

Past & Future of Containerization

Max Körbächer | Co-Founder & Associate Partner @ Liquid Reply

Introduction

Max Körbächer

Co-Founder & Associate Partner, focusing on:

- Platform Engineering
- Application Delivery
- Cloud Native Advisory

CNCF TAG Environmental Sustainability

Kubernetes Release Team from v1.17-1.24

Advisory Board keptn & DevNetwork



Docker has changed the game

Docker/Container changed:

- the way we **design** and **build** applications
- caused a whole **ecosystem** with hundreds of open source systems to appear
- drive adoption from all kind of **cloud provider**
- changed the way we do **automation**
- pushed the development of an **OCI standard**

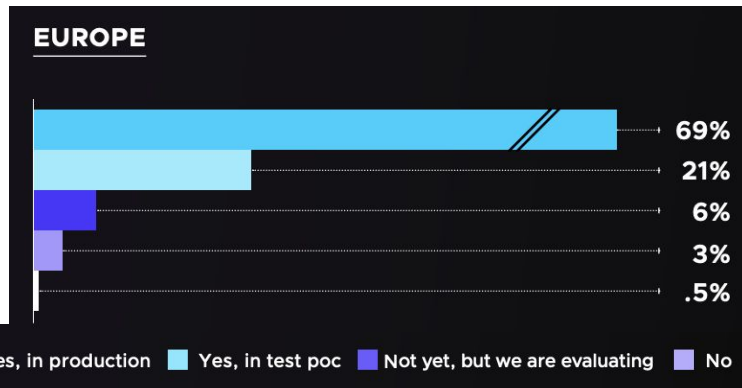
Corporations using or planning to use container

93%



In Containers we trust in Kubernetes we build

- Kubernetes leverages container and got a **defacto standard** for container orchestration (also there are many other nice implementations)
- Kubernetes gets implemented and **used everywhere** (cloud, IaaS, on metal, on edge) - it simplifies a lot, but it also raises the complexity



- In Europe **>= 90%** organizations working with or on K8s - this is comparable to the usage of hypervisor (92%)



Container & Kubernetes

Both together has changed and influenced the ICT world massively

A big bang for a total new market

Boosting open source and a community driven development to new levels

Changed the way we see infrastructure -> Infra as Apps

Security, observability, any kind of extension is seen as a simple plug & play

K8s abstracts away hypervisors, CSP and IaaS

K8s create a knowledge voidness



What next?






Solomon Hykes @solomonstre · 27. März 2019



If WASM+WASI existed in 2008, we wouldn't have needed to created Docker. That's how important it is. Webassembly on the server is the future of computing. A standardized system interface was the missing link. Let's hope **WASI** is up to the task!



Lin Clark  @linclark · 27. März 2019

WebAssembly running outside the web has a huge future. And that future gets one giant leap closer today with...



Announcing WASI: A system interface for running WebAssembly outside the web (and inside it too)

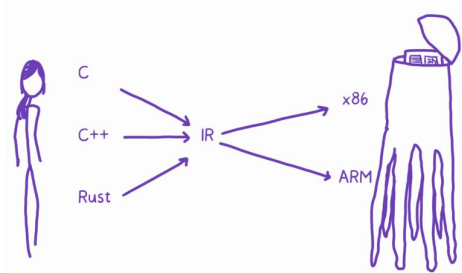


What is WebAssembly (WASM)?

Intermediate Layer

Various programming languages and many different execution environments

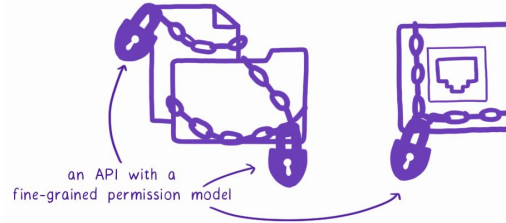
CPU & OS agnostic



Secure

Per default a WASM component is allowed to do nothing

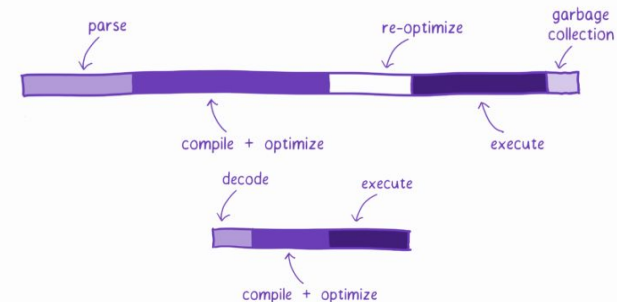
Encapsulated binary, no OS within, nothing to “hack”



