kata containers & gVisor

# a Quantitative Comparison

Xu Wang
hyper.sh; Kata Containers Architecture Committee
Fupan Li
hyper.sh; Kata Containers Upstream Developer

HYPER.SH katacontainers

*"All problems in computer science can be solved by another level of indirection, except of course for the problem of* ***too many indirections****."*

*----David Wheeler*

HYPER.SH kata containers

# Agenda

- Secure Containers, or Linux ABI Oriented Virtualization
- Architecture Comparison
- Benchmarks
- The Future of the Secure Container Projects

# What's Kata Containers

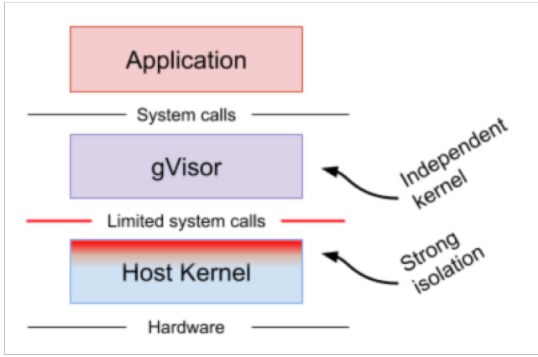
- A container runtime, like runC
- Built w/ virtualization tech, like VM
- Initiated by hyper.sh and Intel®
- Hosted by OpenStack Foundation
- Contributed by Huawei, Google, MSFT, etc.


**Kata Containers is Virtualized Container**

# What's gVisor

- A container runtime, like runC, too
- A user-space kernel, written in Go
  - implements a substantial portion of the Linux system surface
- Developed in Google and was open-sourced in mid-2018

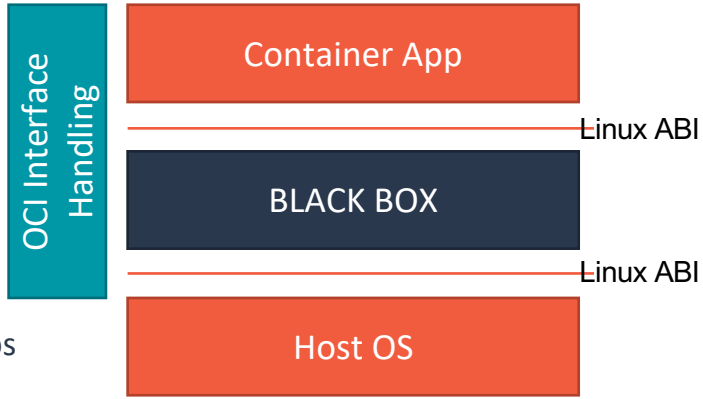
gVisor




gVisor implements Linux by way of Linux.

# The Similarity

- OCI compatible container runtime
  - Linux ABI for container applications
  - Support OCI runtime spec (docker, k8s, zun…)
- Extra isolation layer
  - Guest Kernel, or say, Linux* in Linux
  - Do not expose system interface of the host to the container apps
- Goals
  - Less overhead, better isolation

