UNIVERSITY OF TARTU

Serverless

04.06.2020

Sander Kuusemets

Table of contents

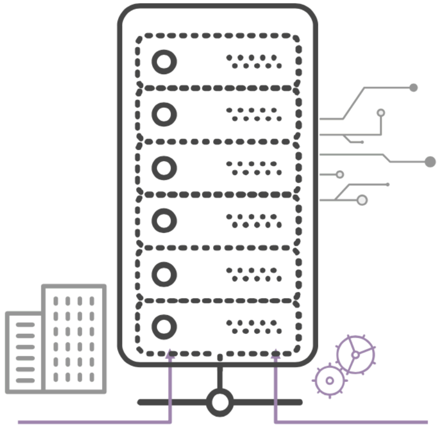
- Introduction to serverless
- Architecture
- Positives
- OpenFaaS
- Kubeless
- My opinion + negatives of serverless

History of DevOps

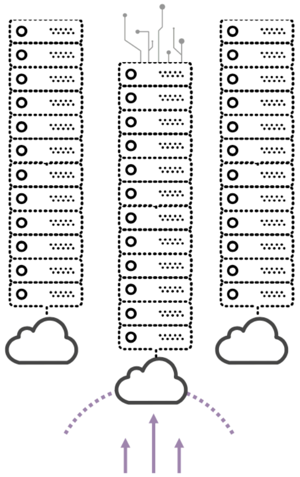
Until not so long ago:

Physical servers:

Virtual servers:



Virtual servers in the cloud:



+ Every step was better, achieving higher density and capability.

+ Virtual machines allowed disaster recovery.

+ Things moved towards hardware independence.

- Yet there is a high overhead for virtualization, compared to other possibilities.

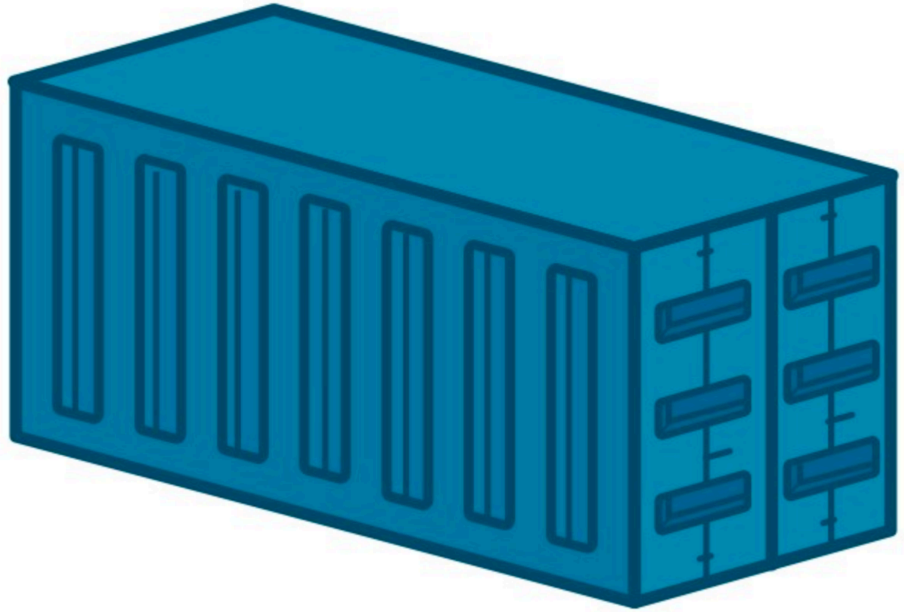
- But it still took time to migrate and restart VMs.

- But not operating system independence.

Then came containers

+ Had many positives, like immutable infrastructure and much less overhead.

- The tooling for management was not there yet, containers were built and managed like VMs.
- Had to worry about base images and packages, introducing overhead and security problems.



