WASM Day 2023 Recap

- Video Playlist
- Event Schedule

fujitatomoya@github / tomoyafujita@linkedin

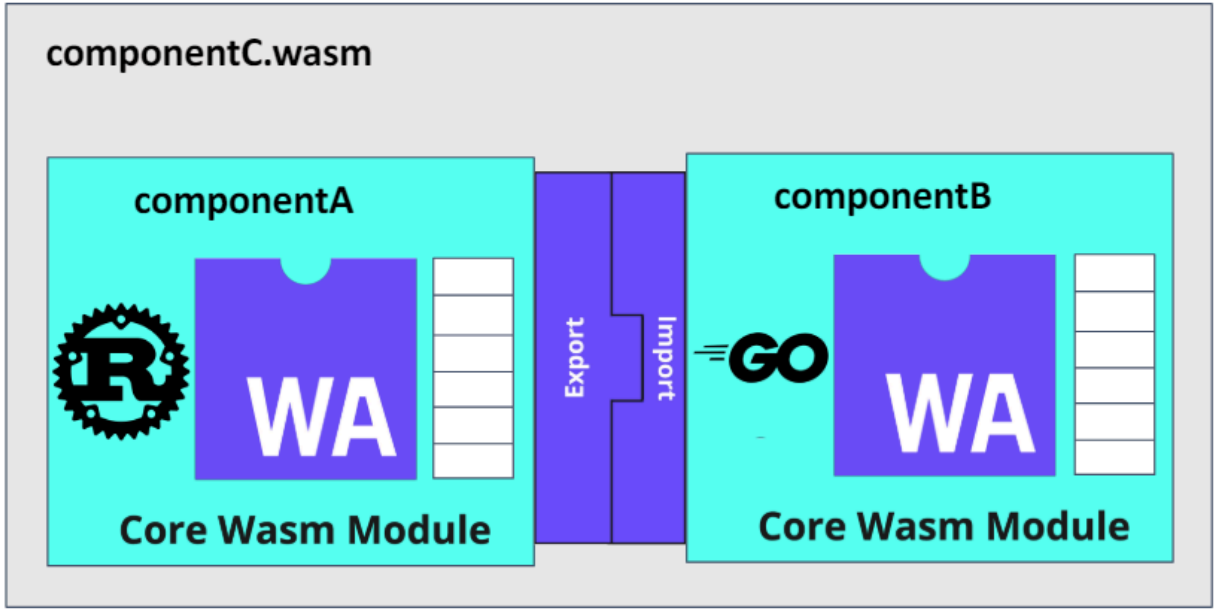
WebAssembly Component Model: Enhancing Security, Productivity, and Green Computing

WebAssembly component model for

- Isolated, memory containment.
- Mitigatable, fix and replace where it needs to be.
- Portable, language agnostic.

