ImageJockey: A Framework for Container Performance Engineering

Takeshi Yoshimura, Rina Nakazawa, Tatsuhiro Chiba

IBM Research - Tokyo

IEEE CLOUD 2020

Container image

- A container image is a key software building block in the cloud
  - Assembles software binaries for apps on container platforms
  - Enables lightweight app deployment regardless of underlying S/W & H/W

- >100K images are distributed in Docker Hub [https://hub.docker.com]
  - DB & web images for large-scale services
  - Programming runtime images for software development
  - Linux OS images to containerize new apps

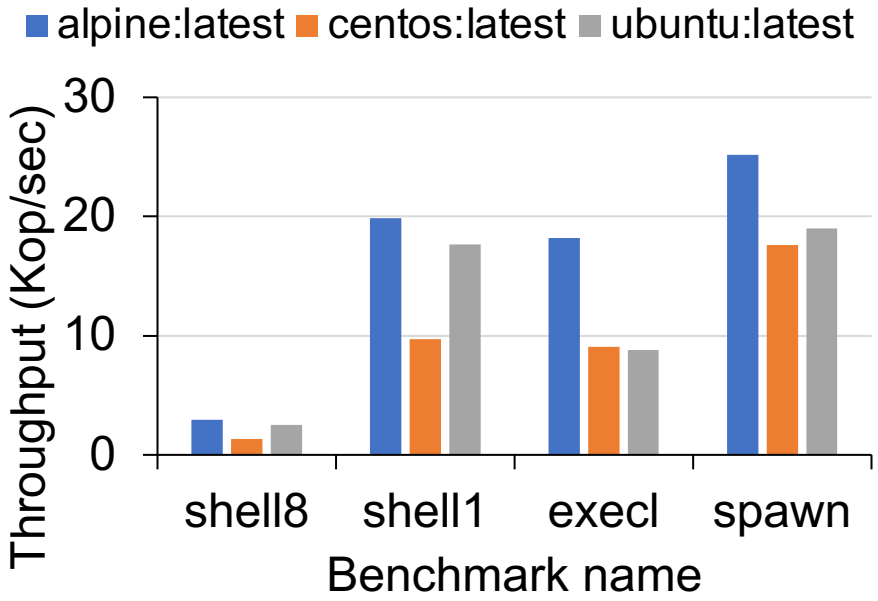
Problem: Too many similar images

- Docker Hub offers many Linux OS images
  - Ubuntu: 240 images, 278 tags
  - CentOS: 21 images, 43 tags
  - Alpine: 38 images, 48 tags
  - Busy Box, Fedora, Debian, ALT Linux, Oracle Linux, Clear Linux, etc.

- Many Linux apps can run on various Linux images
- Do Linux images highly affect app performance?

UnixBench performance

- Alpine showed >50% UnixBench throughput compared with Ubuntu and CentOS
  - Non-negligible for performance-sensitive apps
  - Performance variation could happen on other base images, e.g., JDK (see our paper)
- Performance testing remains important for containerized apps



m4.xlarge on AWS EC2 with four virtual CPUs, 16 GB RAM, and a 100 GB attached volume. Host: Ubuntu 18.04LTS, Docker19.01, Linux 4.15.

Contribution

- *ImageJockey*: a framework for container performance engineering
  - Help engineers write, run, and analyze benchmarks for container images
- Case study with ImageJockey
  - Performance analysis of OS, web, DB, Python, and Java images