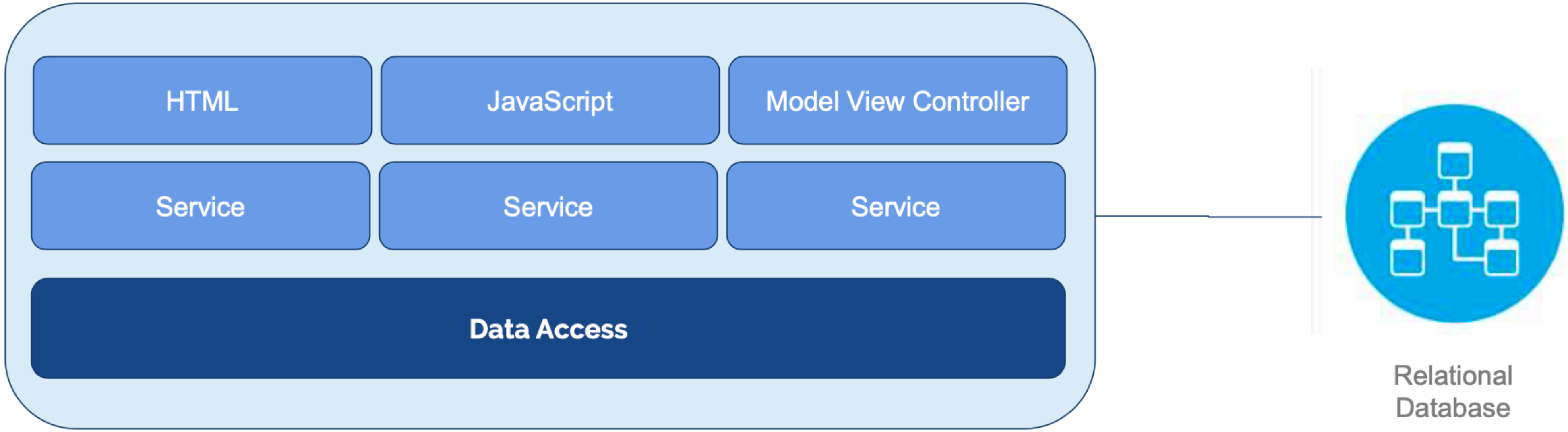
Next logical step.. Skip containers!

- .. well not actually true.
- Serverless still packages functions, the base unit of serverless, inside containers.
- The developer just does not have to worry about building or deploying them.

