welcome/to

//denver/dev/day

NEUDESIC an IBM Company

LUNAVI

Microsoft

UNO PLATFORM

Serverless or Kubernetes? I'll Take Both with Azure Container Apps

Mike Douglas

VP Digital Consulting – Engineering

mdouglas@lunavi.com

@mikedouglasdev

Session Goals

- What is Azure Container Apps
- How to use it
- When to use Azure Container Apps vs other options

About Mike Douglas

- Solution Consultant and VP Digital Consulting – Engineering at Lunavi
- Microsoft MVP – Developer Technologies – DevOps
- Organizer of Omaha DevOps Meetup
- Competitive Robotics Club Coordinator for 7th – 8th Graders
- @mikedouglasdev on twitter

What is Azure Container Apps

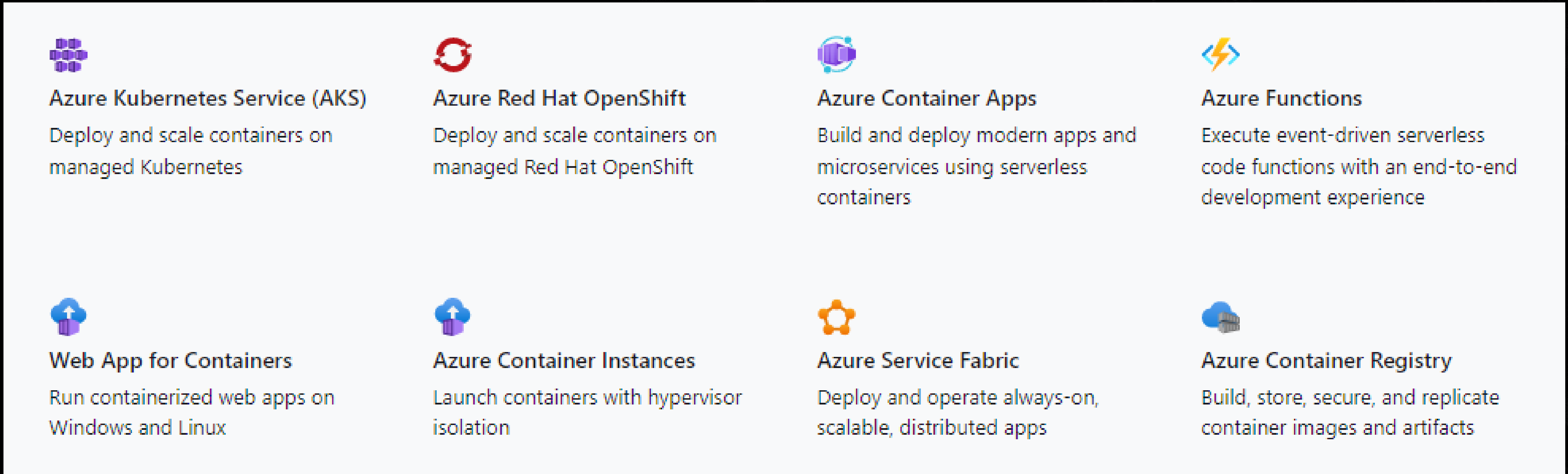
IAAS vs PAAS vs Serverless

- IAAS
  - VMs
- Responsible for
  - Patching
  - Backups
  - OS
  - Security
  - Pay for reserved compute

- PAAS
  - App Services
  - Azure SQL
- Responsible for
  - IAM/RBAC
  - Reserving Compute
  - Network Access Restrictions
  - Pay for reserved compute

- Serverless
  - Azure Functions
  - Azure Container Apps
- Responsible for
  - Only pay for what you use

Azure Container Offerings



Azure Kubernetes Service (AKS)

Deploy and scale containers on managed Kubernetes

Azure Red Hat OpenShift

Deploy and scale containers on managed Red Hat OpenShift

Azure Container Apps

Build and deploy modern apps and microservices using serverless containers

Azure Functions

Execute event-driven serverless code functions with an end-to-end development experience

