CNCF COIMBATORE MEETUP

Scaling without Limits

The Serverless way

21-Sep-2024

ABOUT ME

Sivasubramanian Bagavathiappan (Siva)

Senior Architect & SRE Leader

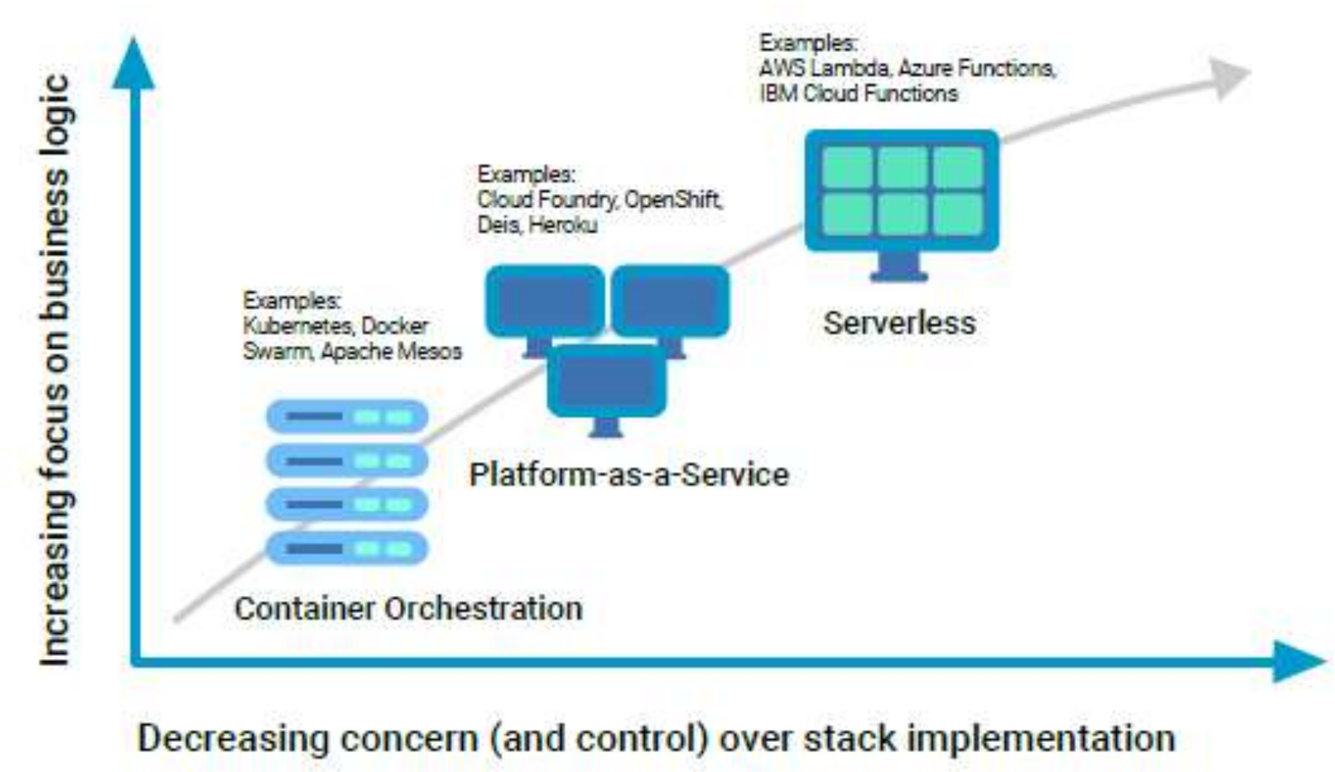
19

Years of Experience

2

Patents Filed

WHAT IS SERVERLESS?



