National Technical University of Athens
School of Electrical & Computer Engineering
Embedded Systems Design - MicroLab

SERVERLESS ON-EDGE

A Kubernetes deployment evalution

Project by: Tzomaka Aphrodite

TABLE OF CONTENTS

01
INTRODUCTION

A brief overview of the technologies we used

DOES SERVERLESS MEAN NO SERVERS?

WHAT SERVERLESS MEANS IS:

- No need for server/container/OS management

Focus on app-building ➔ **FASTER TIME-TO-MARKET!**

- Auto – scaling
- High availability – Fault tolerance
- No idle capacity (**Pay only what you use!**)

EDGE COMPUTING

Placing workload as close to where data is being created as possible

MANAGEMENT

Distribute workloads at massive scale without needing individual administrators

Different utilities, operating systems and architectures

DIVERSITY

SECURITY

Vulnerability in transferring sensitive data towards the cloud

EDGE
IN TERMS OF NUMBERS

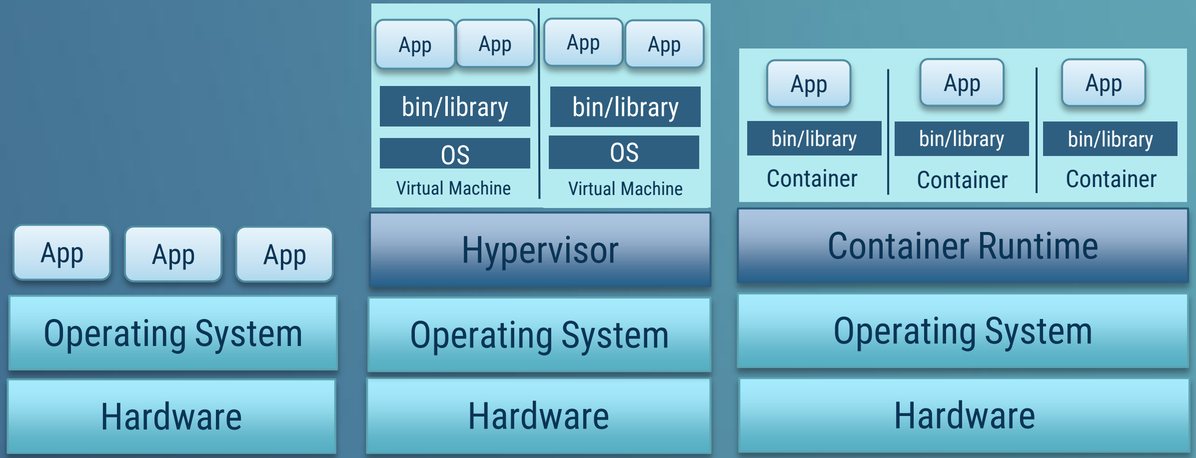
150billion — Edge devices by **2025**

55billion — Edge devices by **2022**

15billion — Edge devices in **2019**s marketplace

CONTAINERIZATION

- Resource utilization: High efficiency & density
- Apps deployed & managed dynamically due to smaller size & modularity
- Easier creation of containers images than VMs images
- Cloud & OS distribution portability



KUBERNETES

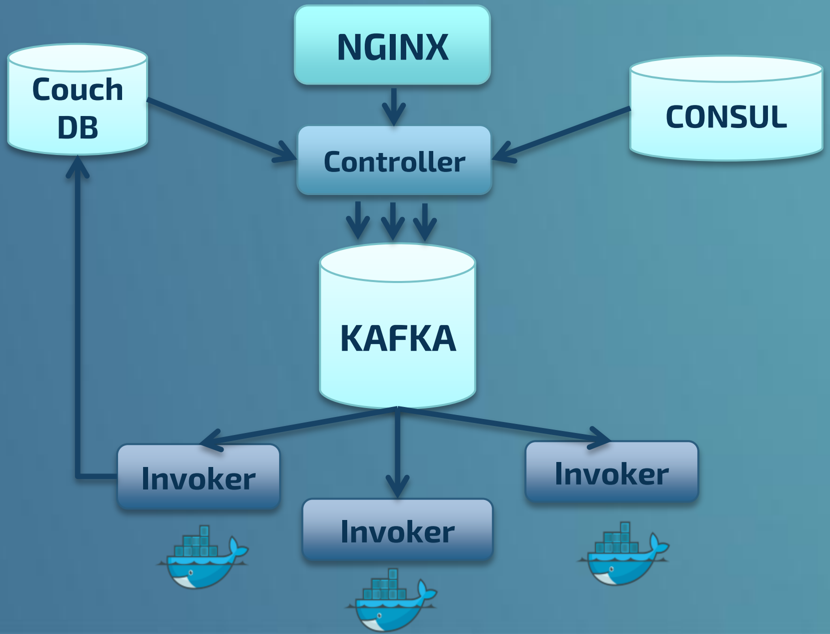
Orchestration tool that allows running & managing container-based workloads