Web App for Containers

Run containerized web apps on Windows and Linux

Azure Container Instances

Launch containers with hypervisor isolation

Azure Service Fabric

Deploy and operate always-on, scalable, distributed apps

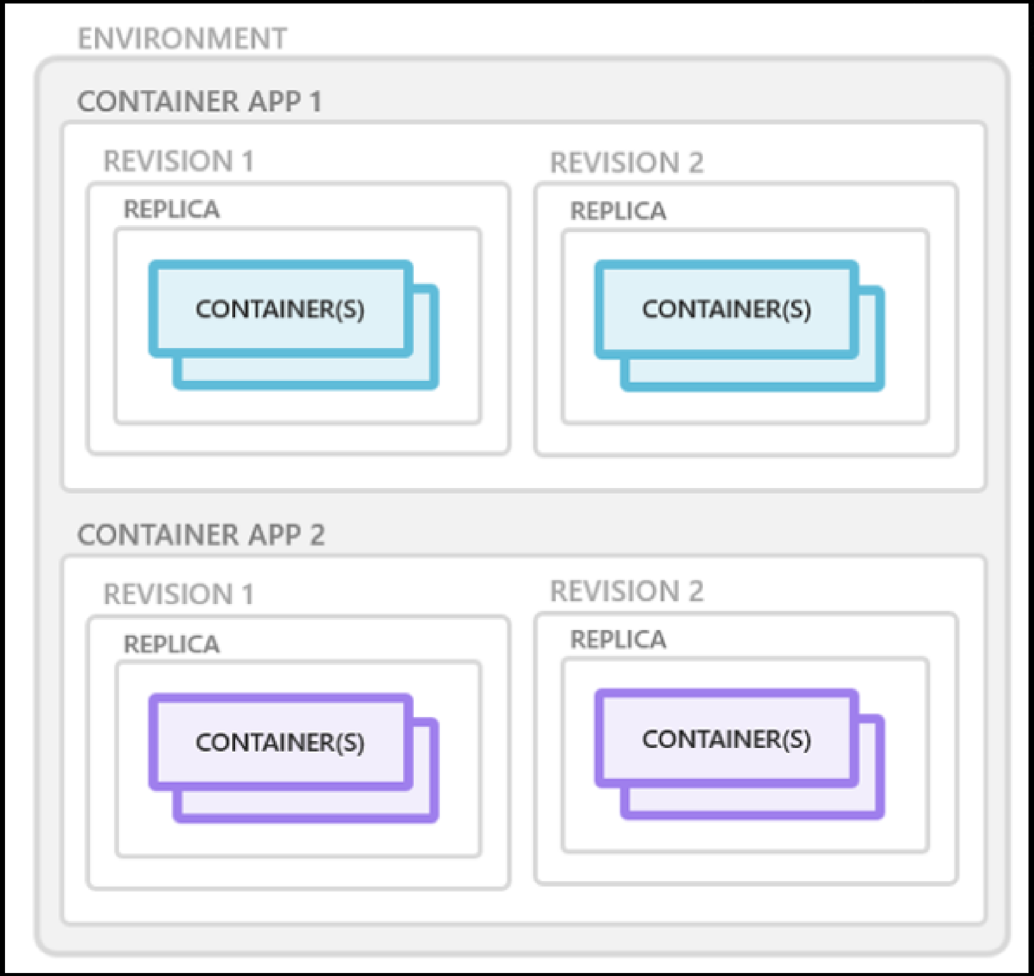
Azure Container Registry

Build, store, secure, and replicate container images and artifacts

Azure Container Apps

- Serverless Platform
- Powered by Kubernetes
- Run microservices and containerized applications
- Event-driven application architecture
  - Scale based HTTP, CPU/Memory, or KEDA
- Abstracts and simplifies features and benefits of Kubernetes
- Powerful and secure enough to run any workload

Azure Container Apps

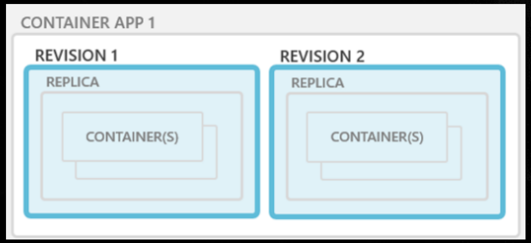


Containers

- Supports any Linux x86-64 container image (no required base)
- Can’t run privileged containers (root access)
- Specify vCPU and Memory
- Set Health Probes (Liveness, Readiness, Startup)
- Run as Pod