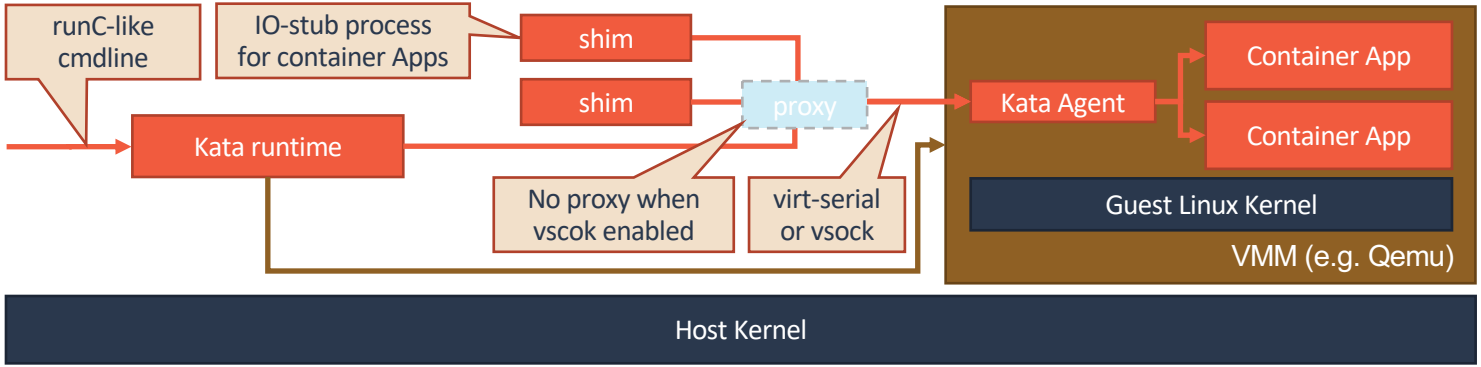



General Architecture of
Kata Containers and gVisor

# Architecture Comparison

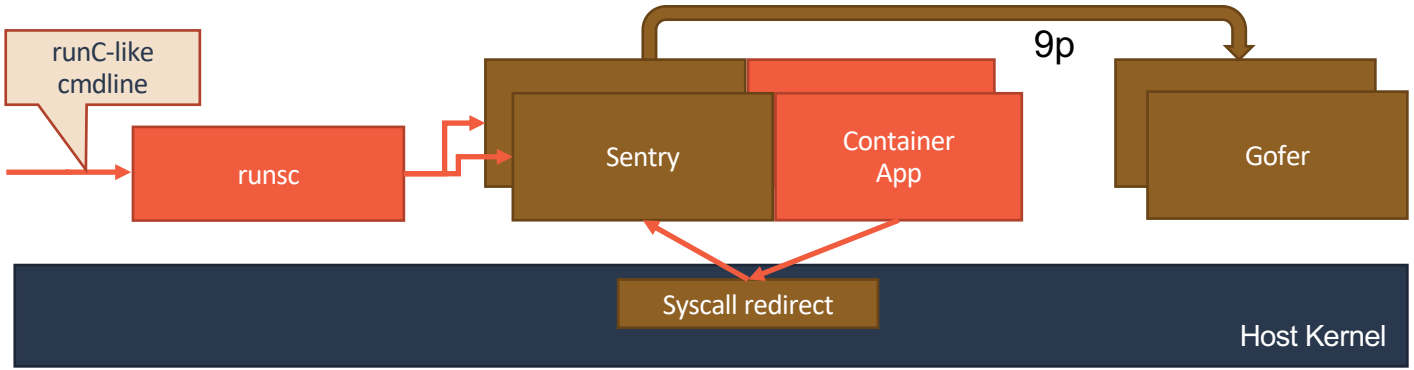
# Run Kata Containers

- Employ existing VMM (or improved) and standard Linux kernel
- Per-container shims and per-sandbox proxy are stand-alone binaries, or built-in as library (such as in containerd shim v2 implementation)
- An agent in guest OS based on libcontainer



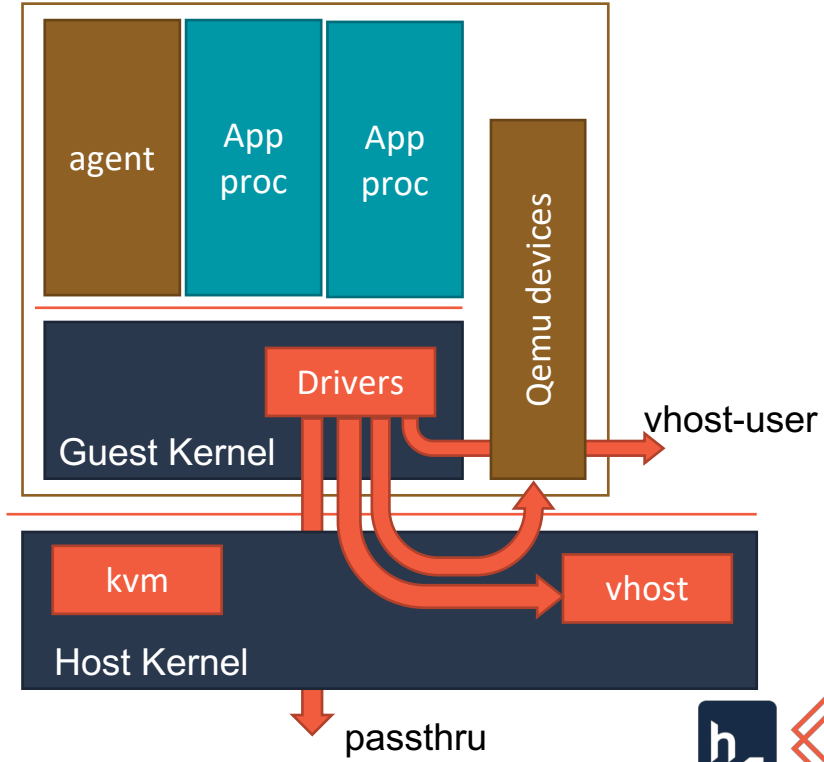
# Run gVisor

- All in one binary (runsc +  Sentry +Gofer)
- Sentry as a user-space kernel, and per-container process stub
- Gofer for 9p IO



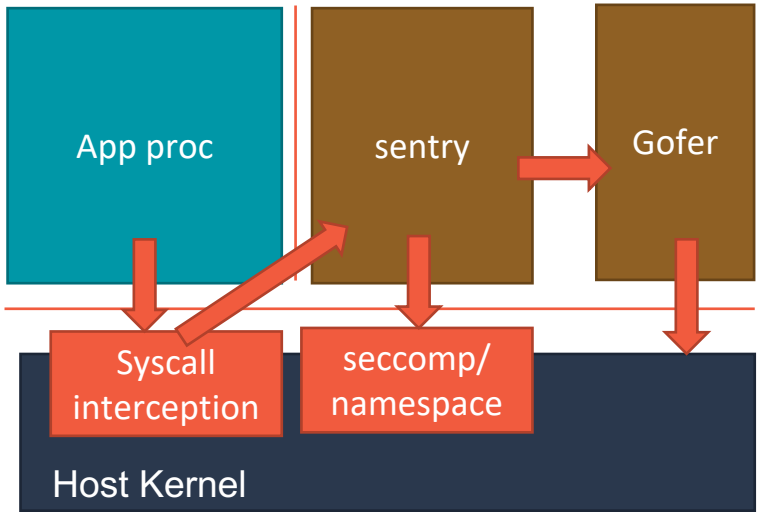
# Detailed Architecture – Kata

- Classic hardware-assisted virtualization
  - Provide virtual hardware interface to Guest
- 2 layers of isolation
  - kvm and CPU ring protection
- Many IO optimization ways
  - Pass through
  - Vhost and vhost-user



