2..n
Batch applications
Workload 1,2..n
Utilities and system management
Azure SSD managed disk
Accelerated networking with RDMA
Azure Files, Azure NetApp Files, CIFS, NFS
3
4
5
Azure Virtual Machines
Partner runtime server
(App server environment for Cobol, PL/I)
Transaction system manager
(CICS,IMS Workloads)
Batch system manager
Scheduling agent
System management and partner services integrations
Azure SSD managed disk
Accelerated networking with RDMA
Azure Files, Azure NetApp Files, CIFS, NFS
Azure Cache for Redis
RDMA memory sharing
6
Azure database service layer
Virtual network 2
Secondary region
Azure Virtual Machines
Partner runtime server
(App server environment for Cobol, PL/I)
Transaction system manager
(CICS,IMS Workloads)
Batch system manager
Scheduling agent
System management and partner data services integrations
Azure SSD managed disk
Accelerated networking with RDMA
Azure Files, Azure NetApp Files, CIFS, NFS
Azure database
(Failover/ backup)
Azure Site Recovery to secondary Azure region for recovery
9
Azure Private Link for Azure Storage
8
Azure Storage account
(Landing zone from external)
Azure Blob container 1
Azure Blob container 2
Azure Blob container [n]
Azure Data Factory v2
(Event-trigger-based pipeline)
Azure Storage account
(Local working share)
Azure Files
File share
Microsoft Azure

Image credit: https://learn.microsoft.com/en-us/azure/architecture/example-scenario/mainframe/mainframe-rehost-architecture-azure

Stage 1 - Automate

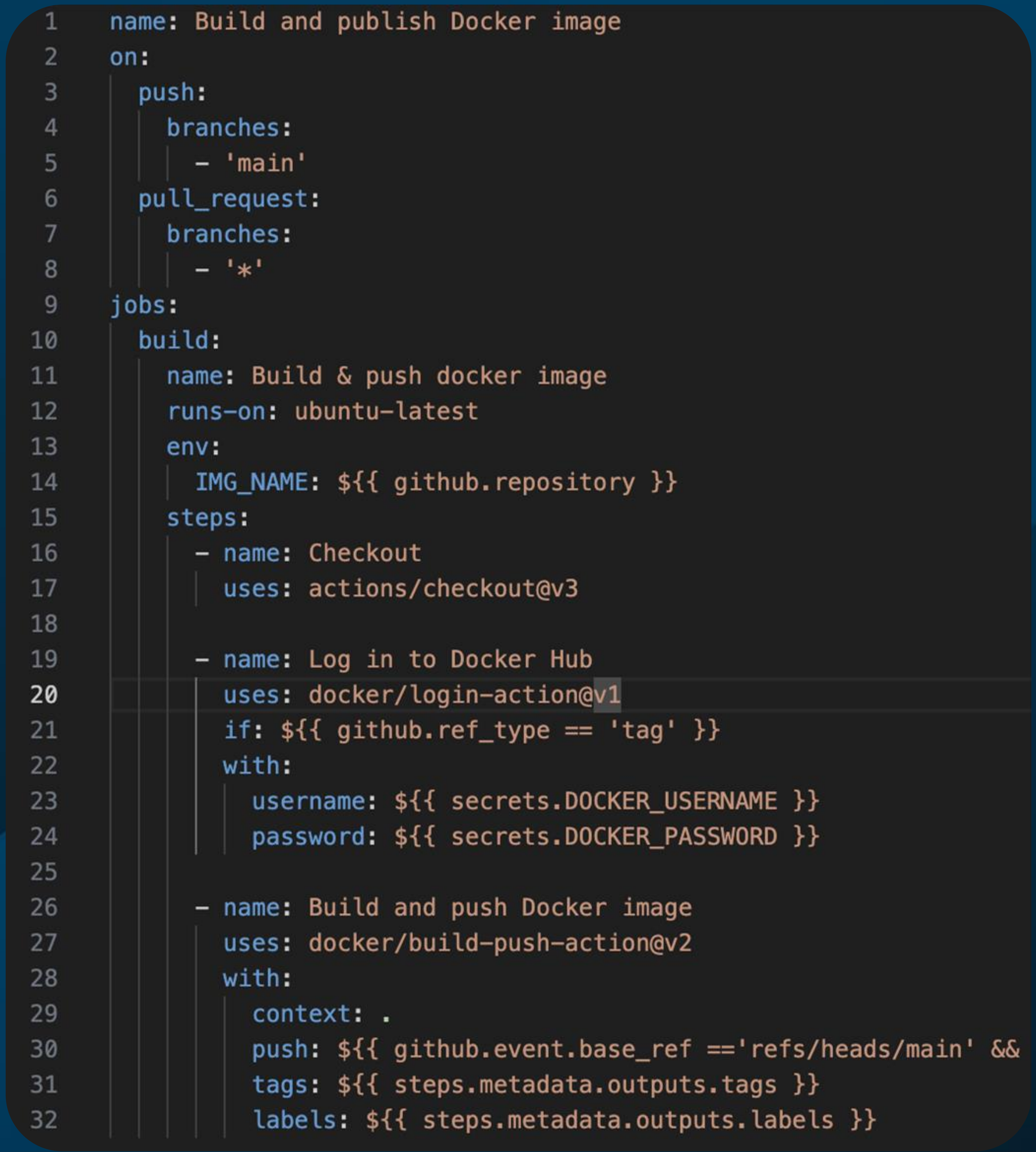
- Auth Service
  - High change rate
  - Frequent (slow) deploys
- Build
- Unit tests
- Static code analysis



```yaml
name: Auth Service CI

on: [push]

jobs:
  build:

    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v4
      - name: Use Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '20.x'
      - name: Npm Install
        run: npm ci
      - name: Build node
        run: npm run build --if-present
      - name: Run Unit Tests
        run: npm test
      - name: Run ESLint
        run: eslint . --ext .js,.jsx,.ts,.tsx
```