- Service discovering & load balancing
- Storage orchestration
- Automated rollouts & rollbacks
- Self – healing
- Secret and configuration management

- We used a lightweight version of this tool, k3s.

OPENWHISK

- Open source, distributed serverless platform.
- Executes functions in response to events at any scale.
- Manages the infrastructure, servers and scaling using Docker containers.



02

OUR CLUSTER

Binding all those stuff together

THE TOPOLOGY

Master Node

- Ubuntu VM/Intel x86-64/4CPUs/8GB RAM
- Manages the worker nodes & the pods in the cluster.
- Makes global decisions.
- Detects/responds to cluster events.

Worker Node

- Raspberry Pi 4/Arm64
- Hosts the pods.
- Represents the edge-node.

SETTING UP

- Installation of Helm 3, a tool to simplify the deployment & management of apps like OpenWhisk on our k3s (lightweight Kubernetes) cluster
- Label both nodes as invokers (via kubectl)
- Creation of mycluster.yaml file to record key aspects of our k3s cluster to configure OpenWhisk deployment
- Installation/Upgrade of our deployment via Helm CLI (helm install / helm upgrade)
- Use of docker images that correspond to the nodes architecture – multi-arch build

03

METRICS

How we chose to evaluate our cluster

METRICS

STARTUP LATENCY

On-demand initiation & high concurrency lead to a startup latency that is hard to reduce.

COMMUNICATION PERFORMANCE

Examine the interaction of functions that compose an application and other cloud services.

ServerlessBench

IBM owperf

STATELESS EXECUTION

Data transmission overhead if we need to maintain states. Loss of info that could benefit us.

PARALLELIZATION LIMITS

What is the limit of concurrency that we can exploit so as to keep throughput high?