Source: CNCF WG-Serverless Whitepaper 1.0

KEY CHARACTERISTICS

Zero Server Management

Event Driven

Autoscaling

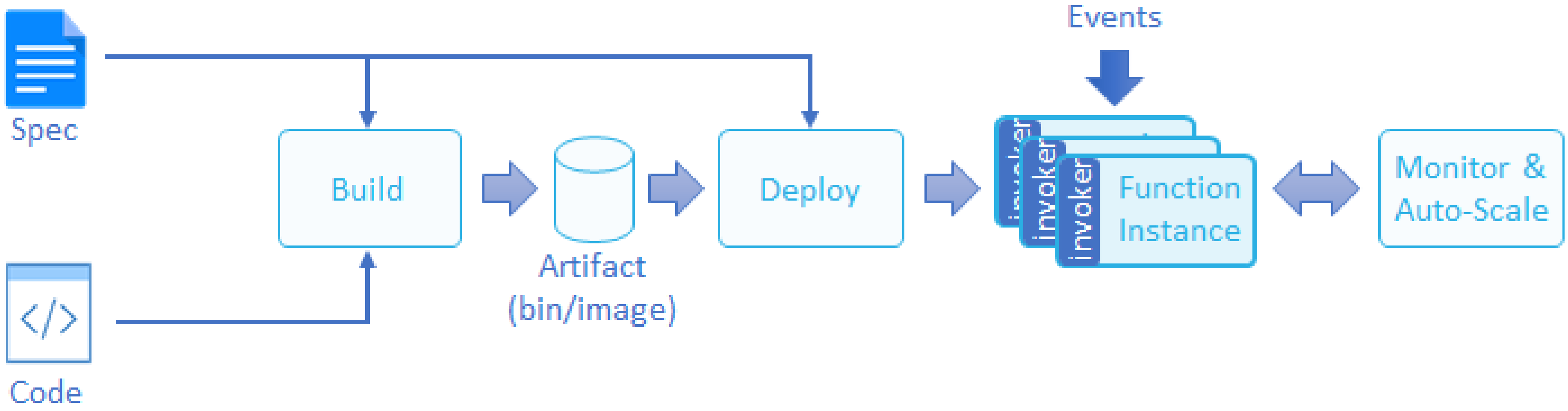
Pay per usage

Cold Start

WHAT'S NOT SERVERLESS?

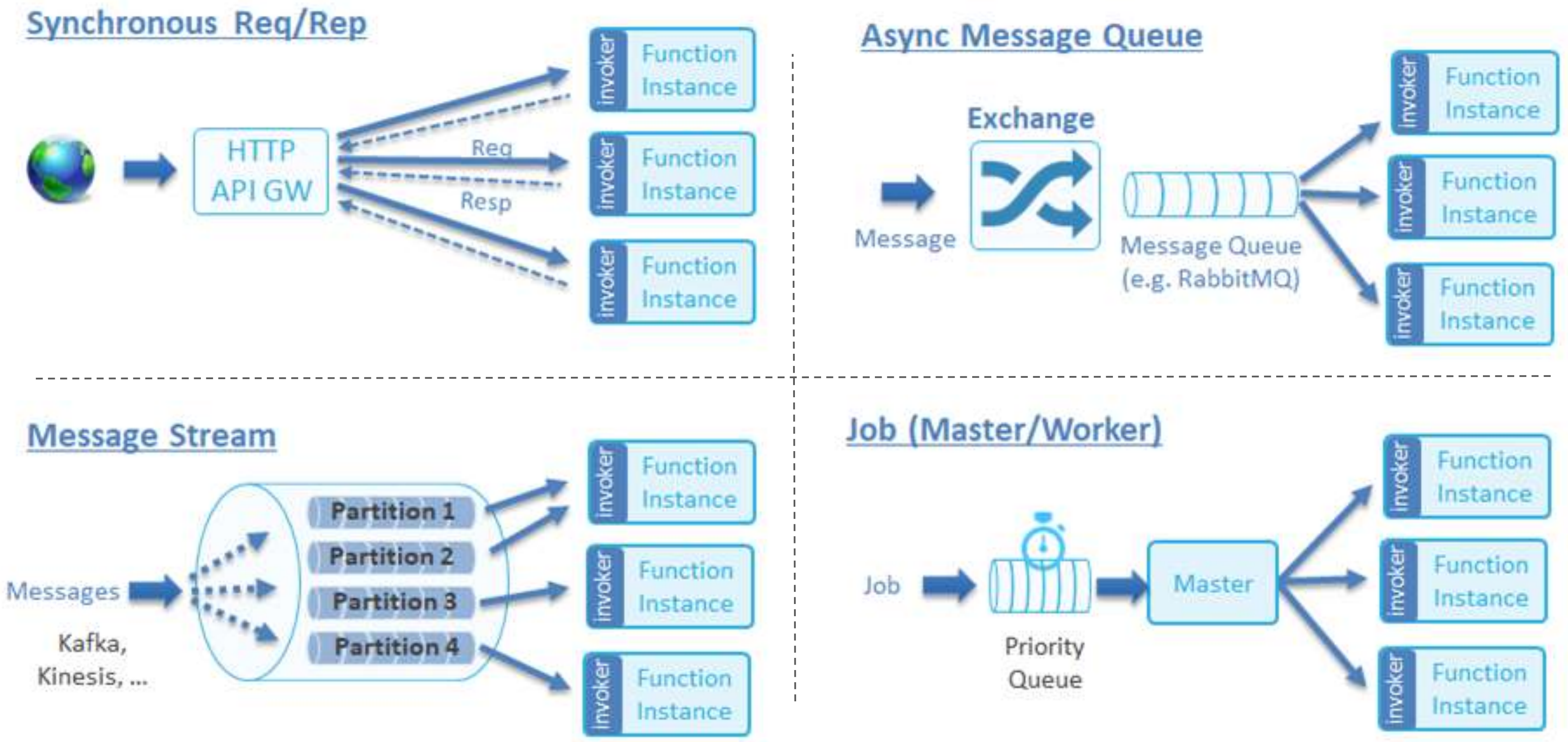
- **IaaS** in Cloud & **Self hosted/managed** datacenter
- **PaaS** with limited or no auto scaling capabilities
- **Container orchestration** platform where still platform management is still needed
- **Cloud services** such as Blob storage, Database as a Service (DBaaS)