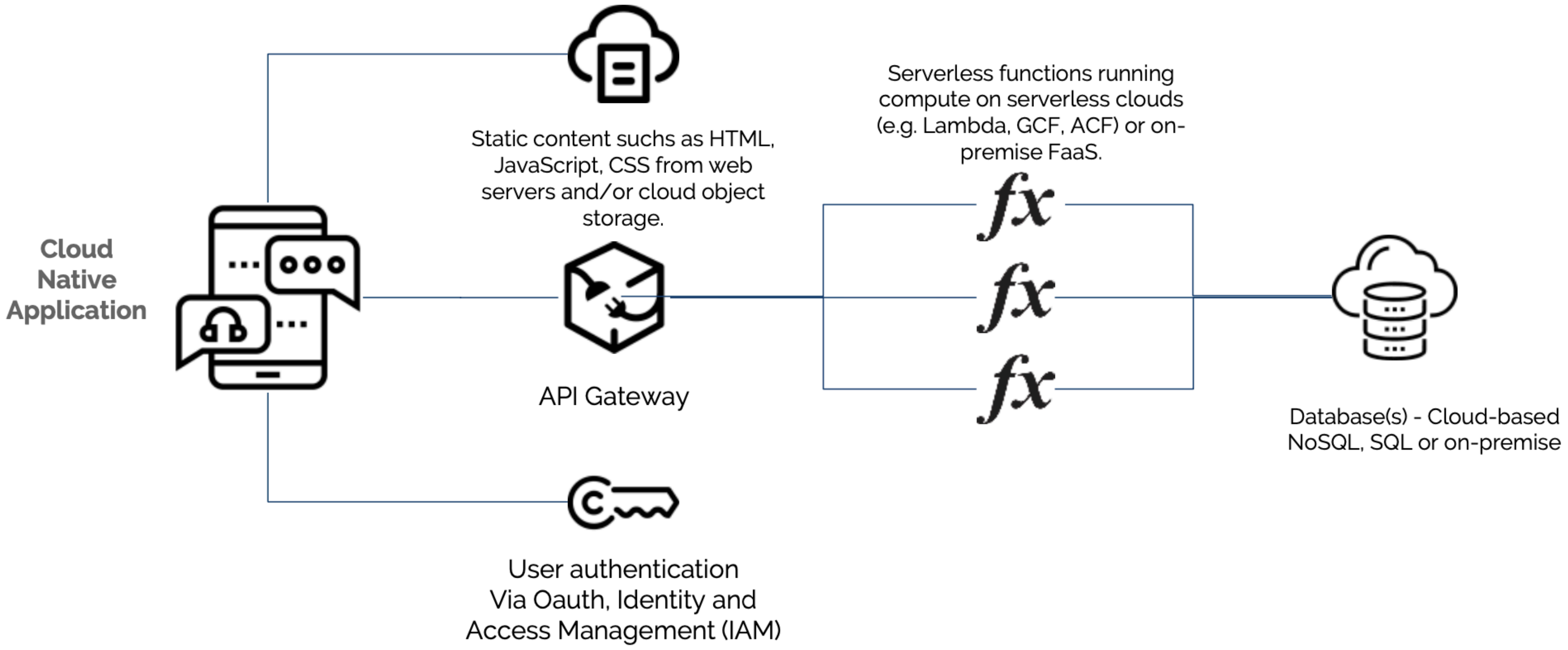
So.. What is serverless?

- Uses functions instead of containers. You write a function, you deploy it, you run it.
- Provides a level of abstraction for hardware, so developers do not have to think about it.
- Forces a stateless, cloud based microservices paradigm.
- Event driven.
- Can scale up to thousands of replicas, or down to zero, avoiding costs.