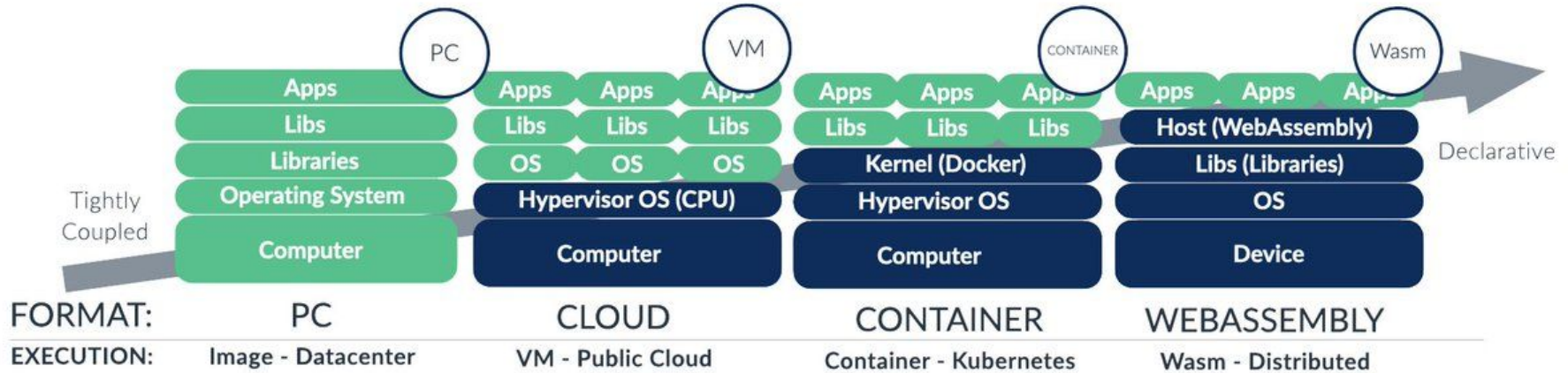
Fast(er)

Drastically short startup time (x100 faster than a container)

Micro footprint, measured in MB not GB



A new paradigm (?)



Where can WebAssembly be applied?

***outside the Browser**



Language Interoperability

Write that library once in a language of your choice; use in any language.

Figma
Lichess.org
Google Earth
Adobe Photoshop



Plugin Systems

Never trust third parties!

Envoy / Istio
Kubewarden
MS Flight Simulator
Minecraft
RedPanda



Embedded Sandboxing

Prevent yourself against bugs of third party libraries.

Firefox
HTTP Servers



Blockchains

Write Smart Contracts in a language of your choice.

CosmWasm
eWASM



Containerisation

Universal Runtime, capability based security model.

Krustlet
Hippo
wasmCloud
Lunatic
WasmEdge



Serverless Platforms

Minimal Startup time, maximal isolation.

Cloudflare Workers
AWS Lambda
Atmo (Suborbital)
Fastly Compute@Edge



Example implementations



KRUSTLET

Kubernetes

Kubernetes API
Server

Node

Krustlet

replace

Wasmtime

require

wasi

WASM Module

require

A Krustlet Kubernetes Stack



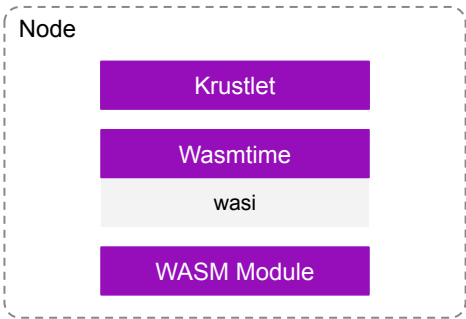
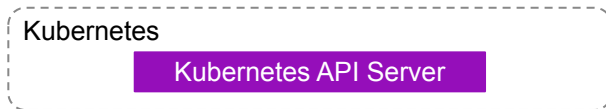
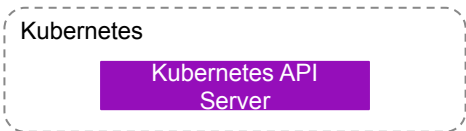
Example implementations