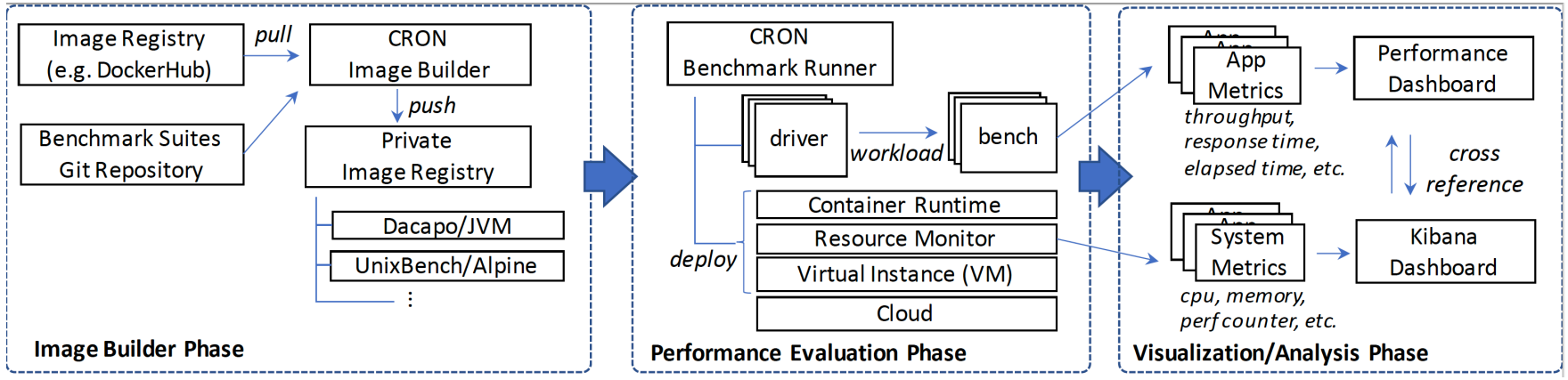
Existing performance testing tools

- CloudBench [Silva+ IC2E13], CloudPerf [Michael+ ICPE17], ReBench, ASV, etc.
  - Enable deep analysis of workloads with various performance metrics
  - Focus on non-container workloads: bare-metal and hypervisors

- Unique challenges to containerize performance analysis:
  - Inter-container communication
  - Metrics collection
  - Different image build scheme
  - Frequent updates of base images

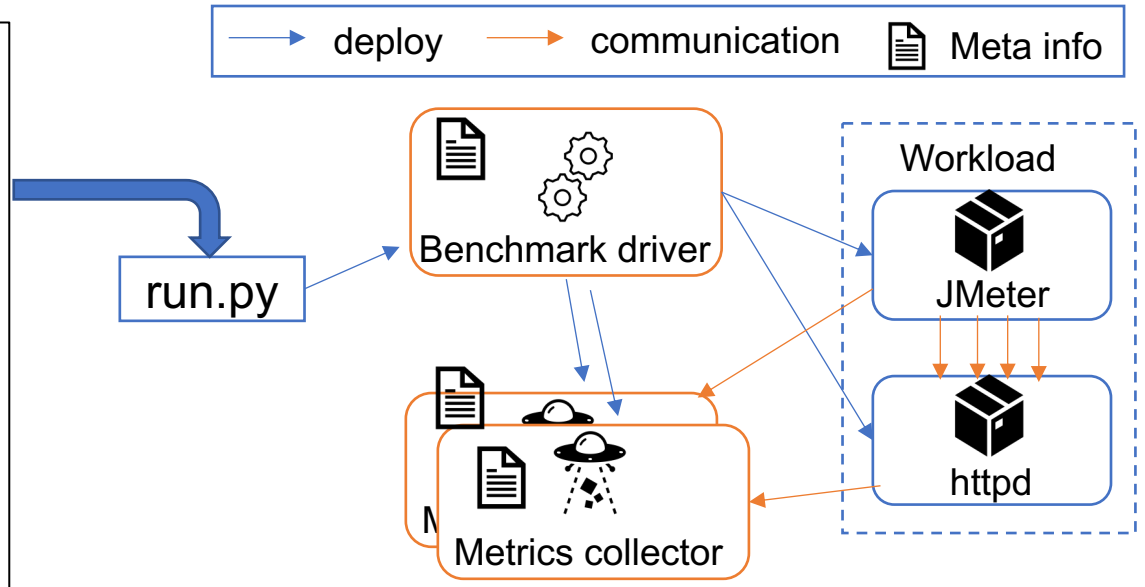
ImageJockey: framework for container performance engineering