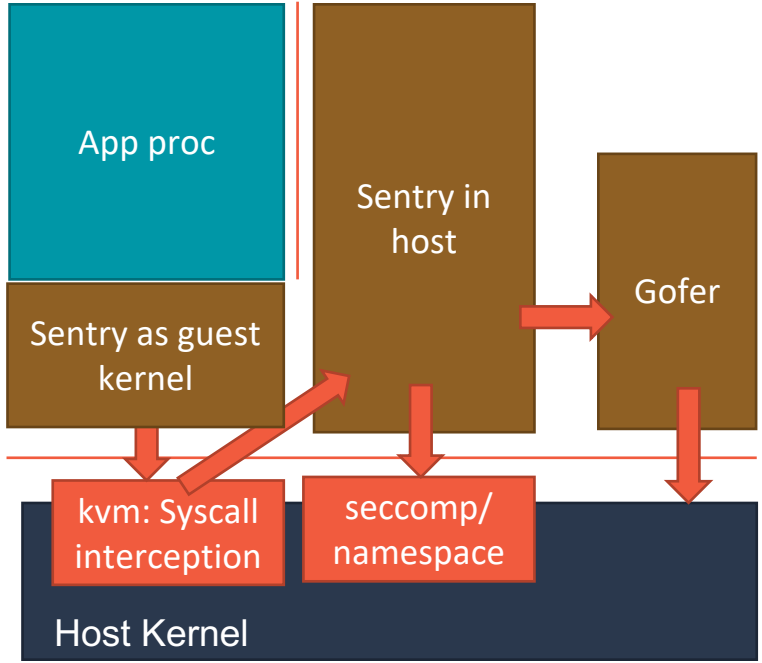
# Detailed Architecture – gVisor (1)

- 2 layers of isolations
  - Minimal kernel and written in Golang
    - Safer but the compatibility…
  - Ring protection and small set of syscalls
    - Avoid access more buggy syscalls
- File operations are proceeded in separated gofer process
- Syscall interception by ptrace or kvm
  - The performance impactions…



# Detailed Architecture – gVisor (2)

- In gVisor-kvm:
- Sentry is both host process and guest kernel
  - Ring0 lower in guest
  - Upper in VM
  - In Host
- Kvm is for higher syscall interception performance rather than isolation
  - No insolation between different modes of sentry itself



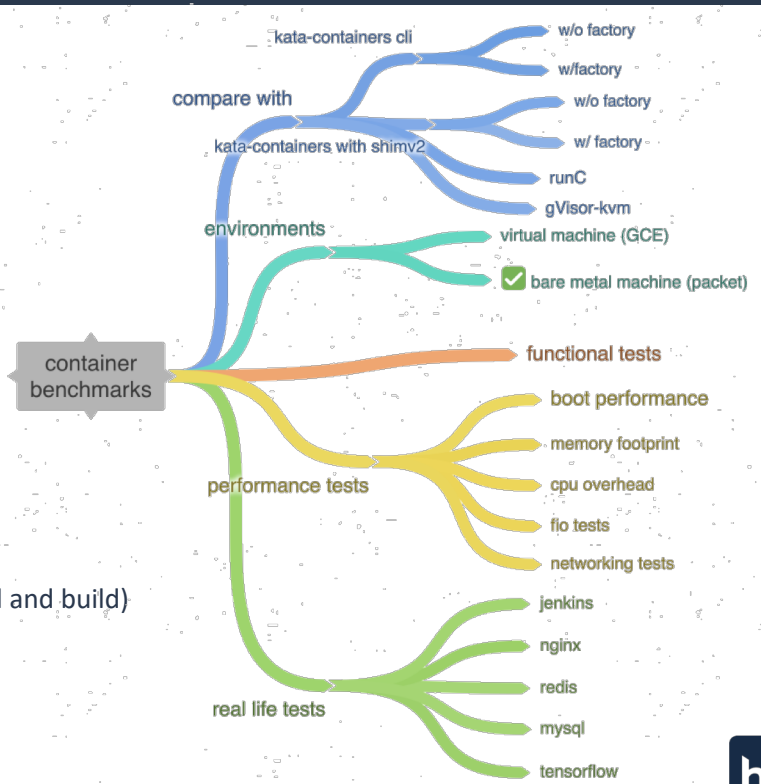
# Expectations based on the Arch

- Isolation:
  - Both include 2 isolation layer – better isolation than runC
- Compatibility:
  - Kata should have better compatibility over gVisor
- Performance:
  - Both should have little overhead on CPU/Memory
  - gVisor should have smaller memory footprint over kata, and may boot faster
  - gVisor may have some pain on syscall heavy (include IO heavy) workloads
  - Kata may have some IO optimization ways