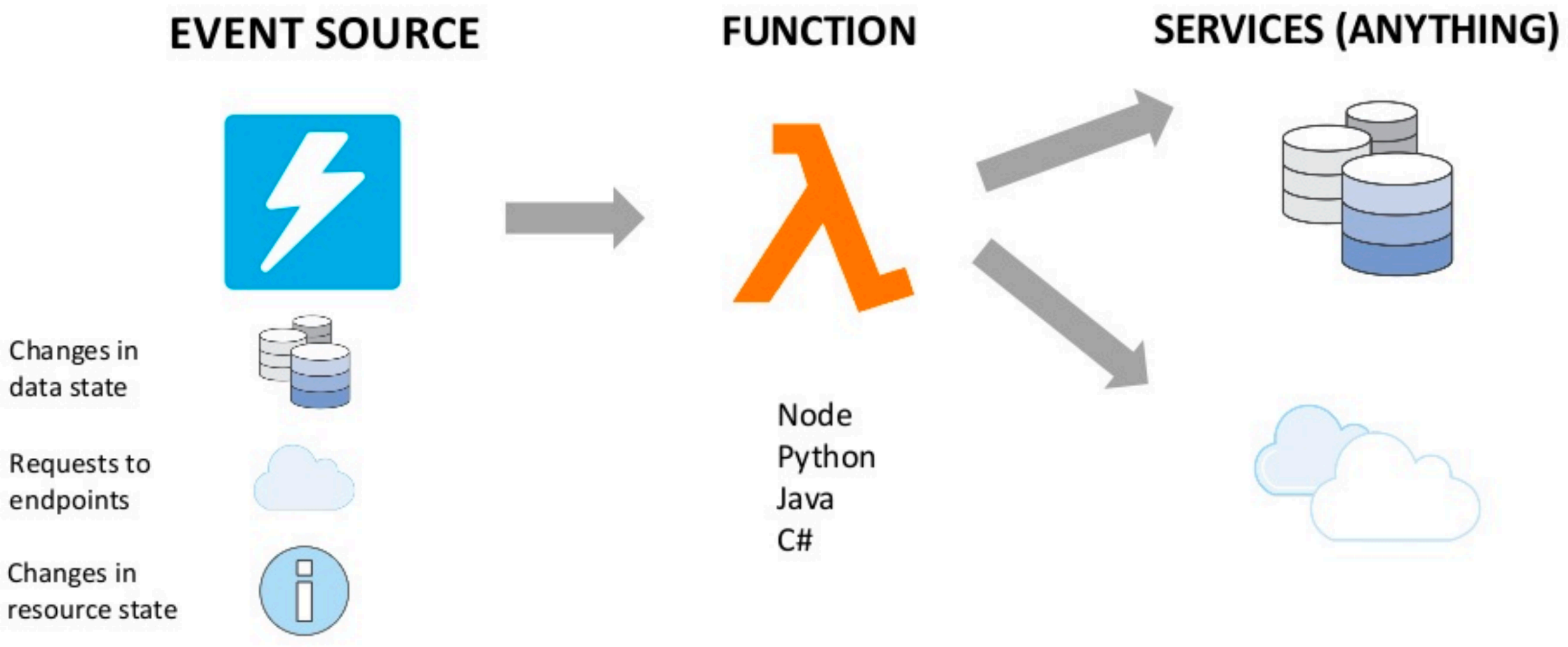
Monolithic architecture



Cloud based architecture

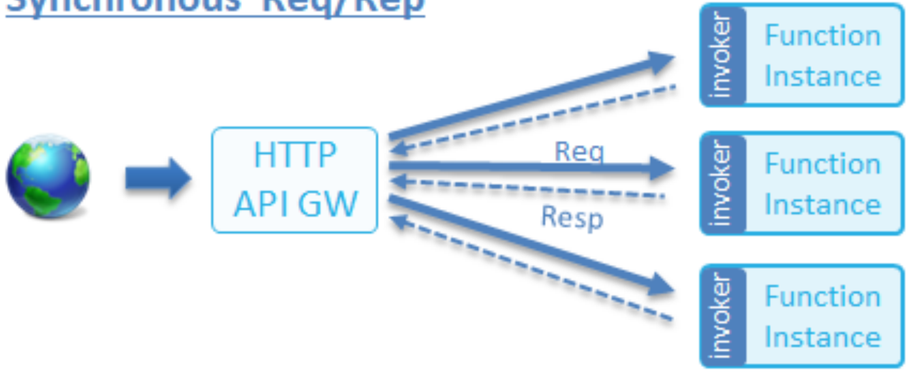

Cloud Native Application
Static content suchs as HTML, JavaScript, CSS from web servers and/or cloud object storage.
API Gateway
User authentication Via Oauth, Identity and Access Management (IAM)
Serverless functions running compute on serverless clouds (e.g. Lambda, GCF, ACF) or on-premise FaaS.
fx
fx
fx
Database(s) - Cloud-based NoSQL, SQL or on-premise