Stage 2 - Replatform

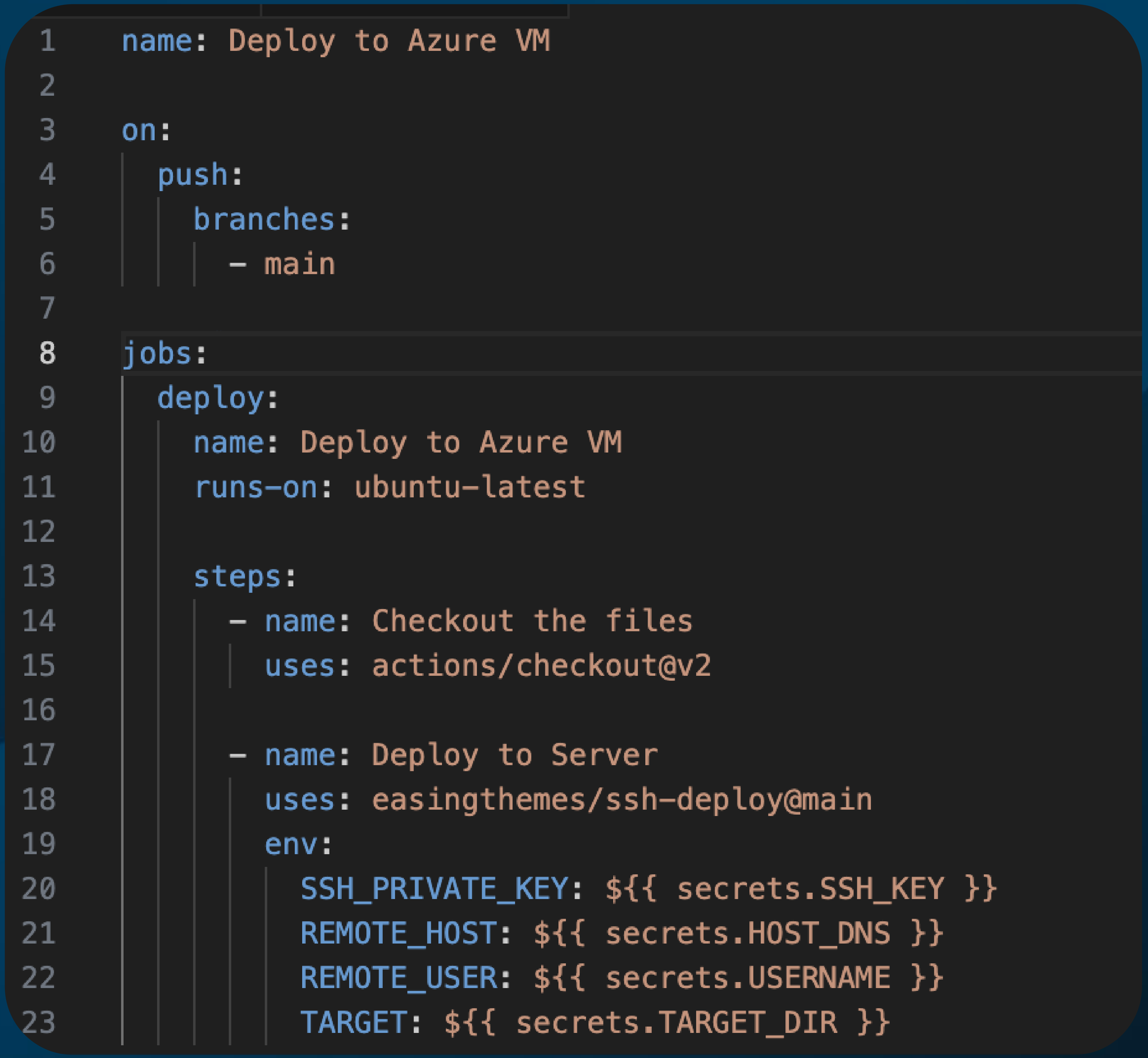
- Containerize
- Push to Registry
- Image Scanning



```yaml
name: Build and publish Docker image
on:
  push:
    branches:
      - 'main'
  pull_request:
    branches:
      - '*'
jobs:
  build:
    name: Build & push docker image
    runs-on: ubuntu-latest
    env:
      IMG_NAME: ${{ github.repository }}
    steps:
      - name: Checkout
        uses: actions/checkout@v3

      - name: Log in to Docker Hub
        uses: docker/login-action@v1
        if: ${{ github.ref_type == 'tag' }}
        with:
          username: ${{ secrets.DOCKER_USERNAME }}
          password: ${{ secrets.DOCKER_PASSWORD }}

      - name: Build and push Docker image
        uses: docker/build-push-action@v2
        with:
          context: .
          push: ${{ github.event.base_ref =='refs/heads/main' &&
          tags: ${{ steps.metadata.outputs.tags }}
          labels: ${{ steps.metadata.outputs.labels }}
```