FUNCTION LIFECYCLE



Source: CNCF WG-Serverless Whitepaper 1.0

FUNCTION INVOCATION TYPES



Source: CNCF WG-Serverless Whitepaper 1.0

INTERESTING FACTS

Adoption Rate in public cloud

60%

70% AWS customers

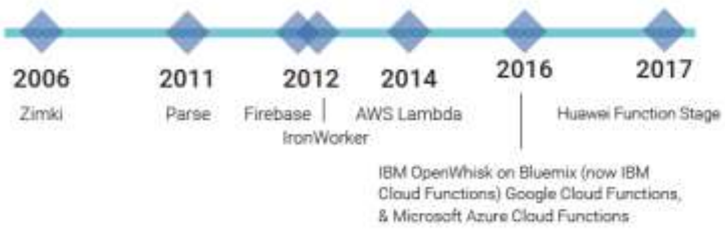
60% GCP customers

49% Azure customers

Source: datadoghq.com

1st Serverless platform traced back to 2006

Zimki


2006
Zimki
2011
Parse
2012
Firebase
IronWorker
2014
AWS Lambda
2016
IBM OpenWhisk on Bluemix (now IBM Cloud Functions) Google Cloud Functions, & Microsoft Azure Cloud Functions
2017
Huawei Function Stage