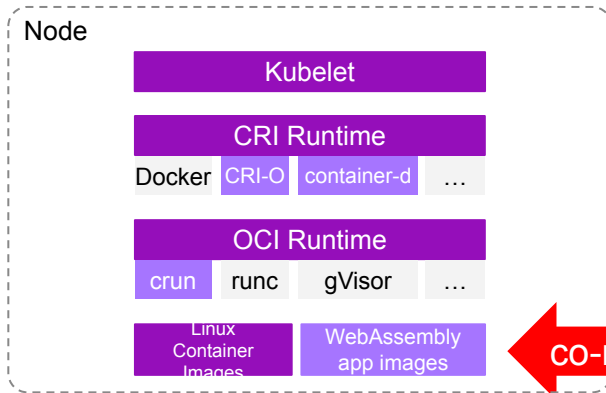
KRUSTLET



WasmEdgeRuntime



A Krustlet Kubernetes Stack



The Container Eco-System



Example implementations



KRUSTLET

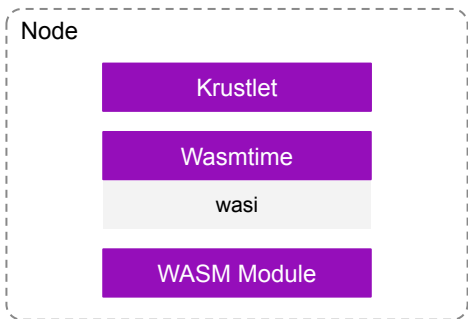
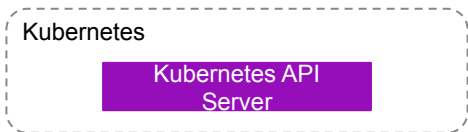


WasmEdgeRuntime

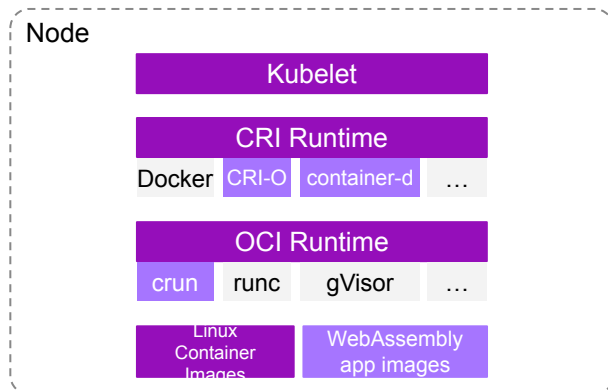
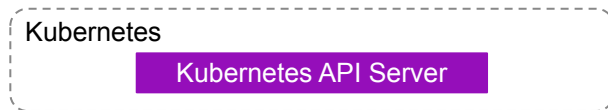


wasmcloud

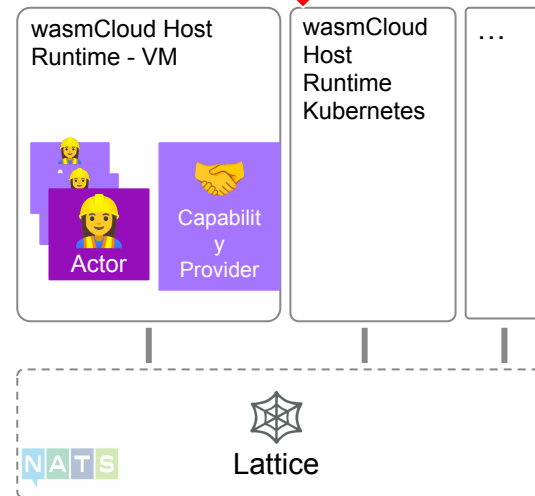
new platform



A Krustlet Kubernetes Stack



The Container Eco-System



Example implementations



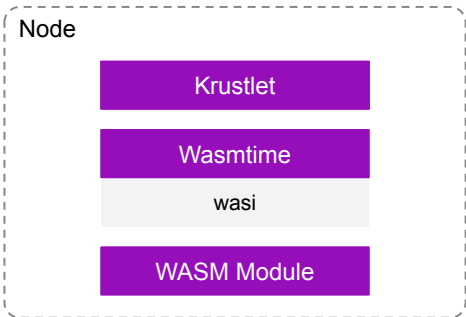
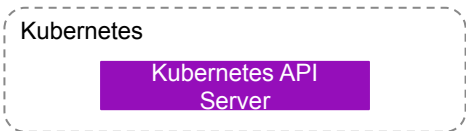
KRUSTLET