Revisions

- Immutable snapshot of a container app version
- Run multiple revisions concurrently
- Support A/B Testing and Blue/Green Deployments
- Some Application-scope changes that affects all revisions
- Revision labels allow keep same URL moving across revisions



Environments

- Secure boundary around groups of containers
- Utilizes same Virtual Network
- Reasons for
  - Managing related services
  - Applications and Services communicate “internally” with each other
- Reasons for different environments
  - Don’t want to share compute resources
  - Don’t want to communicate “internally”

Billing

- Pay for only what you use
- Scale to zero
- Pay per Container App, no billing at Environment
- First 2 million requests each month are free
- First 180,000 vCPU-seconds each month are free
- First 360,000 GiB-seconds each month are free
- Can pay for Dedicated Instance(s)
  - Cheaper idle time to keep instances warmed up
- No premium type SKU required to get more advanced features

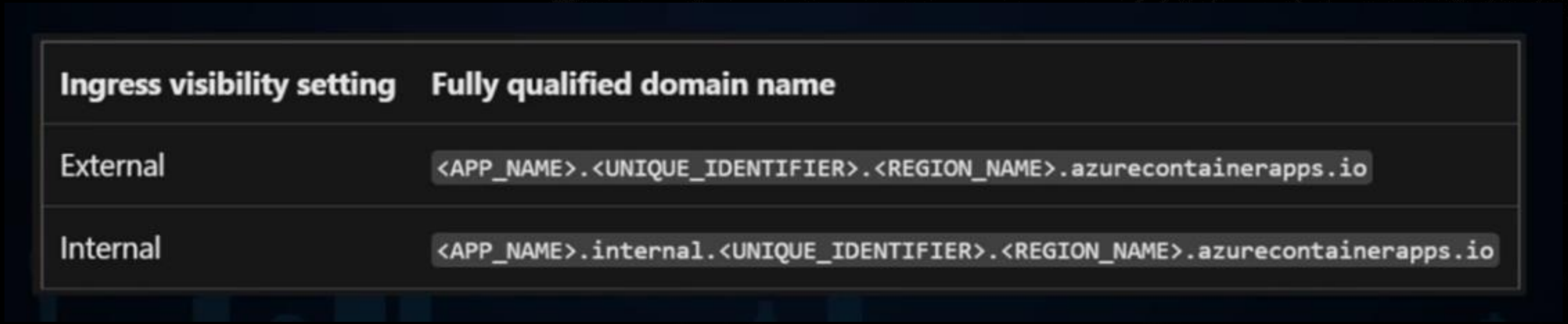
Demo

Create first container app

Using Azure Container Apps

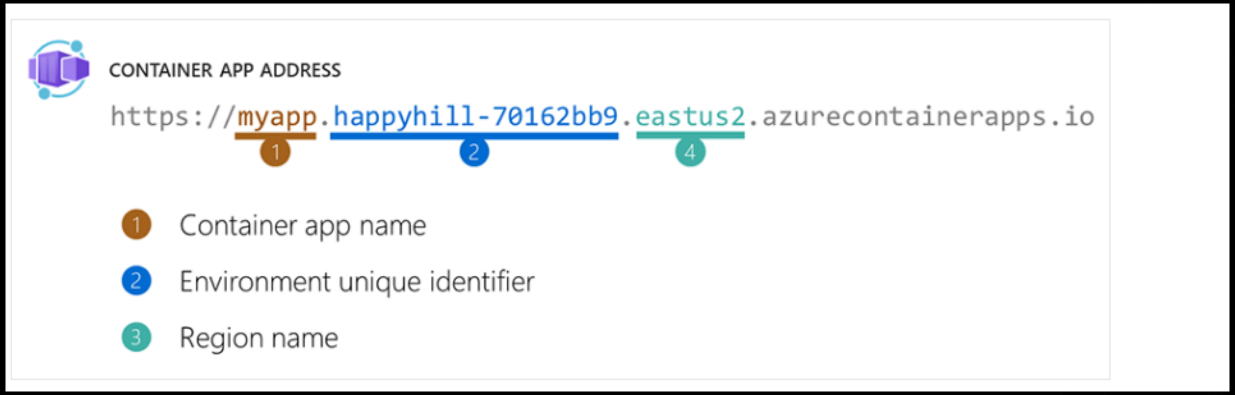
Ingress and Discoverability

- Ingress



Ingress visibility setting	Fully qualified domain name
External	`<APP_NAME>.<UNIQUE_IDENTIFIER>.<REGION_NAME>.azurecontainerapps.io`
Internal	`<APP_NAME>.internal.<UNIQUE_IDENTIFIER>.<REGION_NAME>.azurecontainerapps.io`