- Help engineers write, run, and analyze benchmarks for container images
  - **Python library** and **benchmark driver** to run benchmarks
  - **Periodic image builder** to keep benchmark images up to date
  - **Metrics collection** to analyze containerized benchmarks
  - **Visualization** for large-scale experimental results



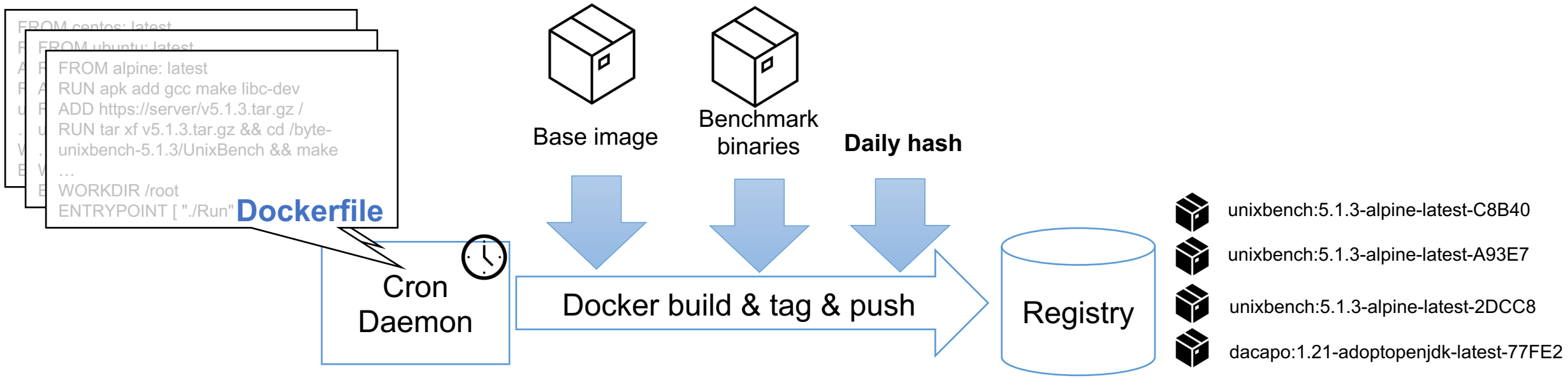
Python library and benchmark driver

```
ports ={'9080/tcp': '9080', '9443/tcp': '9443'}
dt_img = conf.get_image('daytrader')
bd = BenchmarkDescription("daytrader", ...)
dt = DockerMon(ContainerRunner(dt_img, bd, ports=ports))
time.sleep(10)
bd2 = BenchmarkDescription('client_stdout', ...)
jm_img = conf.get_image('jmeter')
args =  ['-JHOST=' + dt.co.ip(), '-JTHREADS=4']
jmeter = ContainerRunner(jm_img, bd2, args, locallog=True)
dm = DockerMon(jmeter)
jmeter.join()
log = jmeter.log()
dm.join()
dt.stop()
... # parse log
```