Stage 3 - Rehost

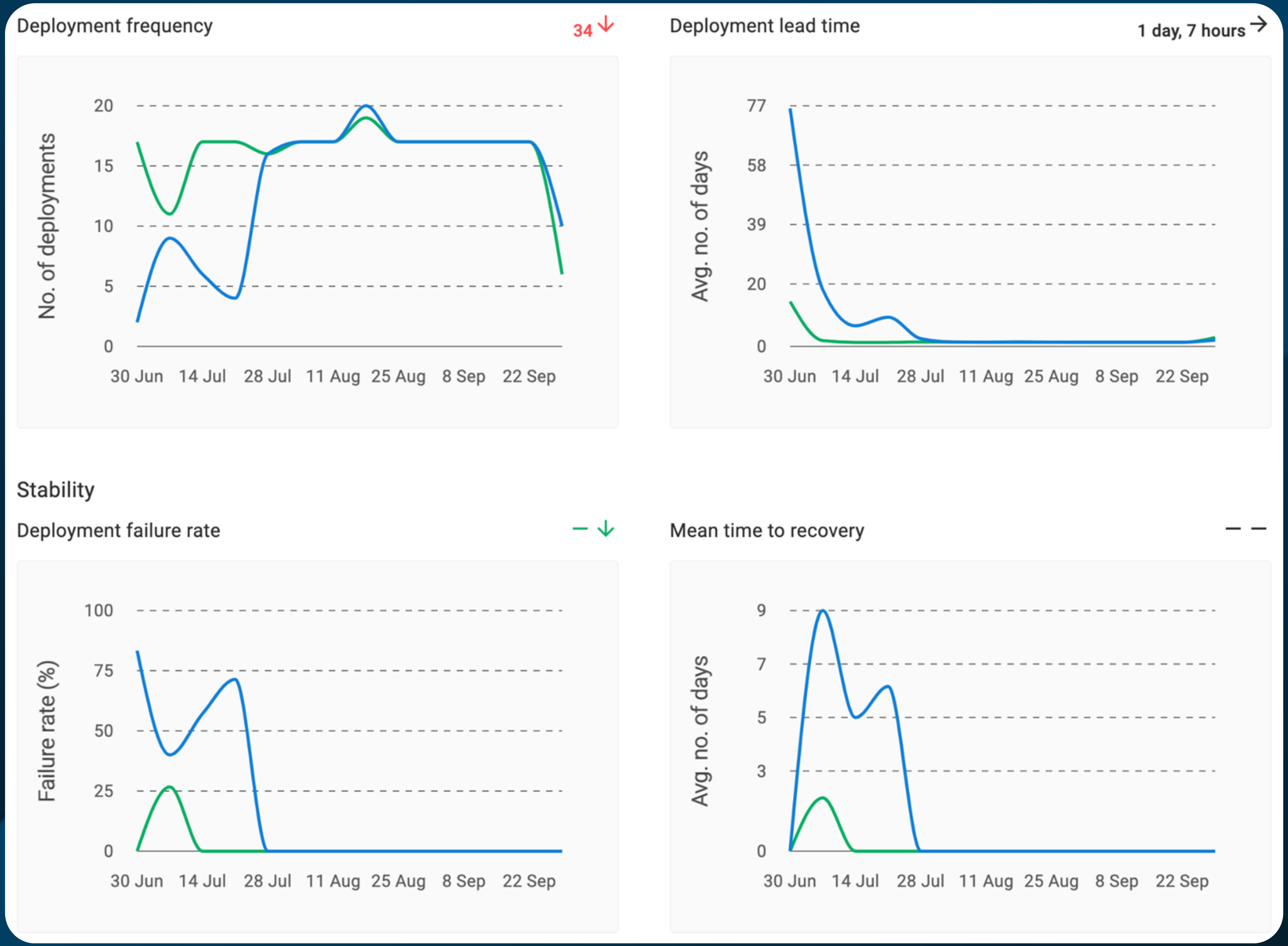
- Legacy service
- Running on server on premises
- Migrate to cloud (e.g. Azure VM)