- Discoverability



CONTAINER APP ADDRESS

`https://myapp.happyhill-70162bb9.eastus2.azurecontainerapps.io`

1 Container app name

2 Environment unique identifier

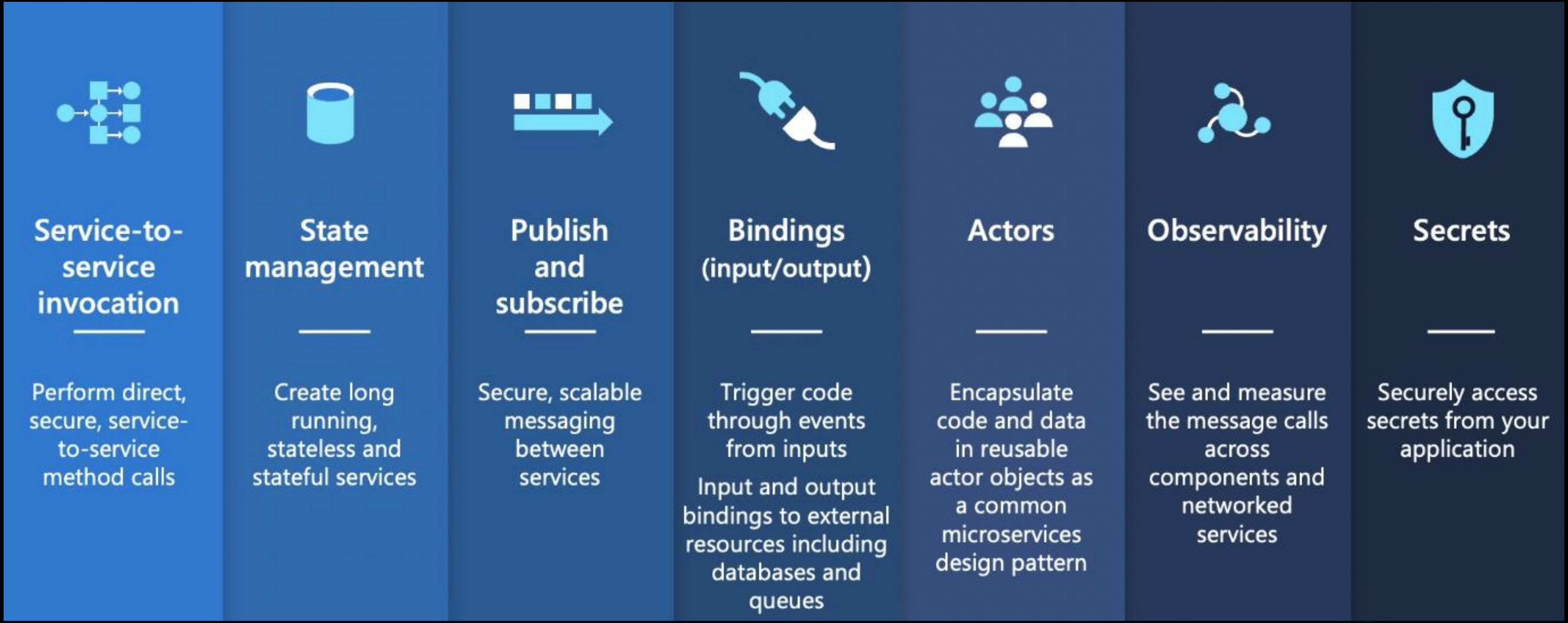
3 Region name

Demo

Create Blazor Weather App with .NET Weather Forecast API

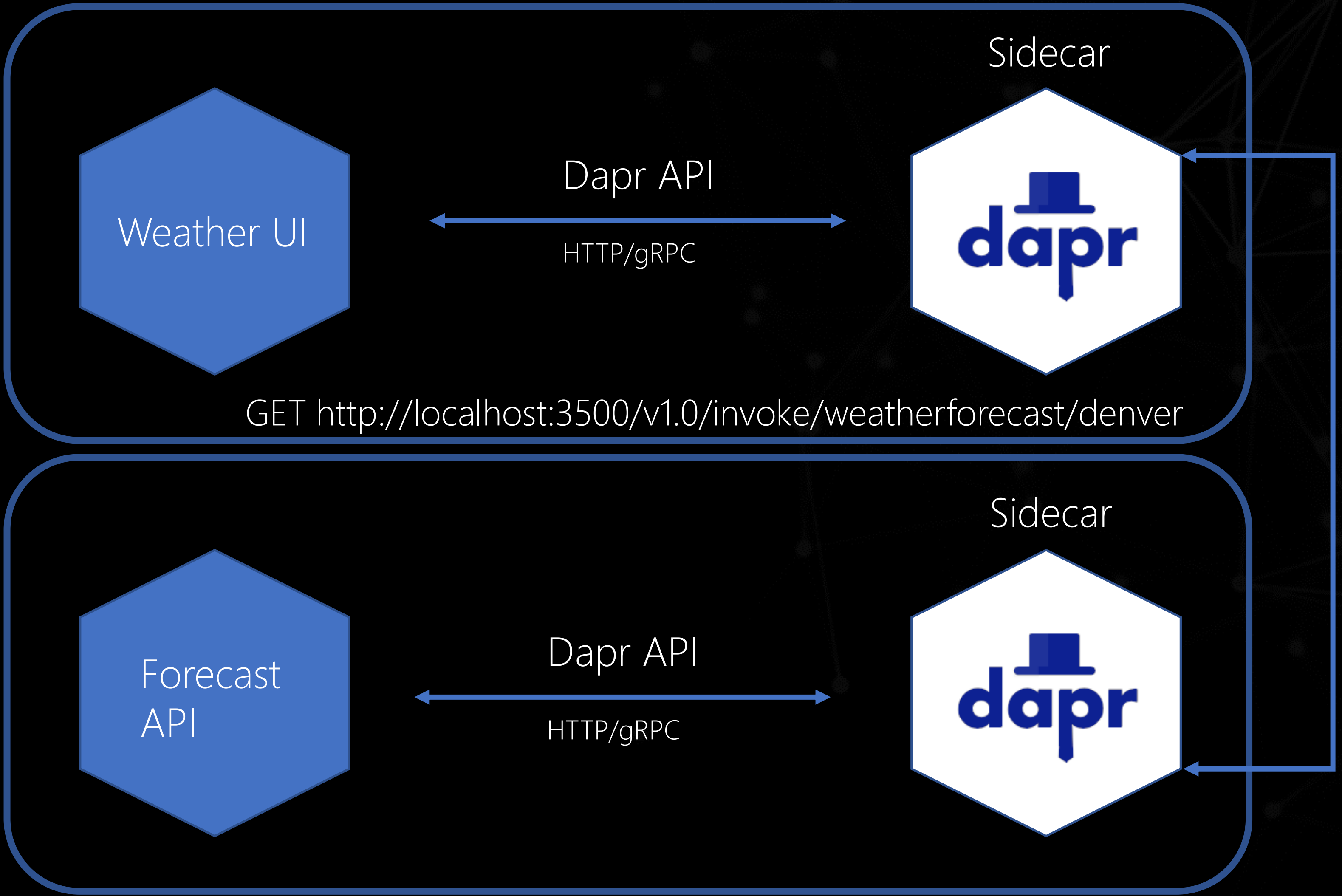
Dapr

- Full managed version of Dapr



Service-to-service invocation	State management	Publish and subscribe	Bindings (input/output)	Actors	Observability	Secrets
Perform direct, secure, service-to-service method calls	Create long running, stateless and stateful services	Secure, scalable messaging between services	Trigger code through events from inputs Input and output bindings to external resources including databases and queues	Encapsulate code and data in reusable actor objects as a common microservices design pattern	See and measure the message calls across components and networked services	Securely access secrets from your application