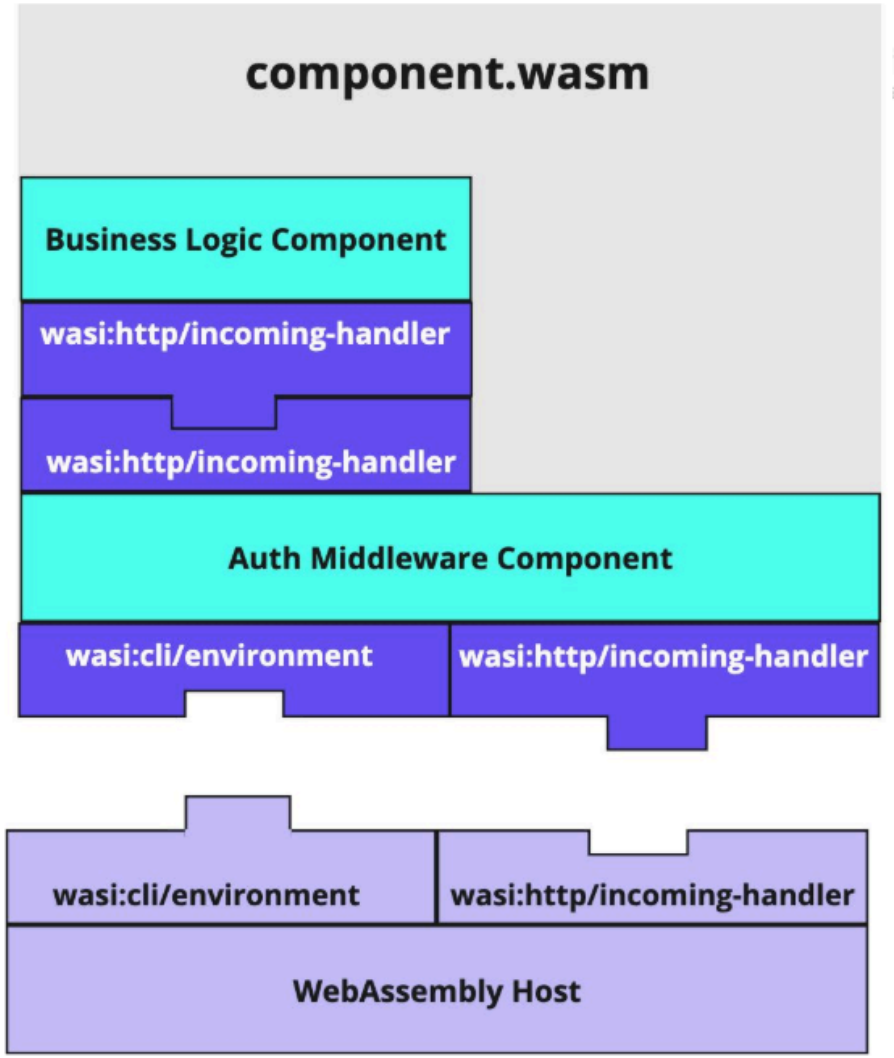
WASM Component Model

is a broad-reaching architecture for building interoperable Wasm libraries, applications, and environments.



WASM Component Model Example

- Isolate responsibility and functionality.
- Only replace where it needs to be.
- Focus on application business logics.
- Language-free Modularity.



Running Linux-Based Containers on Wasm and Browser with Container2wasm Converter

- Convert Linux containers to WASM with CPU emulators.
- Leveraging existing apps on browser.
- Leveraging Wasm features for existing applications.

container2wasm

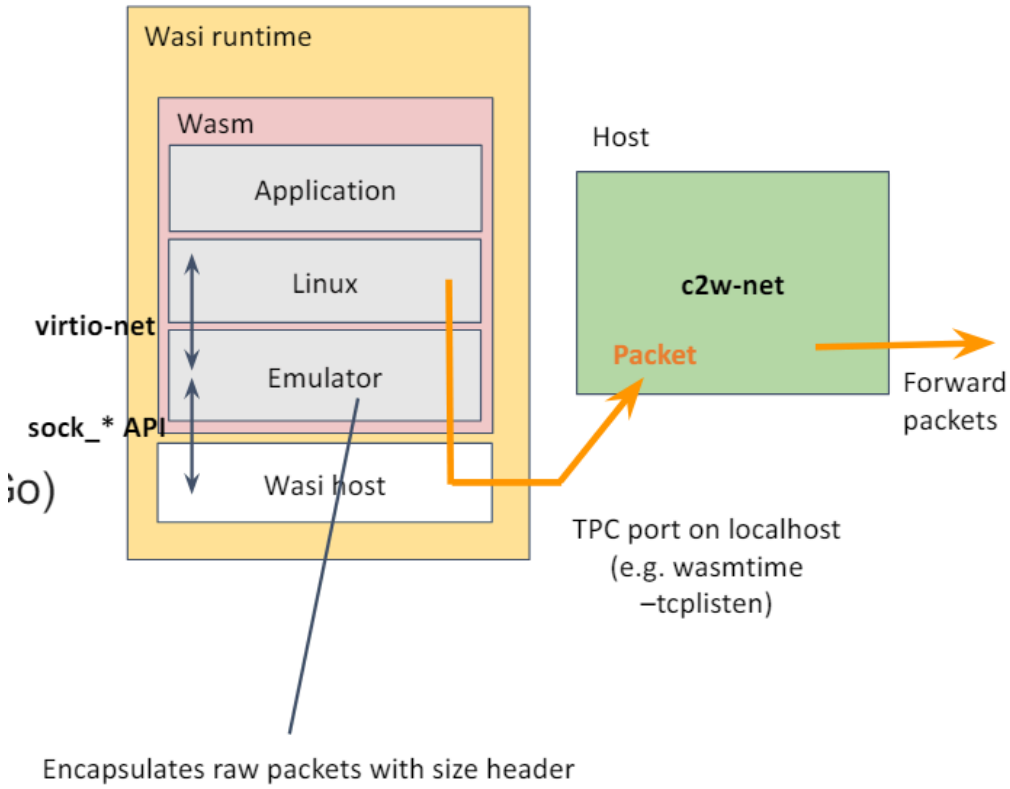
```
$ c2w ubuntu:22.04 ubuntu.wasm
$ wasmtime ubuntu.wasm uname -sm
Linux x86_64
```

- Apps/Linux Kernel in the containers are running on the emulator. (either x86_64 or RISC-V)
- Dependencies (emulator, kernel, container rootfs, runc, etc.) are packaged into a single WASM image

- file system is provided by wasi-vfs through the emulation.
- network is provided outside of wasm runtime, running on the host system.

so this is still limited use case, but possibly explore the current containers running in browsers and wasm runtime. Downside are,

- emulation overhead, image size and loading time and so on...