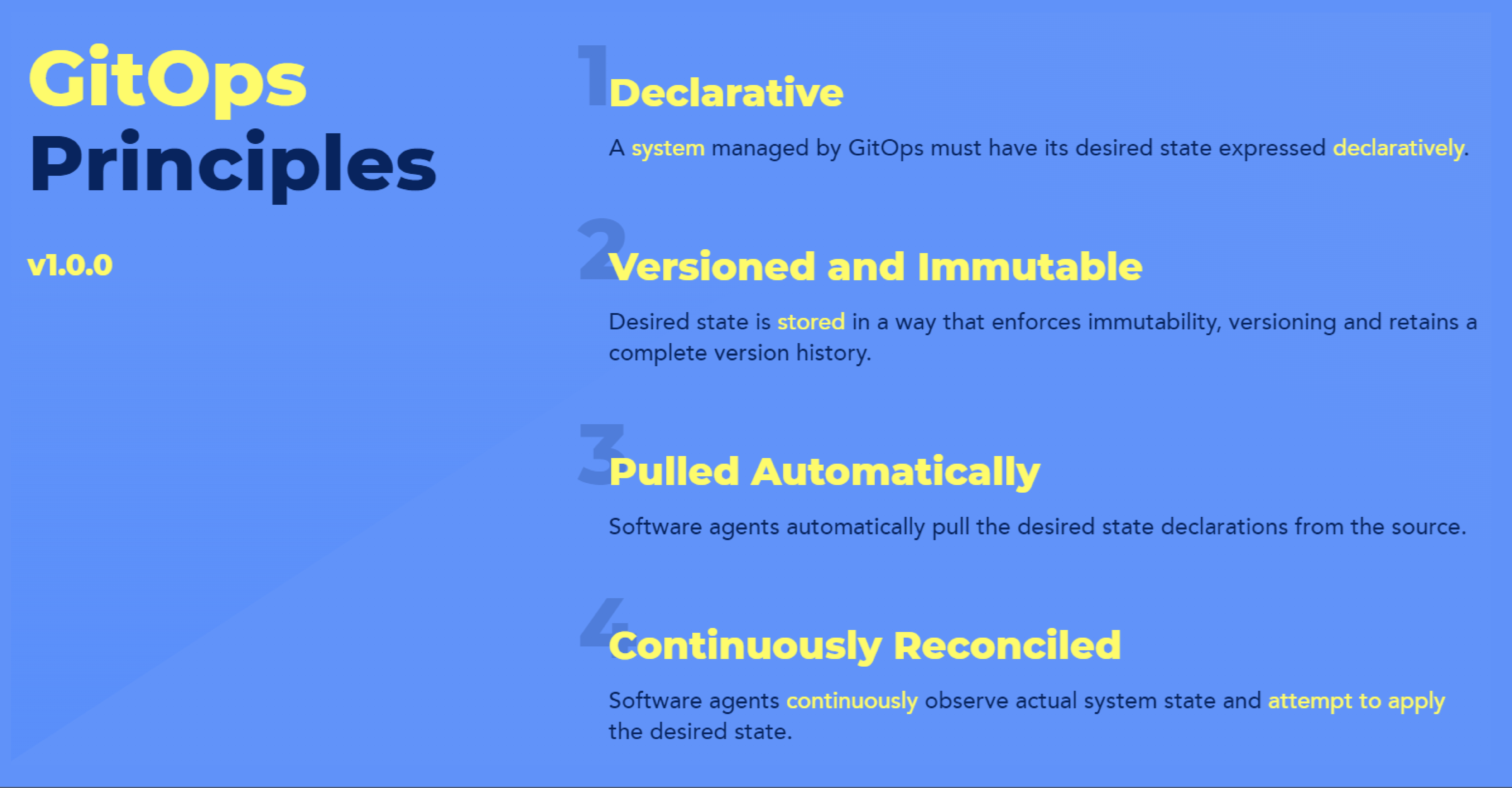
```yaml
name: Deploy to Azure VM

on:
  push:
    branches:
      - main

jobs:
  deploy:
    name: Deploy to Azure VM
    runs-on: ubuntu-latest

    steps:
      - name: Checkout the files
        uses: actions/checkout@v2

      - name: Deploy to Server
        uses: easingthemes/ssh-deploy@main
        env:
          SSH_PRIVATE_KEY: ${{ secrets.SSH_KEY }}
          REMOTE_HOST: ${{ secrets.HOST_DNS }}
          REMOTE_USER: ${{ secrets.USERNAME }}
          TARGET: ${{ secrets.TARGET_DIR }}
```

Rinse & Repeat

- Continuously measure
- Build feedback loops
- Apply lessons learned



Industry Trends



GitOps Principles

v1.0.0

1 Declarative

A system managed by GitOps must have its desired state expressed declaratively.

2 Versioned and Immutable

Desired state is stored in a way that enforces immutability, versioning and retains a complete version history.

3 Pulled Automatically

Software agents automatically pull the desired state declarations from the source.

4 Continuously Reconciled

Software agents continuously observe actual system state and attempt to apply the desired state.

From OpenGitOps.dev

GitOps tooling

What about AI? How can it help?

Modernise your applications

Leverage tools to help you update your applications so they can be containerised and run on modern infrastructure

Examples:

- GitHub co-pilot
- Docker extension

Modernise your pipelines

Leverage tools to deploy releases, get deploy status, diagnose deployment problems, toggle on/off feature toggles.

Examples
GitHub Co-pilot extensions

- Launchdarkly
- Octopus Deploy

Monitoring and alerting

Leverage tools to monitor your systems and alert you to errors. Ideally, suggest solutions.

Examples:

- DataDog
- Splunk
- Sumo logic
- GitHub co-pilot extensions

Thank you

Rob Pearson @ Octopus.com

Come say hello at our booth!

Octopus Deploy