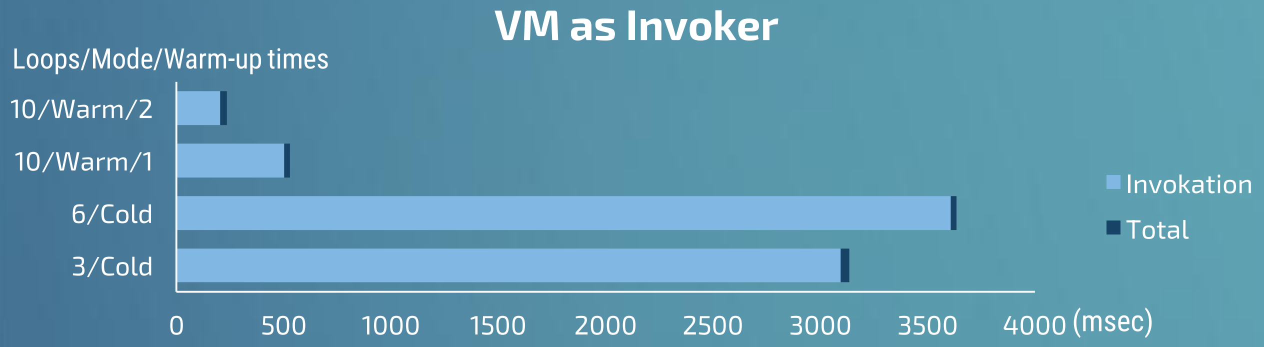
04

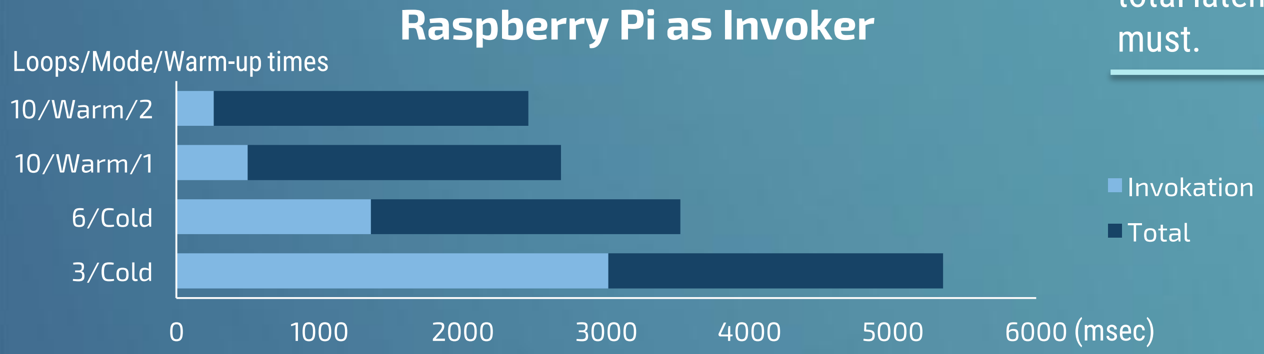
TESTCASES RESULTS

Benchmarking our cluster & making some implications

STARTUP LATENCY TESTCASE:

STARTUP BREAKDOWN – A 'Hello' Java application to measure startup & execution latency (cold/warm start)





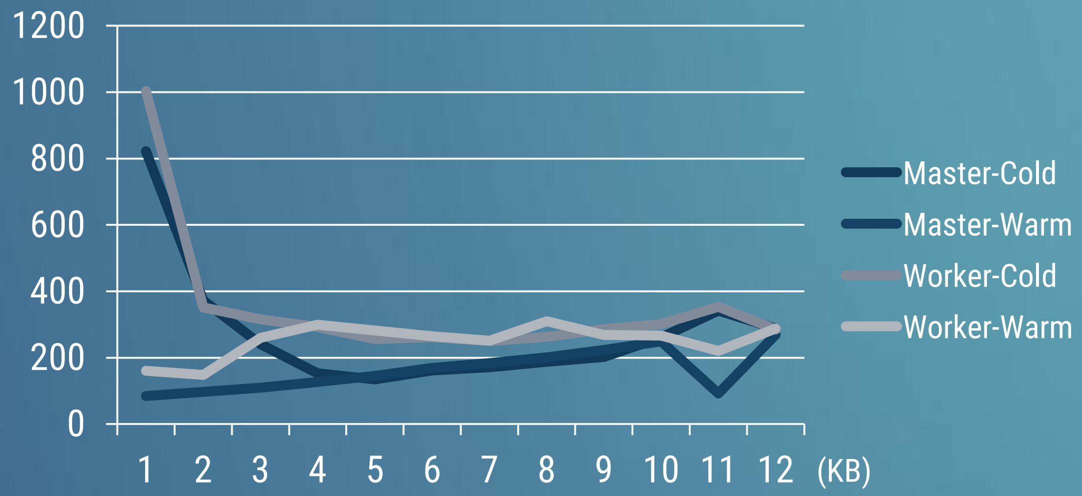
Startup overhead holds a major percentage of the total latency making methods like prewarming a must.

COMMUNICATION PERFORMANCE TESTCASE:

DATA TRANSFER COSTS - A node.js serverless application which transfers images with different sizes

The latency is neglectably affected by passing small amount of data between subsequent functions.