Dapr



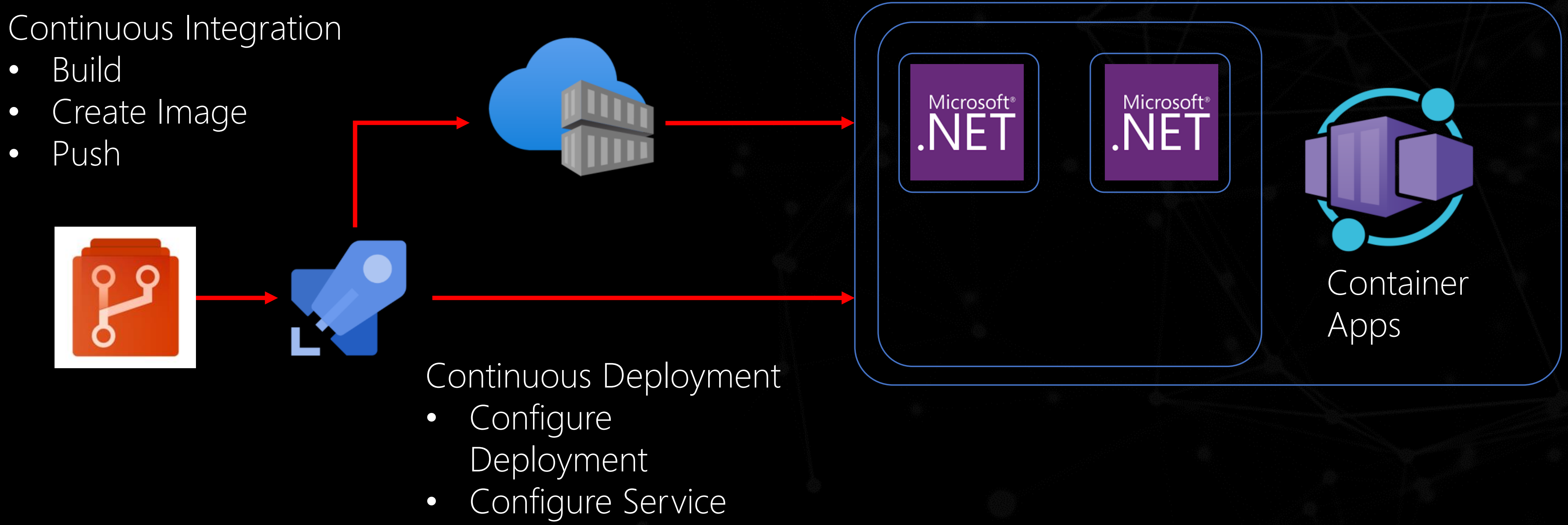
GET http://localhost:3500/v1.0/invoke/weatherforecast/denver