Potential cloud cost savings

50%

80% Only serverless

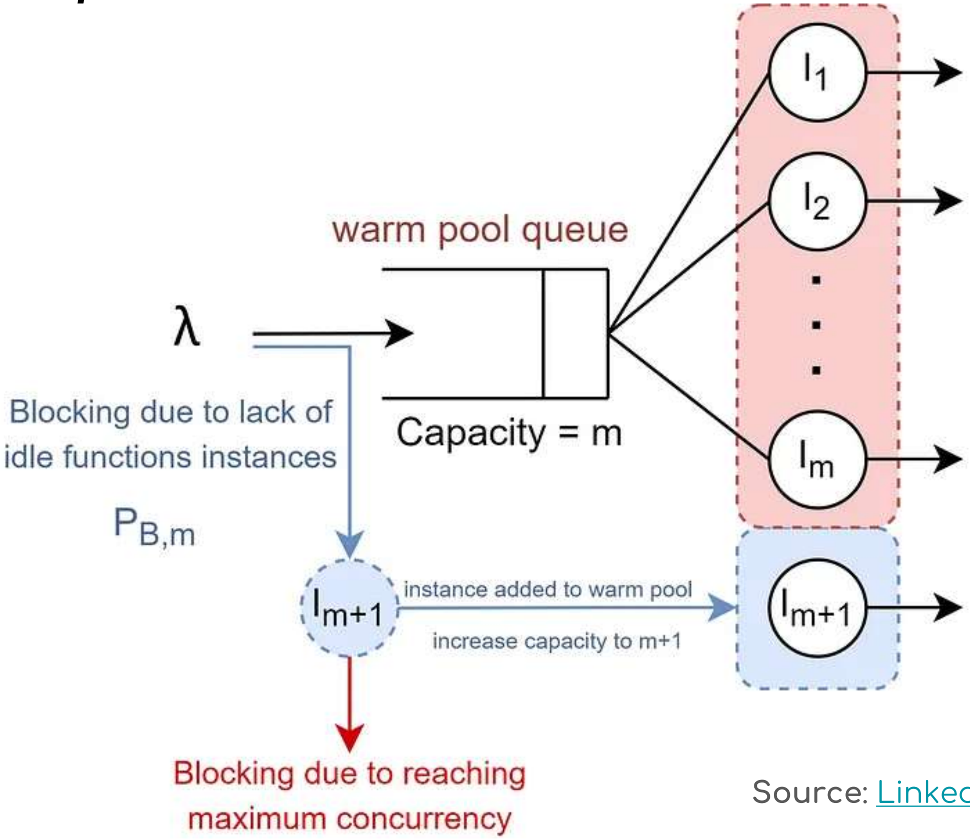
50% Multi-service architecture

20% Beginners

Source: techdogs.com

SCALING WITH SERVERLESS

Scale-Per-Request

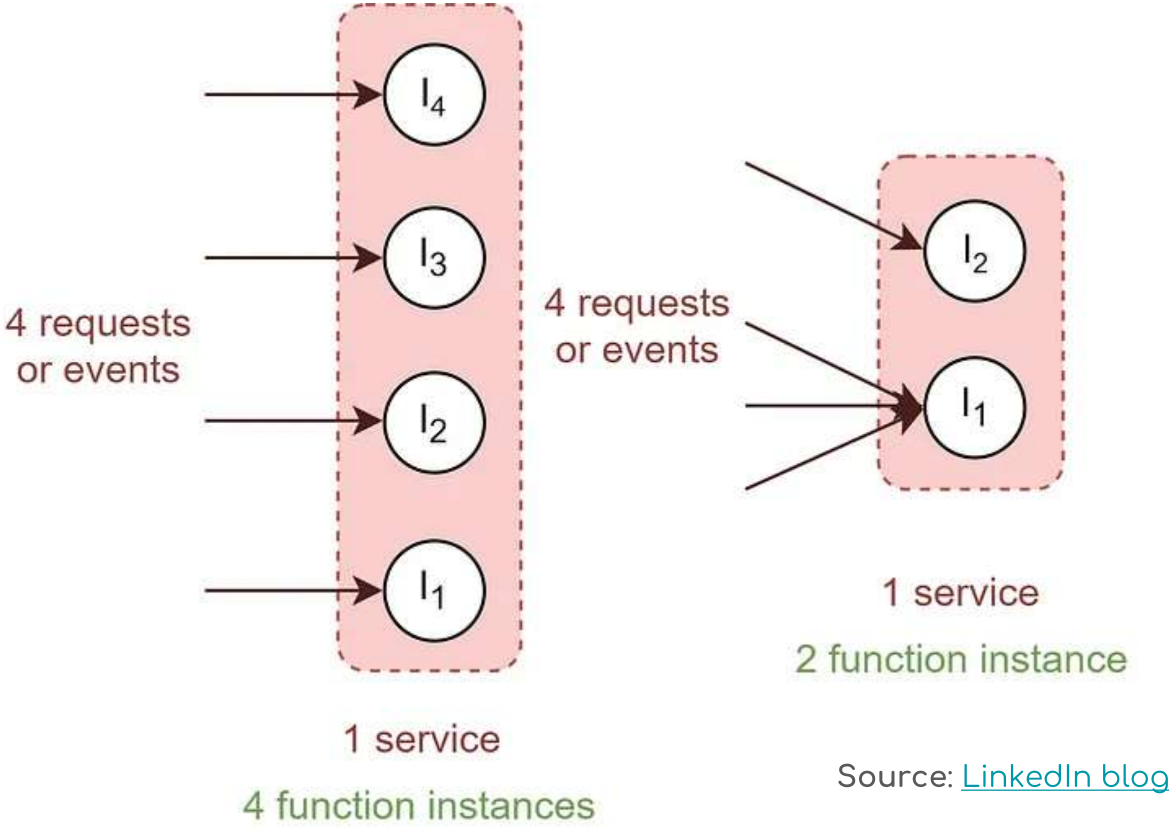


warm pool queue
λ
Capacity = m
I_1
I_2
I_m
Blocking due to lack of idle functions instances
$P_{B,m}$
I_{m+1}
instance added to warm pool
increase capacity to m+1
I_{m+1}
Blocking due to reaching maximum concurrency