# Benchmarks and Analytics

# Benchmarks & Setup

- Functional Tests
  - Syscall coverage
- Standard Benchmarks
  - Boot time
  - Memory footprint
  - CPU/Memory Performance
  - IO Performance
  - Networking Performance
- Real-life cases
  - nginx, redis, tensorflow
  - gVisor didn't complete a mysql test, and a jenkins test (pull and build) failed due to the 'git clone' failure

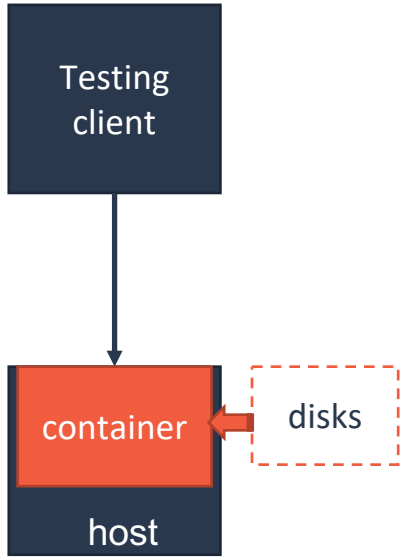


# Testing Setup

- Testing Setup
  - Most of the Cases: server on packet.net

    CPU: 4 Physical Cores @ 2.0 GHz (1 × E3-1578L)
    Memory: 32GB DDR4-2400 ECC RAM
    Storage: 240 GB of SSD (1 × 240 GB)
    Network: 10 Gbps Network (2 × INTEL X710 NIC'S IN TLB)
  - Disk IO cases: a more powerful server with additional disks

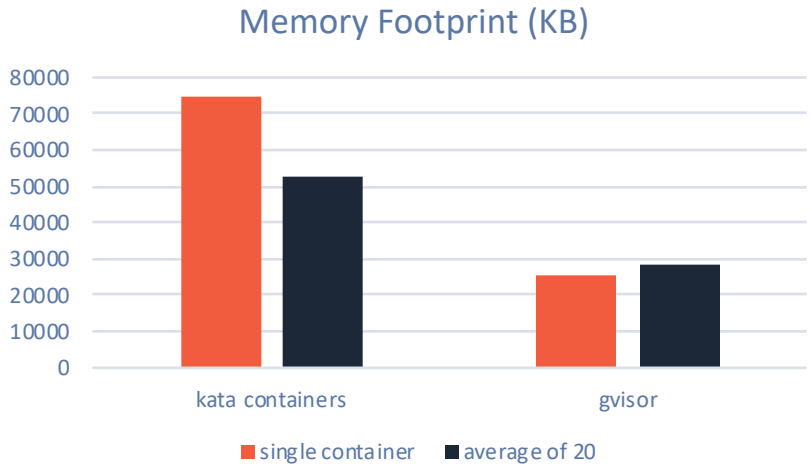
    CPU: 24 Physical Cores @ 2.2 GHz (2 × E5-2650 V4)
    Memory: 256 GB of DDR4 ECC RAM (16 × 16 GB)
    Storage: 2.8 TB of SSD (6 × 480GB SSD)
  - Most of the test containers have 8GB memory if not further noted
  - For networking tests, two servers play roles as client and server



# Syscall Coverage & Memory Footprint

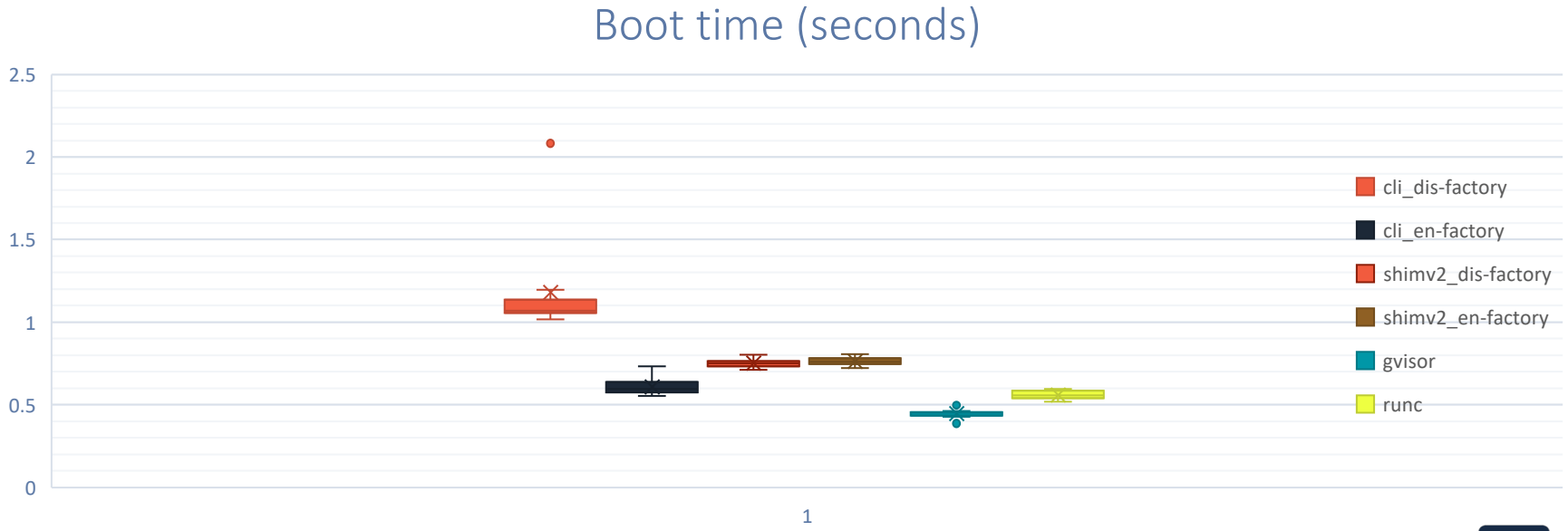
- gVisor will call about 70 syscalls according to the code
- For kata, we collect statistics of the qemu processes in different cases
  - apt-get: 53
  - redis: 35
  - nginx: 36