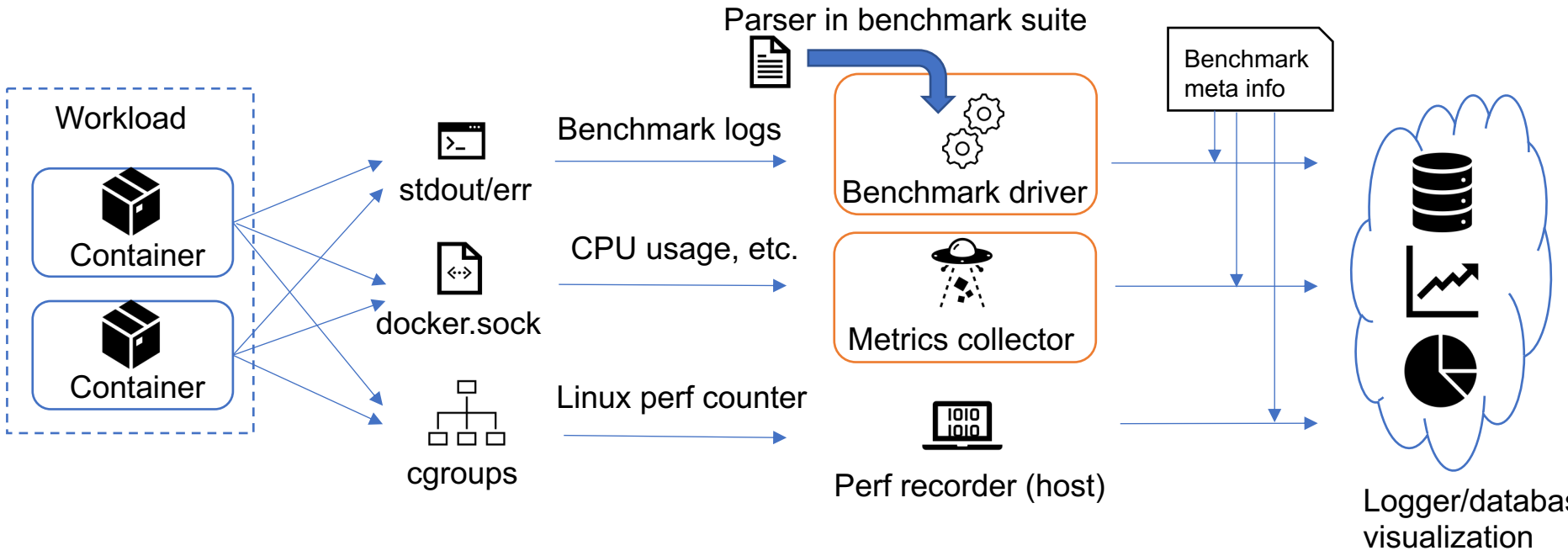
- Python library helps users define typical container communication and metrics collection
  - Benchmark driver deploys containers and associates them with benchmark meta info

Periodic image builder



- Build, update, and preserve benchmark images with daily hash tags
  - Developers define & register Dockerfiles for benchmark images
  - Can work with GitOps

Metrics collection



- Leverage Docker/cgroups APIs to collect computing resource usage/perf
- Allow custom log parser in benchmark suites to collect app metrics

Visualization with web dashboard

- Enable capturing overall trends of experiments, status, and failures
- Summarize application metrics e.g., elapsed time and throughput

Case Study

- Comparison of Java, Python, DB, web, and OS images
  - Tested under an AWS EC2 m4.xlarge from February 14 to March 25
  - Measured total throughput or elapsed time for every benchmark