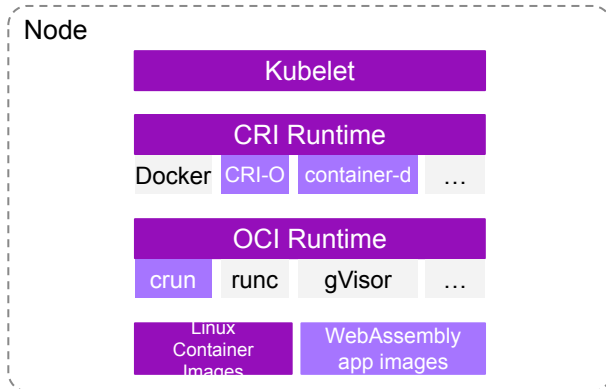
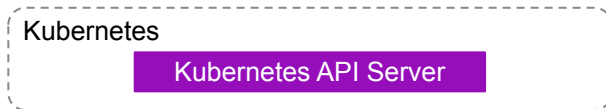
WasmEdgeRuntime



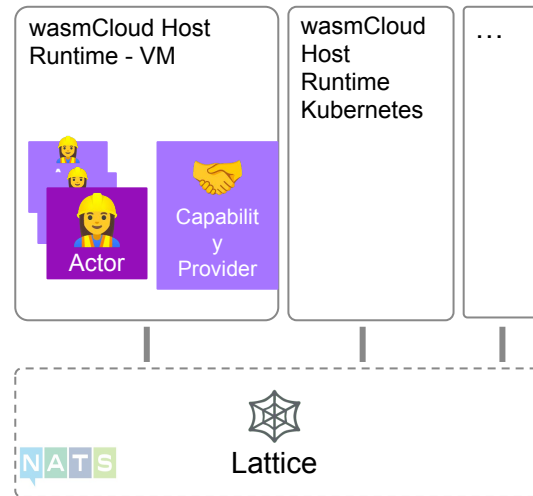
wasmcloud



A Krustlet Kubernetes Stack

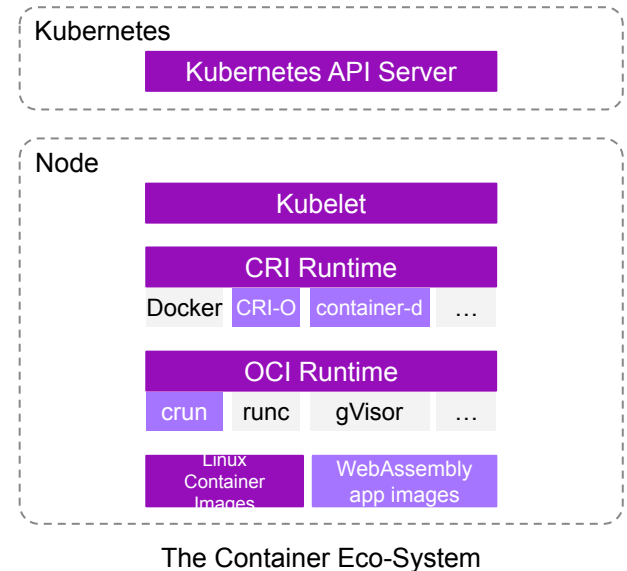


The Container Eco-System



Let's think about the WASM potentials based on WasmEdge

- Especially targets the integration in **various Kubernetes distributions, CRI runtimes** as well as **OCI runtimes** - therefore a good match to run WASM side by side with classic containers
- Runs also stand alone for **modern web apps**, to host **serverless functions** and being “embedded” in any kind of **edge device**.
- It leverages all **advantages of WASM** and bring it into a strong ecosystem **without being invasive**



WASM can extend Container

	Docker-like container	WebAssembly
Performance	OK	Great
Resource footprint	Poor	Great
Isolation	OK	Great
Safety	OK	Great
Portability	OK	Great
Security	OK	Great
Language and framework choice	Great	OK (yet)
Ease of use	Great	OK (yet)
Manageability	Great	Great



An aerial photograph of a city, showing a complex network of highways and roads. The roads are filled with cars, and there are green trees interspersed among the buildings. The overall scene is a dense urban environment.

1

WebAssembly's potential is beyond the browser

3

WASM will not substitute containers & K8s, but extend it

5

The developer experience of/for WASM will be the game changer

2

WASM enables use cases that are not possible with container & K8s

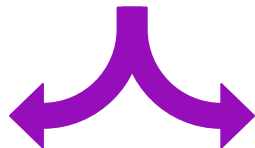
4

WASM lacks harmonization and makes it difficult for programming languages to adapt



Containers for lifting, WASM for re-creating

Go with the Container flow



Build with WASM for the future

Containers will stay and drastically increase in usage over the next years.

But for future developments WASM might be in many cases a better choice.

 Simple to use

 Big eco-system

 Consistently fast

 Small

 Language support


 “1st born” effect

 Universal

 Reusable

**We believe that WASM & Container will go
along side by side**





**WASM will be
ubiquitous**