Invoke
SaveState
Publish
Secrets
Configuration

Demo

Using Dapr

ACA Pipeline



Continuous Integration

- Build
- Create Image
- Push

Continuous Deployment

- Configure Deployment
- Configure Service

When to use Azure Container Apps

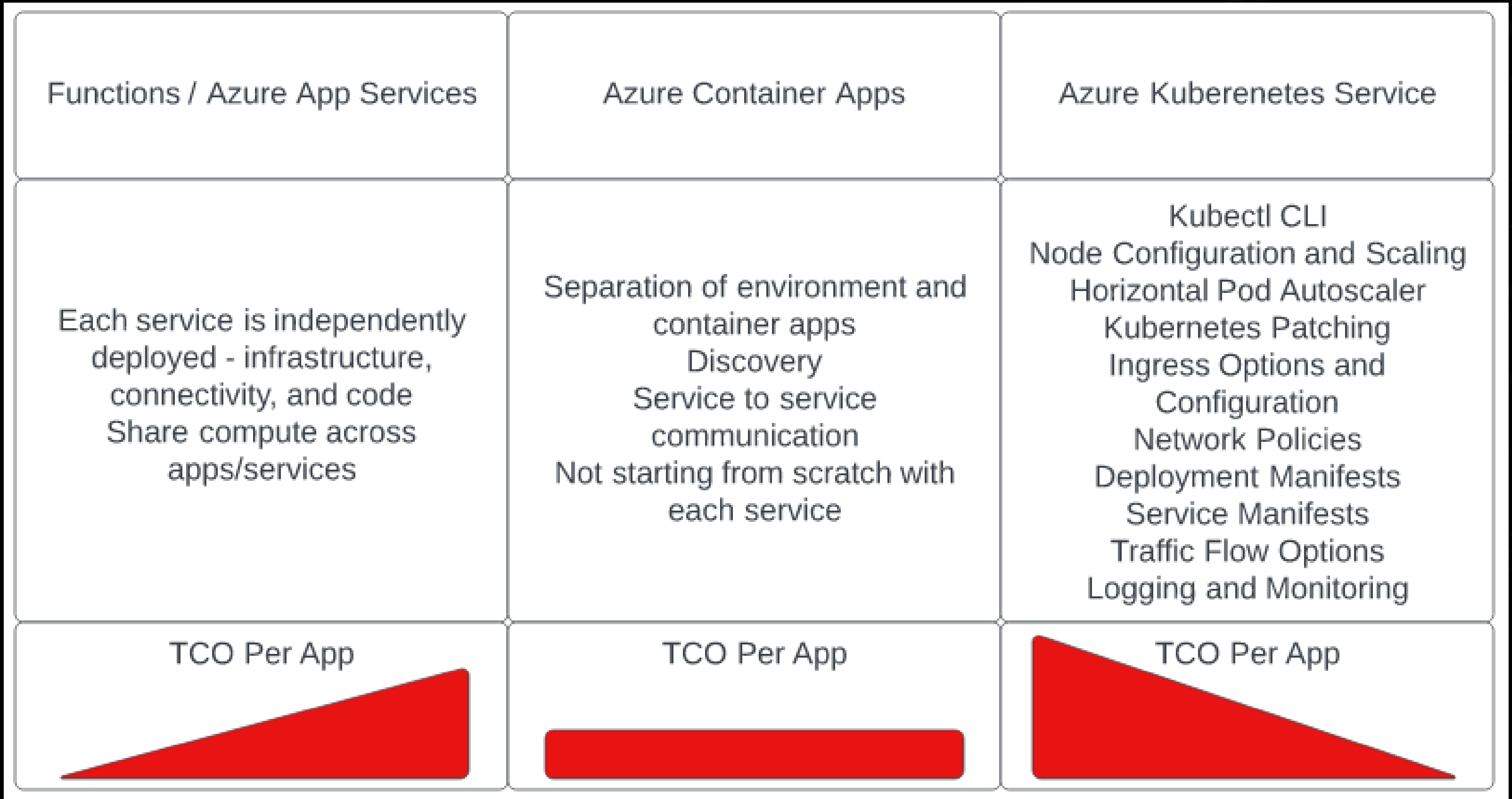
Containers OR No Containers

- No Containers
  - Build and Package to some sort of ZIP
  - Application binaries and dependencies only
  - Dev, Build Agents, and Hosting environment could all be slightly different
  - Environments
    - Azure Functions
    - Azure App Services
  - Application and dependencies vulnerability testing
  - No OS, hosting environment patching, only versioning of runtime/hosting

Containers OR No Containers

- Containers
  - Same container running local as in production
  - Portability across hosting environments, clouds
  - Immutable snapshot of application version
  - Container vulnerabilities in addition to application code and dependencies
  - More than just code but most complexities can be hidden from developers

Container Options



Functions / Azure App Services	Azure Container Apps	Azure Kuberenetes Service
Each service is independently deployed - infrastructure, connectivity, and code Share compute across apps/services	Separation of environment and container apps Discovery Service to service communication Not starting from scratch with each service	Kubectl CLI Node Configuration and Scaling Horizontal Pod Autoscaler Kubernetes Patching Ingress Options and Configuration Network Policies Deployment Manifests Service Manifests Traffic Flow Options Logging and Monitoring
TCO Per App	TCO Per App	TCO Per App

So would I?

- Use ACA over Azure Functions?
- Use ACA over AKS?
- Use ACA over App Services?

Review

Session Takeaways

- Understanding of how get started using Container Apps
- When to use Containers VS AKS Vs Azure Functions / App Services
- A/B and Blue / Green Deployments in action

Thank you

Mike Douglas

VP Digital Consulting – Engineering

mdouglas@lunavi.com

@mikedouglasdev