- With busybox (512MB memory)
- We enabled template for Kata



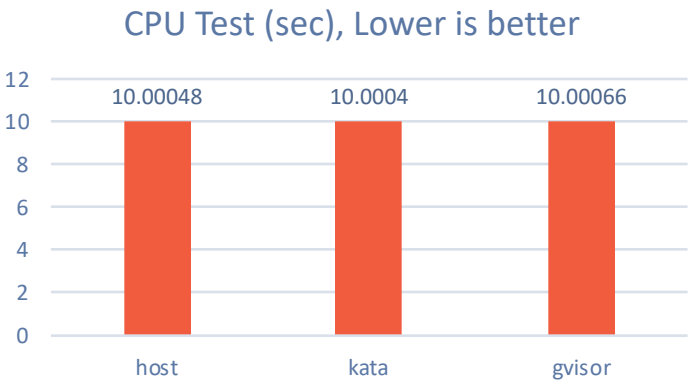
# Boot time

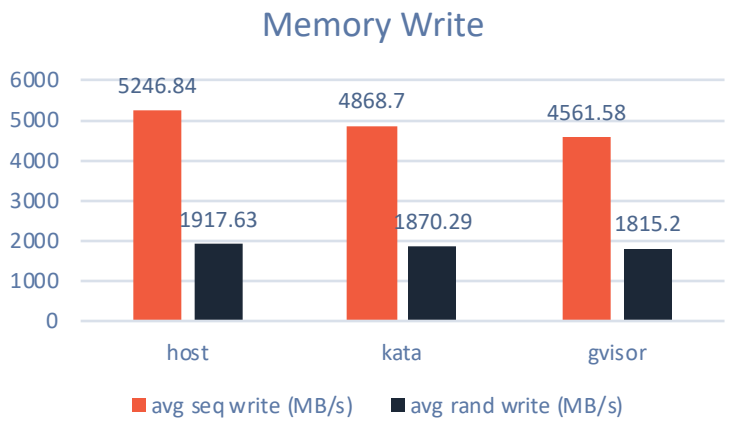
- Boot busybox image
  - Kata (different configurations), gvisor, runc



# CPU/Memory Performance

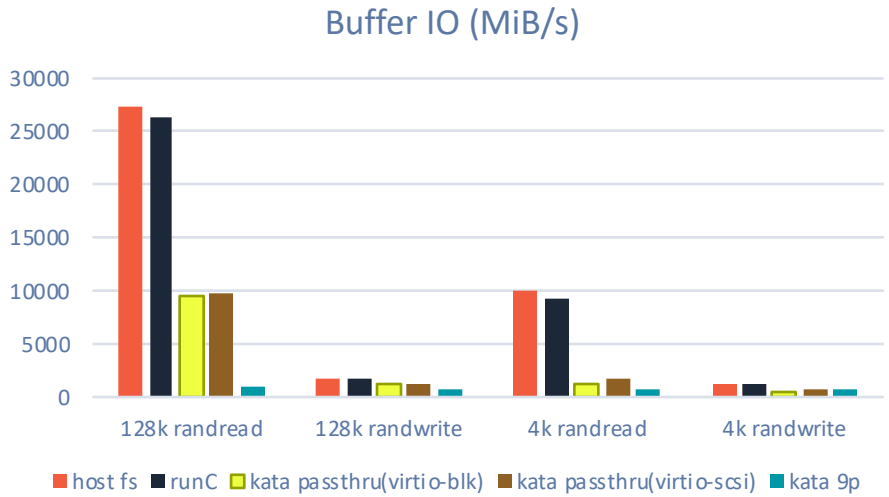
- Sysbench CPU and memory test
  - Test 5 rounds with sysbench and get the average result
- CPU overhead are very limited for both kata and gVisor
- Random memory write: 2.5% vs 5.5% overhead comparing to host
- Seq memory write: 8% vs 13% overhead comparing to host