Serverless basic architecture

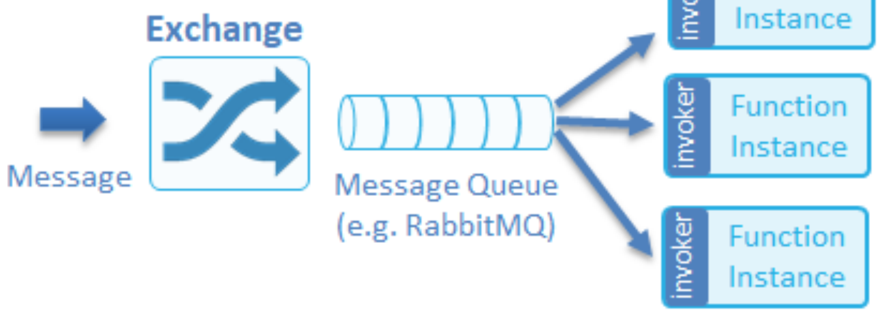


Serverless function triggers

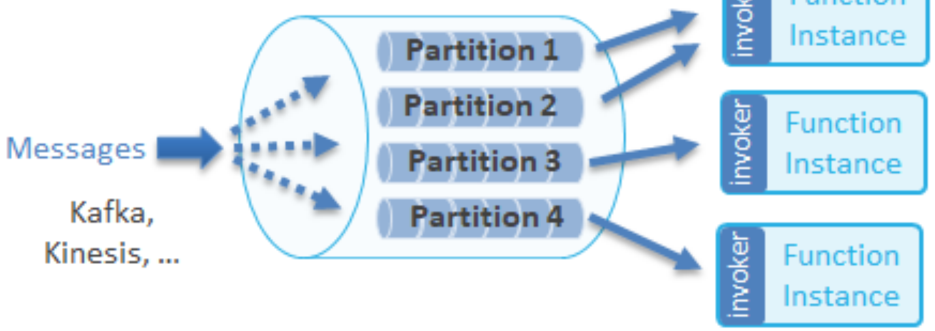
Synchronous Req/Rep



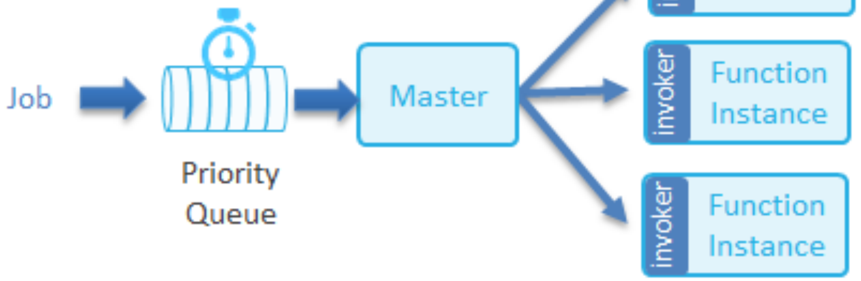
Async Message Queue



Message Stream

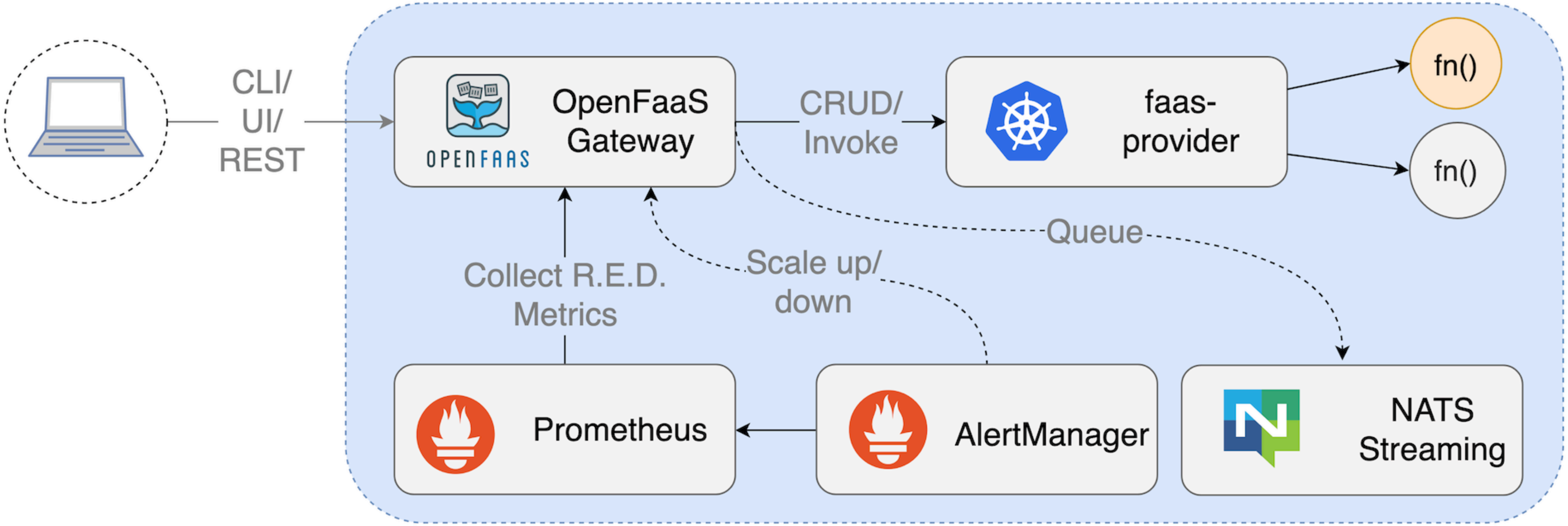


Job (Master/Worker)



Any questions so far?

OpenFaaS (Open source Function-as-a-Service)



OpenFaaS highlights (from their home page)

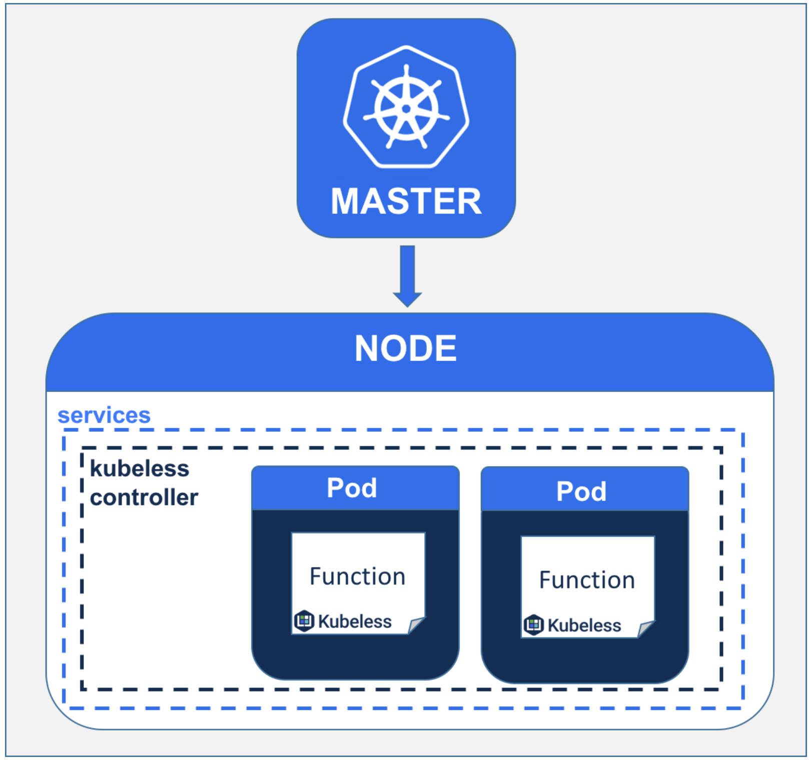
- Ease of use through UI portal and *one-click* install
- Write functions in any language for Linux or Windows and package in Docker/OCI image format
- Portable - runs on existing hardware or public/private cloud - Kubernetes and Docker Swarm native
- CLI available with YAML format for templating and defining functions
- Auto-scales as demand increases

Demo

- Placeholder

Kubeless

- Does things a bit easier.
- Built ground up to integrate with Kubernetes.
- Follows AWS, GCS, Azure etc best practices.
- Has fewer components.
- Might be a tiny bit raw in places.



My opinion on serverless, and it’s negatives

- Makes things extremely straightforward for developers.
- .. not so much for DevOps/SysOps/SRE people.
- Still need to manage deployments, but instead of VMs or containers, now it’s functions.
- Requires a very-very-very good monitoring and alerting infrastructure.
- Allows for use of development best practices for deployments – testing, rolling releases, versioning, backoff-on-failure etc.

UNIVERSITY OF TARTU