- Post-hoc analysis of UnixBench and Dacapo

Tested images and benchmarks

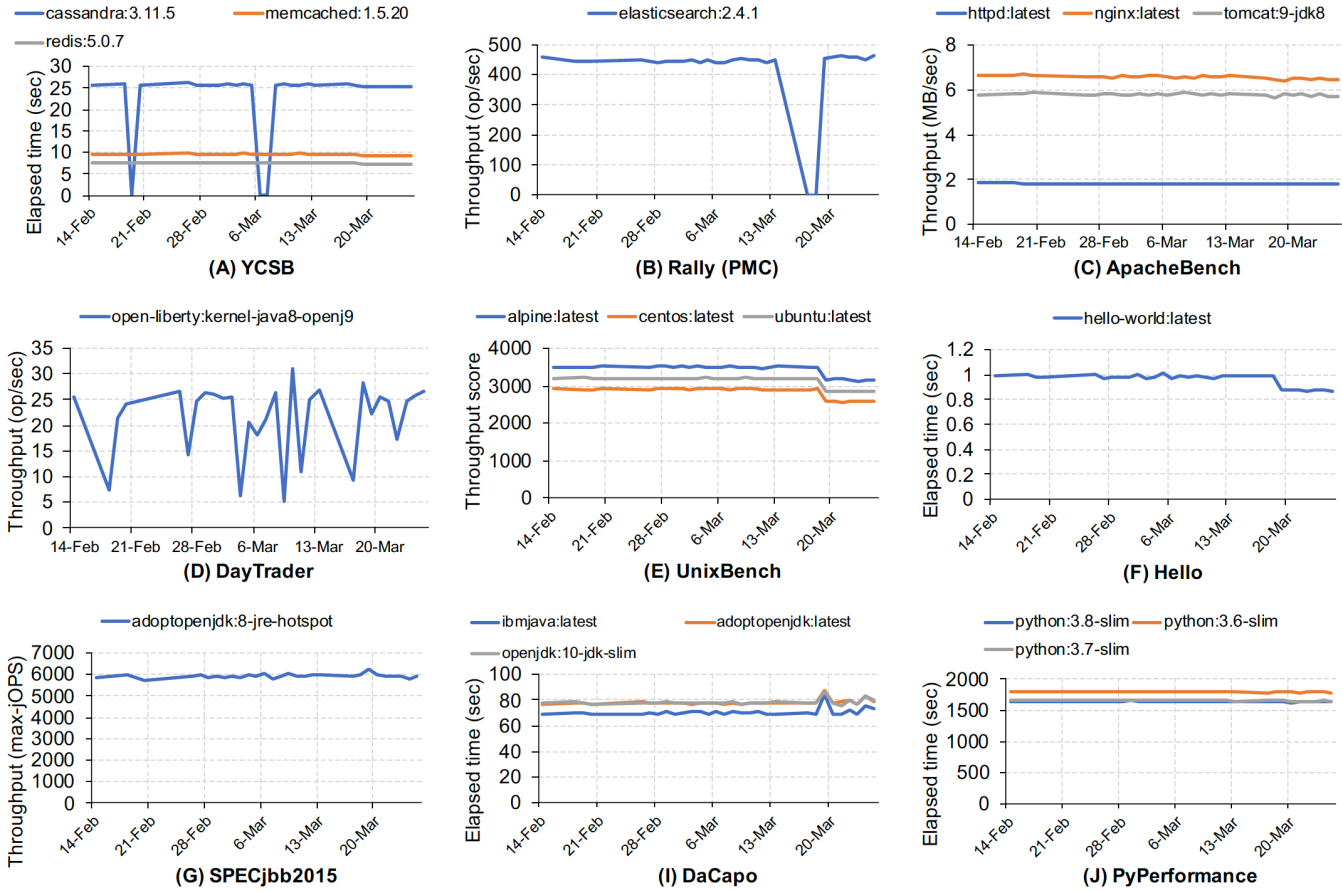
Image type	Image	Tags
Java	adoptopenjdk	latest, 8-jre-hostpot
	openjdk	10-jdk-slim
	ibmjava	latest
Python	python	3.8-slim, 3.7-slim, 3.6-slim
DB	redis	latest
	memcached	latest
	cassandra	latest
	elasticsearch	2.4.1
Web	httpd	latest
	nginx	latest
	tomcat	9-jdk8-adoptopenjdk-hotspot
	open-liberty	kernel-java8-openj9
OS	alpine	latest
	ubuntu	latest
	centos	latest
Misc	hello-world	latest

Benchmark	Evaluated Images	Workload
DaCapo	Java	avrova, fop, h2, luindex, jython, lusearch, pmd, sunflow, tradebeans, tradesoap, xalan
SPECjbb2015	Java	COMPOSITE
pyperformance	Python	2to3, chameleon, crypto_pyaes, deltablue, django_template, frankuch, float, and other 40 workloads
YCSB	DB	Workload A, B, C
Rally	elasticsearch	pmc
ApacheBench	Web	10 threads+1000 requests
DayTrader+JMeter	open-liberty	4 threads
UnixBench	OS	context1, dhry2reg, execl, fsbuffer, fsdisk, fstime, pipe, etc.