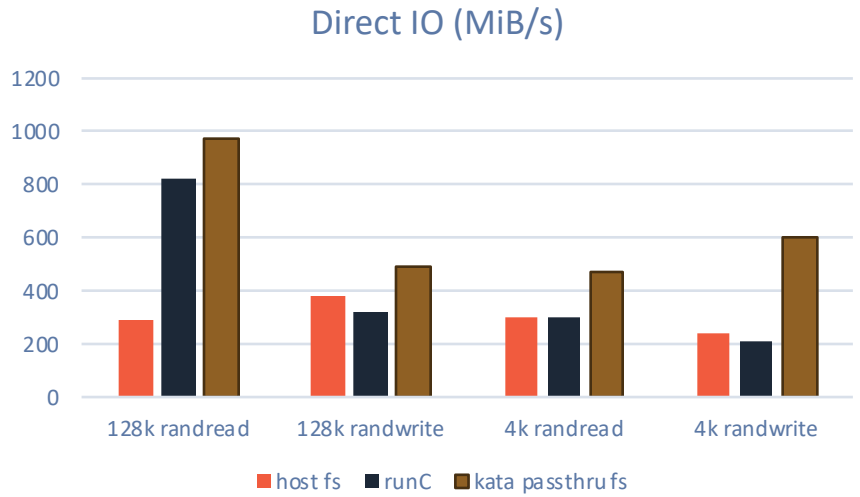


# IO performance (1)

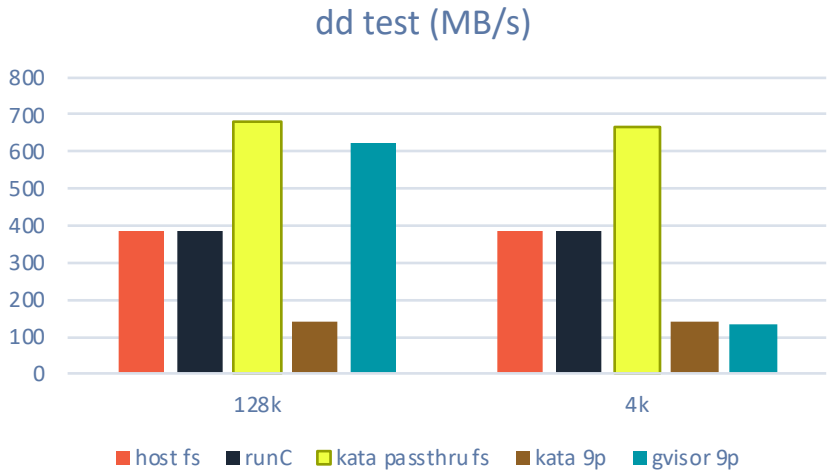
- IO tests with FIO, gVisor didn't complete the test
- Note: The "kata passthru fs" has multi-queue support (in the default kata guest kernel), which is not present in the host driver



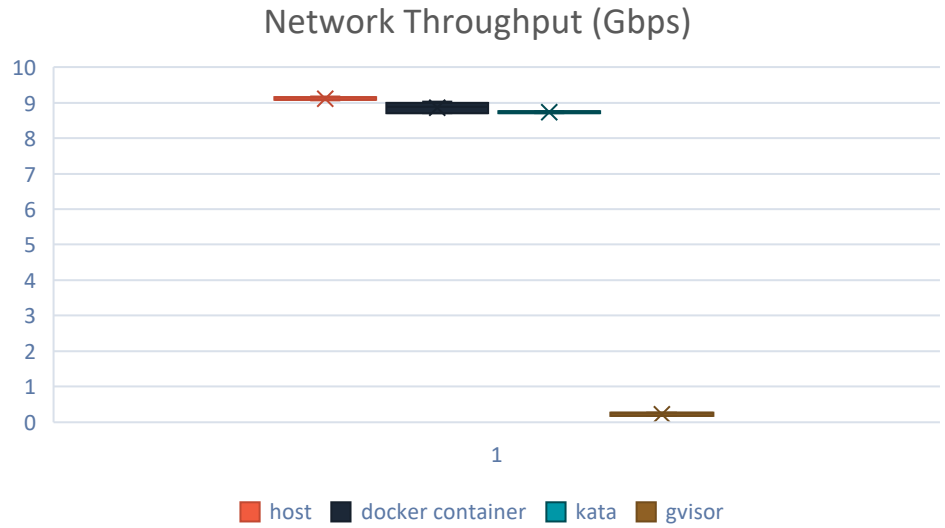
# IO Performance (2)

- IO test with dd command
  - `dd if=/dev/zero of=/test/fio-rand-read bs=4k count=25000000`
  - `dd if=/dev/zero of=/test/fio-rand-read bs=128k count=780000`
- Note: pay attention to the cache of 9p
  - The DIO on a network FS like 9p only means no guest (client) page cache, but there may exist cache on the host (server) side



# Networking Performance

- The networking performance is tested on 10Gb Ethernet
  - In default configurations