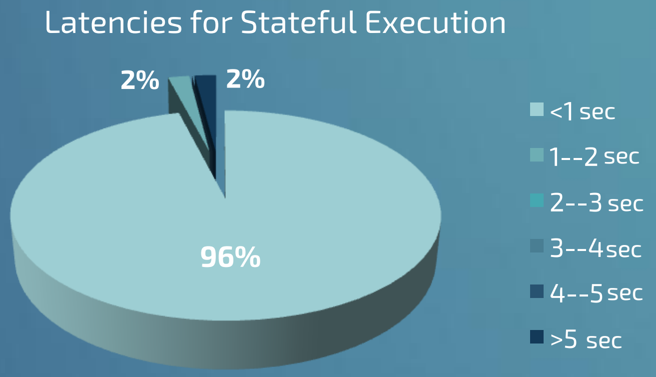
Latency between subsequent functions

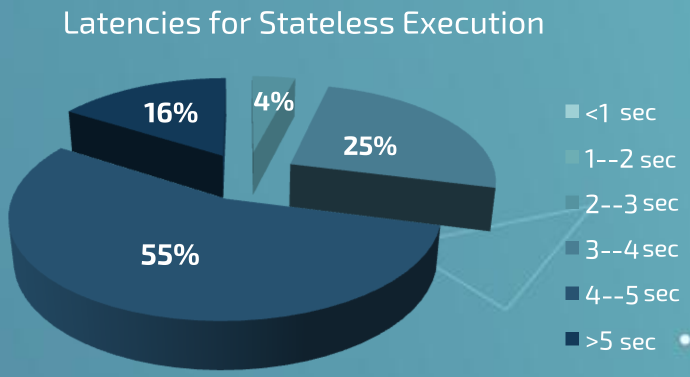


STATELESS EXECUTION TESTCASE:

STATELESS COSTS - A Complex Java application to examine the impact of implicit states transmission

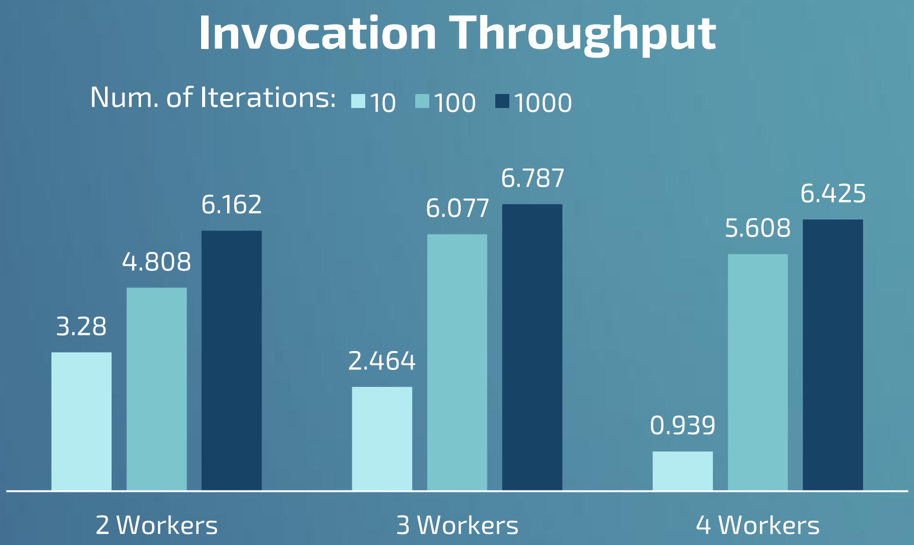
The cost of sharing the implicit states between functions is minimum compared to the execution speedup that they provide.





PARALLELIZATION LIMITS:

IBM OWPERF – *A benchmarking tool for warm latency/throughput measurements*



High concurrency leads to lower throughput due to the complex components/building blocks OpenWhisk uses (CouchDB, Kafka, Zookeeper, etc)

05

REFERENCES & ACKNOWLEDGEMENTS

Not all heroes wear capes :)

REFERENCES

- https://github.com/SJTU-IPADS/ServerlessBench
- https://github.com/IBM/owperf
- https://kubernetes.io
- https://openwhisk.apache.org/
- https://serverlessbench.systems/socc20-serverlessbench.pdf

ACKNOWLEDGMENTS

- Achilleas Tzenetopoulos, Dimosthenis Masouros (Supervisors of this project - Junior Researchers @ MicroLab, NTUA) for their helpful guidance.
- Giannos Gavrielides, Christos Papakostopoulos, Konstantinos Stavrakakis (Undergraduate students @ ECE, NTUA who also worked on similar projects) for the perfect cooperation.
- Aggeliki, Nikodimos, Panagiotis for being my supportive audience at my rehersals (yes, I practiced a lot to make it with presenting! ☺)

National Technical University of Athens
School of Electrical & Computer Engineering
Embedded Systems Design - MicroLab

THANK YOU!

Questions?