Source: LinkedIn blog by Saral Saxena

SCALING WITH SERVERLESS

Concurrency Value based



Source: LinkedIn blog by Saral Saxena

SCALING WITH SERVERLESS

Metric based

Resource Utilization
(CPU, Memory, Disk)

Throughput, latency or Queue depth

Custom metric
(Viewership count, Checkouts etc.)

Schedule or Trigger

Predictive Analytics based on ML models

Best Practices for Development

- Keep it small to perform simple focused task
- Optimize the code for quick start and performant
- Robust Auth-N, Auth-Z with necessary resource privilege
- Automated CI/CD with proper validations and quality gates
- Use external storage to maintain state
- Consider batch for large workloads
- M.E.L.T with alerts for quicker issue isolation
- Cost optimization by optimize resource usage

How to overcome COLD START challenge?

Provisioned Concurrency

Pre-warming

Scheduled pre-scaling

Code Optimization

USE CASES

Real-time data processing

- Signals from IoT devices, social feed, stock market updates etc.
- Event driven pipelines

API / Microservices

- RESTFul APIs
- API Gateway or Router
- Static / Dynamic Web apps

Machine Learning

- Train ML Models on large dataset
- Deploy trained model for real-time actions

Custom Integrations

- 3rd party service integrations
- Automate workflows and tasks

ANTIPATTERNS

High resource intensive workloads

- Complex or compute intensive tasks
- Long-running processes

State management & Dependencies

- Avoid using global variables to store state
- Complex dependencies

Overlooking Security

- Vulnerable if input is not properly validated and secured

Strict Performance & Cost requirement

- High volume and low latency workload
- Legacy systems

OBSERVABILITY

Metrics

Latency, traffic, Errors, Saturation, throttling

Events

Startup, Scale-up, scale-down, deployment

Logs

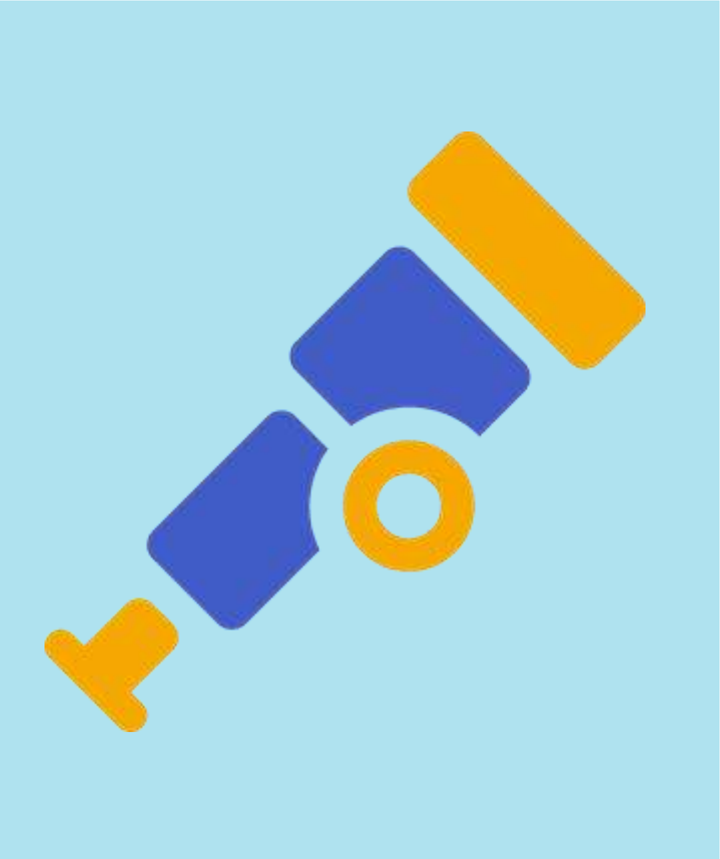
Failures, Exceptions, state transformation