# Real-life case: Nginx

- Apache Bench version 2.3
  - `ab -n 50000 -c 100 http://10.100.143.131:8080/`
- General result

| | runC | Kata* | gVisor | | |
|---|---|---|---|---|---|
| Concurrency Level | 100 | 100 | 100 | | |
| Time taken for tests | 3.455 | 3.439 | 161.338 | seconds | |
| Complete requests | 50000 | 50000 | 50000 | | |
| Failed requests | 0 | 0 | 0 | | |
| Total transferred | 42250000 | 42700000 | 42250000 | bytes | |
| HTML transferred | 30600000 | 30600000 | 30600000 | bytes | |
| Requests per second | 14473.73 | 14541.18 | 309.91 | [#/sec] | (mean) |
| Time per request | 6.909 | 6.877 | 322.677 | [ms] | (mean) |
| Time per request | 0.069 | 0.069 | 3.227 | [ms] | (mean, across all concurrent requests) |
| Transfer rate | 11943.65 | 12127.12 | 255.73 | [Kbytes/sec] | received |

* The www-root of the kata test case is not located on 9pfs

# Real-life case: Nginx (Cont)

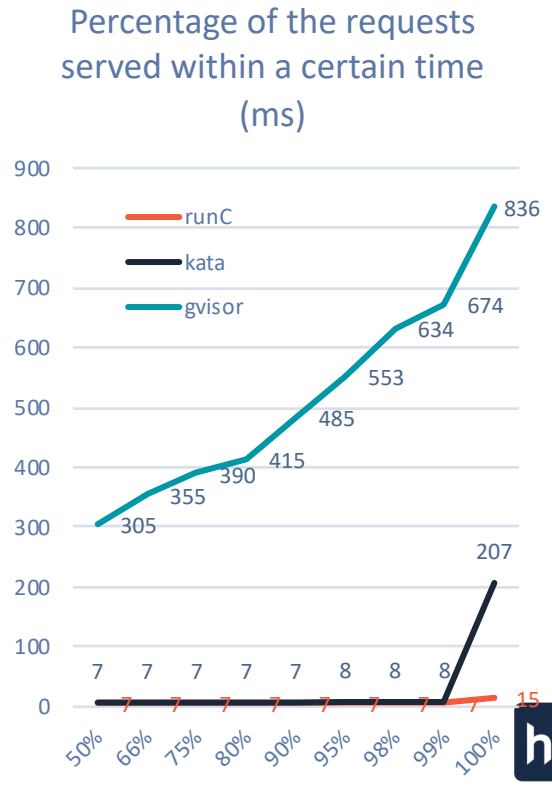
- Detailed Connection time(ms) and percentage

| runc | min | mean[+/-sd] | | median | max |
|---|---|---|---|---|---|
| Connect | 0 | 0 | 0.3 | 0 | 7 |
| Processing | 1 | 7 | 0.3 | 7 | 14 |
| Waiting | 1 | 7 | 0.3 | 7 | 8 |
| Total | 3 | 7 | 0.4 | 7 | 15 |

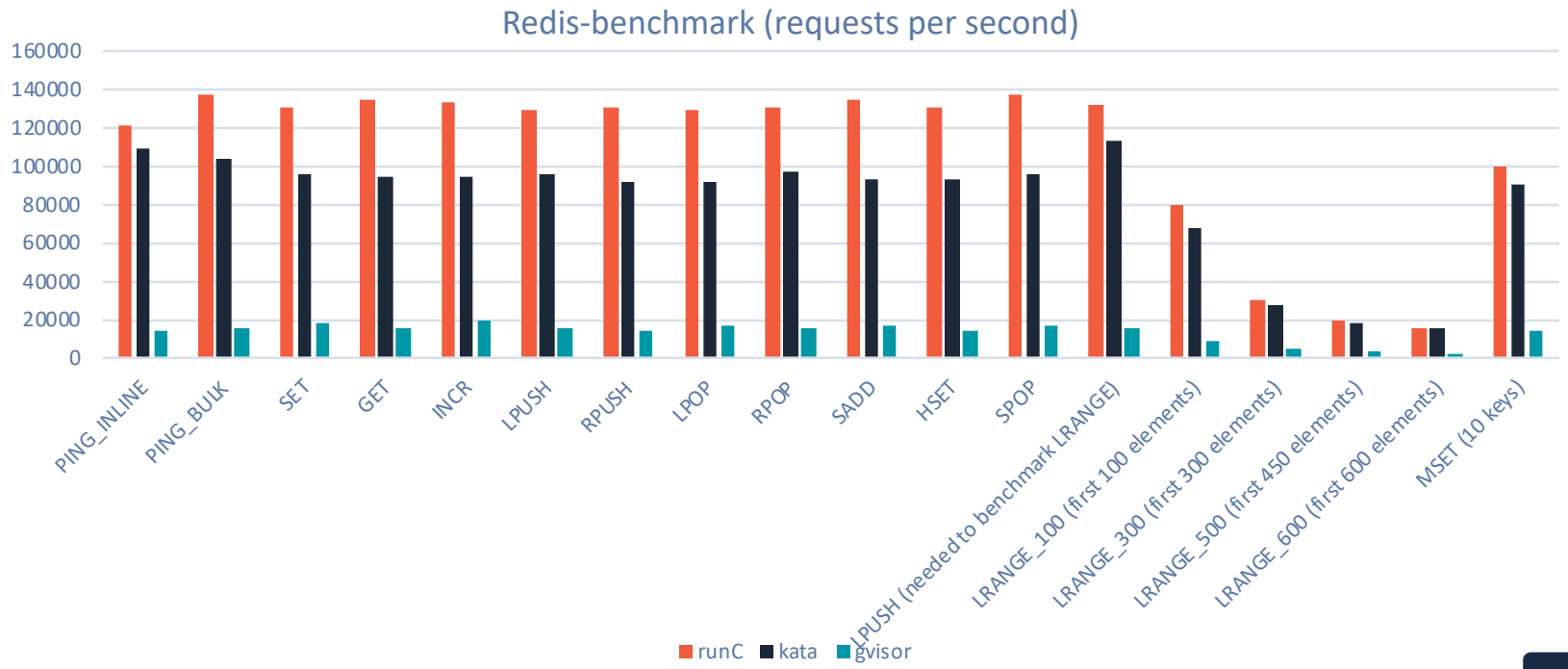
| kata | min | mean[+/-sd] | | median | max |
|---|---|---|---|---|---|
| Connect | 0 | 0 | 0.1 | 0 | 7 |
| Processing | 2 | 7 | 1 | 7 | 207 |
| Waiting | 2 | 7 | 1 | 7 | 207 |
| Total | 3 | 7 | 1 | 7 | 207 |