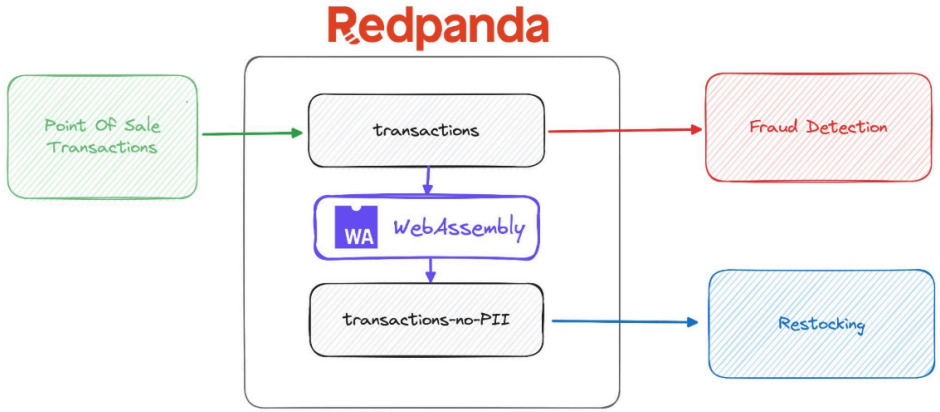


Extending Sidecarless Applications with Wasm in Istio Ambient Mesh

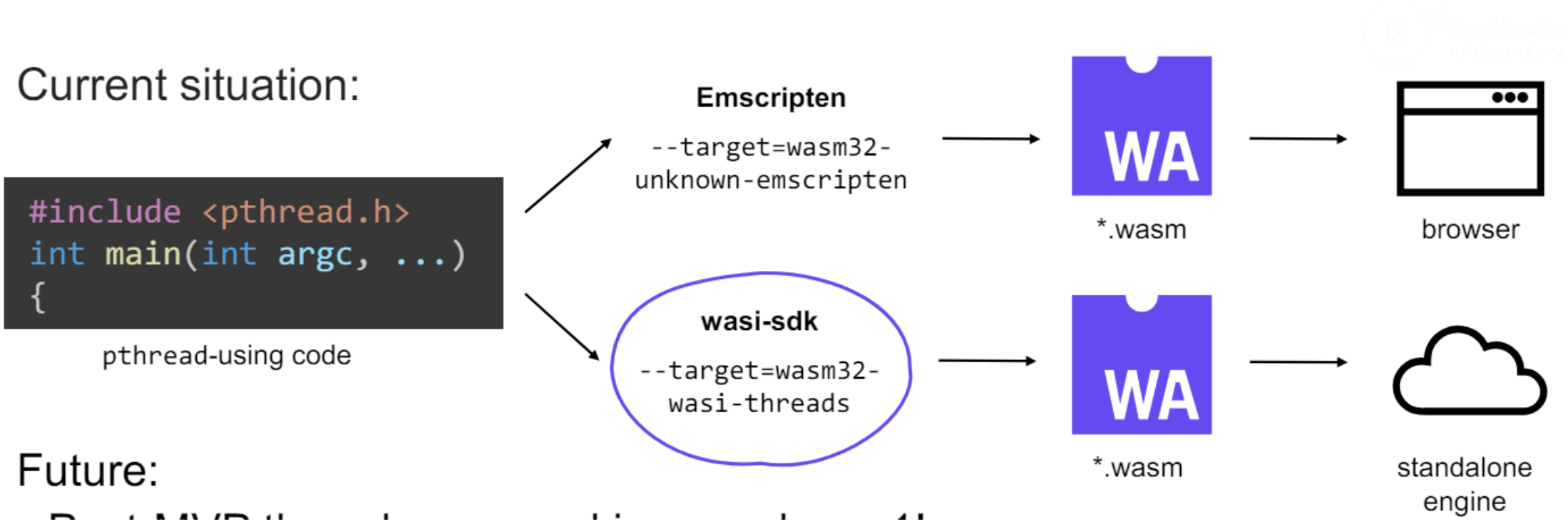
- Istio Ambient Service Mesh uses eBPF for traffic redirection.
- Sidecars must be “injected” into applications by modifying their Kubernetes pod spec and redirecting traffic within the pod.
- eBPF cannot support all the features proxy requires... especially higher levels.
- The Proxy-Wasm sandbox API replaces Mixer as the primary extension mechanism in Istio.

Data Streaming System with WebAssembly

- Inject custom logics in data streaming.
- Language agnostic and isolation.



Exploit Parallelism for AI Workloads with Wasm and OpenMP



- Edge computing has a need for parallel execution (OpenMP) and portability (WebAssembly)
- OpenMP provides fork-join parallelism to exploit inherent parallelism in an application within a shared memory architecture
- OpenMP program in WebAssembly beta, but spawn four threads, print the thread ID compiled with wasi-sdk-20 and executed in Wasmtime.

Wasm Is Becoming the Runtime for LLMs

https://huggingface.co/juntaoyuan/llawa

```
curl -sSf https://raw.githubusercontent.com/WasmEdge/WasmEdge/master/utils/install.sh | bash -s -- --plugins wasi_nn-ggml
curl -LO https://github.com/second-state/llama-utils/raw/main/chat/llama-chat.wasm
wasmedge --dir .:. \
  --nn-preload default:GGML:AUTO:llama-2-7b-chat-wasm-q5_k_m.gguf \
  llama-chat.wasm --reverse-prompt "</s>"
```