Traces

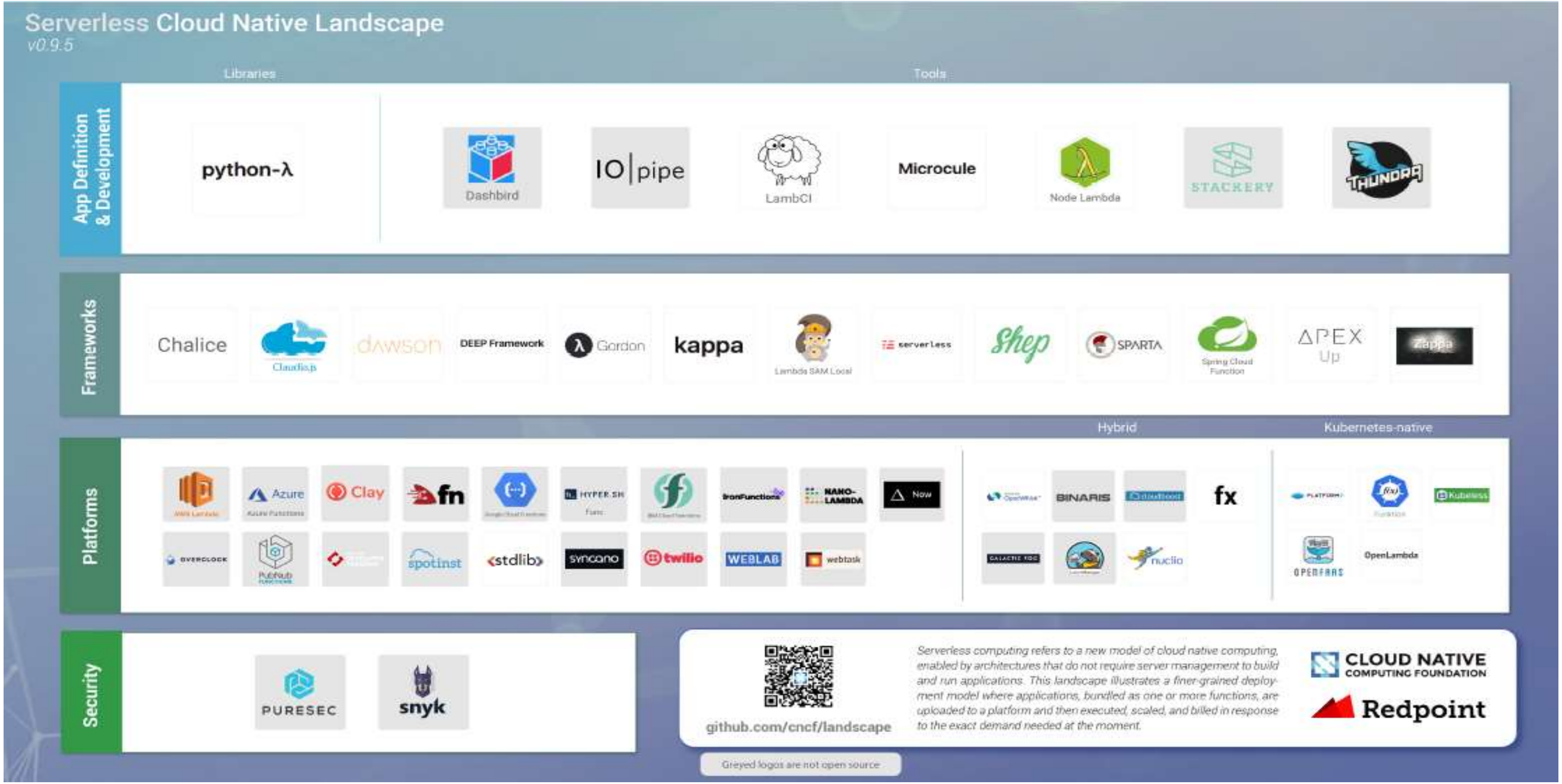
Spans, attributes

Profiling Data

CPU sampling, Memory allocations



CNCF’s Place in Serverless ecosystem



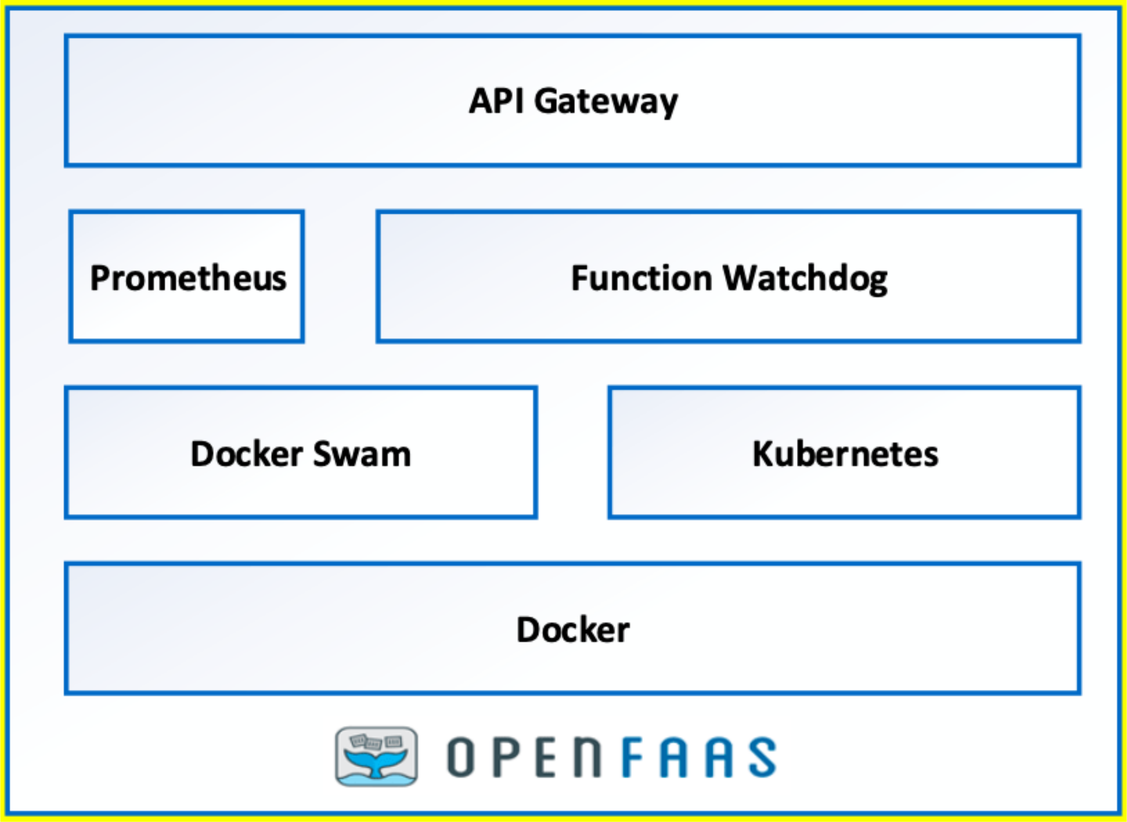
Serverless Cloud Native Landscape

v0.9.5

Serverless computing refers to a new model of cloud native computing, enabled by architectures that do not require server management to build and run applications. This landscape illustrates a finer-grained deployment model where applications, bundled as one or more functions, are uploaded to a platform and then executed, scaled, and billed in response to the exact demand needed at the moment.

Greyed logos are not open source

OpenFaaS


API Gateway
Prometheus
Function Watchdog
Docker Swam
Kubernetes
Docker
OPENFAAS

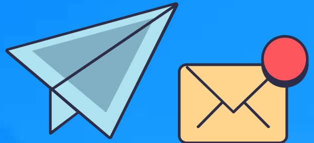
Demo

CONCLUSION

- **Flexibility and customization** in building scalable and event-driven applications.
- **Cost-effectiveness** through pay-per-use-model and eliminates infrastructure management overhead
- **Scalability**
- Broader **community support & vendor independence** on using open source CNCF frameworks.
- **Growing opportunity & maturity**
- Integration with **edge computing**
- **Hybrid** or **multi-cloud** support
- Improved **developer experience**

Thanks!

Want to connect?



kbsivacse@live.com