| gvisor | min | mean[+/-sd] | | median | max |
|---|---|---|---|---|---|
| Connect | 0 | 0 | 0.1 | 0 | 2 |
| Processing | 44 | 322 | 119.8 | 305 | 836 |
| Waiting | 21 | 322 | 119.9 | 304 | 836 |
| Total | 45 | 322 | 119.8 | 305 | 836 |

* The www-root of the kata test case is not located on 9pfs



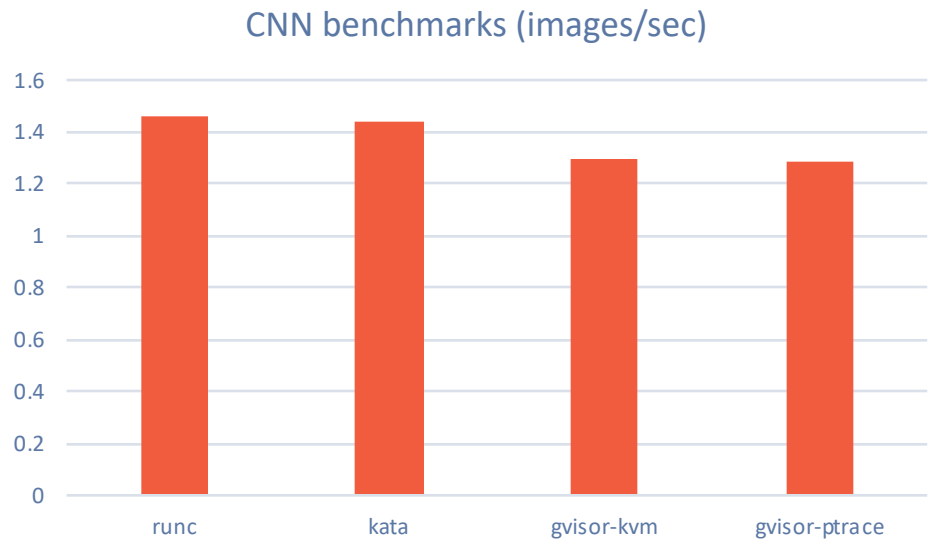
# Real-life case: Redis



# Real-life case: Tensorflow (CPU)

Note: this is a CPU test (No GPU)

| | |
|---|---|
| TensorFlow | 1.10 |
| Model | resnet50 |
| Dataset | imagenet (synthetic) |
| Mode | training |
| SingleSess | False |
| Batch size | 32 global |
| | 32 per device |
| Num batches | 100 |
| Num epochs | 0 |
| Devices | ['/cpu:0'] |
| Data format | NHWC |
| Optimizer | sgd |



# Summary

- Both kata and gVisor
  - Small attack interface
  - CPU performance identical to host and runC
- On Kata Containers
  - Some suggestions: Passthru fs, Factory and/or shimv2, etc.
  - Good performance in most cases with proper configuration
- On gVisor
  - Fast boot performance and low memory footprint
  - Still need improve the compatibility for general usage
- Furthermore tests
  - Fine tuned benchmarks, vhost-user/dpdk etc. and NUMA Specific tests, nested virtualization tests
  - Different kinds of networking tests
  - More real-life cases

# The Future of the Projects

# Kata Containers

- Kata is on the way more efficient
  - The shim v2 shown in the benchmarks is going to be merged soon
  - The nemu will reduce the VMM overhead
  - Multiple network improvement is under developing or testing
- And more featureful
  - Better GPU and other accelerator support
- Contributions are welcome

# gVisor and Similar Projects

- gVisor shows real lightweight secure sandboxing tech
- gVisor need more work on
  - More efficient syscall interception and processing
  - Better compatibility for more applications
- Another project, linuxd, is based on linux kernel and doing similar jobs
  - Lai, Jiangshan, Containerize Linux Kernel, OpenSource Summit 2018, Vancouver (https://schd.ws/hosted_files/ossna18/db/Containerize%20Linux%20Kernel.pdf)

# Q&A

Contributions are Welcome