Comparison of container images

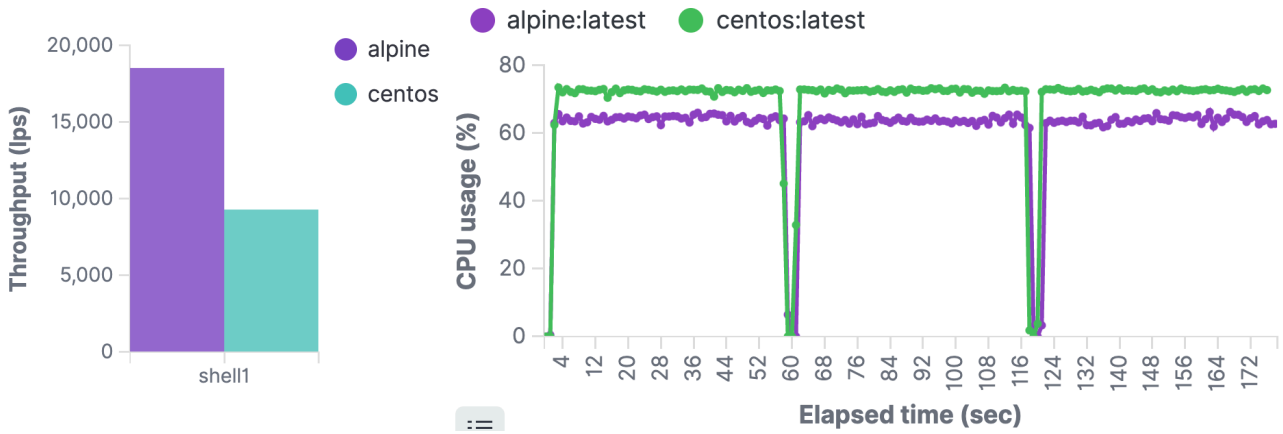
m4.xlarge on AWS EC2 with four virtual CPUs, 16 GB RAM, and a 100 GB attached volume. Host: Ubuntu 18.04LTS, Docker 19.01, Linux 4.15.

- Container images had non-negligible impact on performance
  - Nginx showed 3x better throughput than httpd
  - IBM Java showed better performance than AdoptOpenJDK and OpenJDK 10
  - Newer Python improved performance
  - Alpine showed the best UnixBench throughput
  - Storage reconfiguration affected many images (but not all)



Analysis of UnixBench.shell1

- Internal system library affected CPU usage
  - Alpine uses `musl`, which is a minimum system library for containers
  - Images providing both `fat' and `slim' (or -alpine) need to assess both behaviors inherent with different system libraries



Score (left) and CPU usage (right) for UnixBench.shell1

UnixBench.shell1: Hot functions of alpine:latest

```
Samples: 2K o
  Overhead  C          Shared Object          Symbol
+    6.09%  od         ld-musl-x86_64.so.1    [.] memcpy
+    5.37%  sort       ld-musl-x86_64.so.1    [.] getc_unlocked
+    3.93%  od         ld-musl-x86_64.so.1    [.] printf_core
+    2.42%  sort       ld-musl-x86_64.so.1    [.] memcpy
+    1.78%  grep       ld-musl-x86_64.so.1    [.] regexec
```

UnixBench.shell1: Hot functions of centos:latest

```
Samples: 2K
  Overhead  Command    Shared Object          Symbol
+    1.85%  sort       ld-2.28.so             [.] strcmp
+    1.58%  sh         [kernel.kallsyms]      [k] __do_page_fault
+    1.41%  sort       ld-2.28.so             [.] do_lookup_x
+    1.18%  od         libc-2.28.so           [.] vfprintf
+    1.08%  sh         [kernel.kallsyms]      [k] filemap_map_pages
+    1.04%  od         ld-2.28.so             [.] strcmp
```

Summary

- *ImageJockey*: a framework for container performance engineering
  - Periodic image updates, benchmark driver, metrics collection, and visualization
- Case study with ImageJockey
  - Non-negligible performance variation of OS, web, DB, Python, and Java images
  - Different system libraries of Alpine changed UnixBench CPU usage