RedHat OpenShift Container Platform on Z Networking Performance

2020-11-11

Dr.-Ing. Axel Busch

axel.busch@ibm.com
Linux on IBM z Performance

ibm.biz/perfradar – the Performance Radar Blog

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States, other countries, or both.

Not all common law marks used by IBM are listed on this page. Failure of a mark to appear does not mean that IBM does not use the mark nor does it mean that the product is not actively marketed or is not significant within its relevant market.

Those trademarks followed by ® are registered trademarks of IBM in the United States; all others are trademarks or common law marks of IBM in the United States.

For a more complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml:

*CICS®, DataPower®, DB2®, e business(logo)®, ESCON, eServer, FICON®, IBM®, IBM (logo)®, IMS, MVS, OS/390®, POWER6®, POWER6+, POWER7®, Power Architecture®, PowerVM®, PureFlex, PureSystems, S/390®, Sysplex Timer®, System p®, System p5, System x®, System z®, System z9®, System z10®, WebSphere®, X-Architecture®, z13®, z13s®, z Systems®, z9®, z/Architecture®, z/OS®, z/VM®, z/VSE®, zEnterprise®, zSeries®, IBM Z®, IBM z Systems®, IBM z13®, IBM z13s®, IBM z14®, IBM LinuxONE

The following are trademarks or registered trademarks of other companies.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both. Open vSwitch and OvS are trademarks of The Linux Foundation.

*** All other products may be trademarks or registered trademarks of their respective companies.**

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured Sync new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

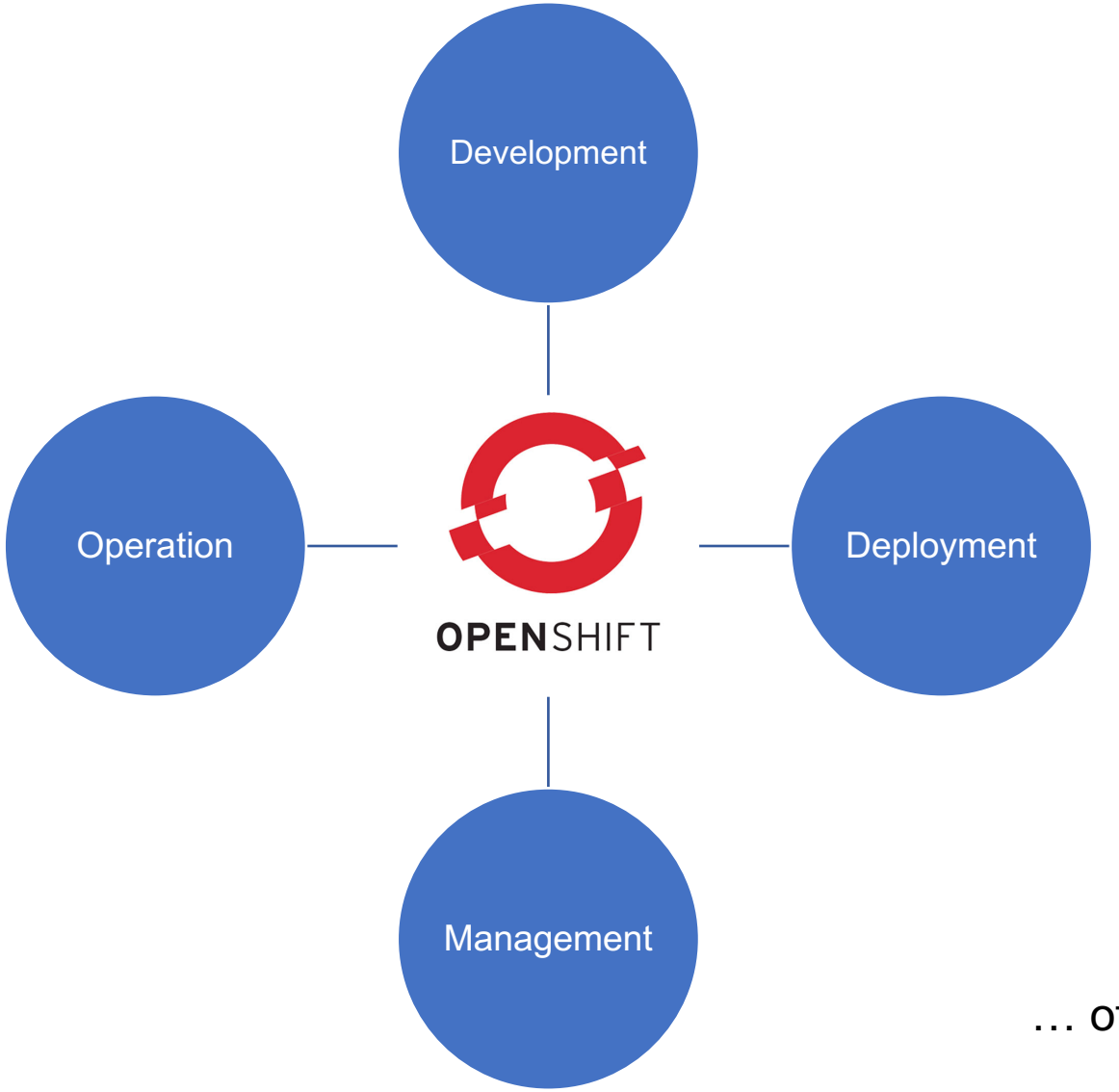
This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained Sync the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

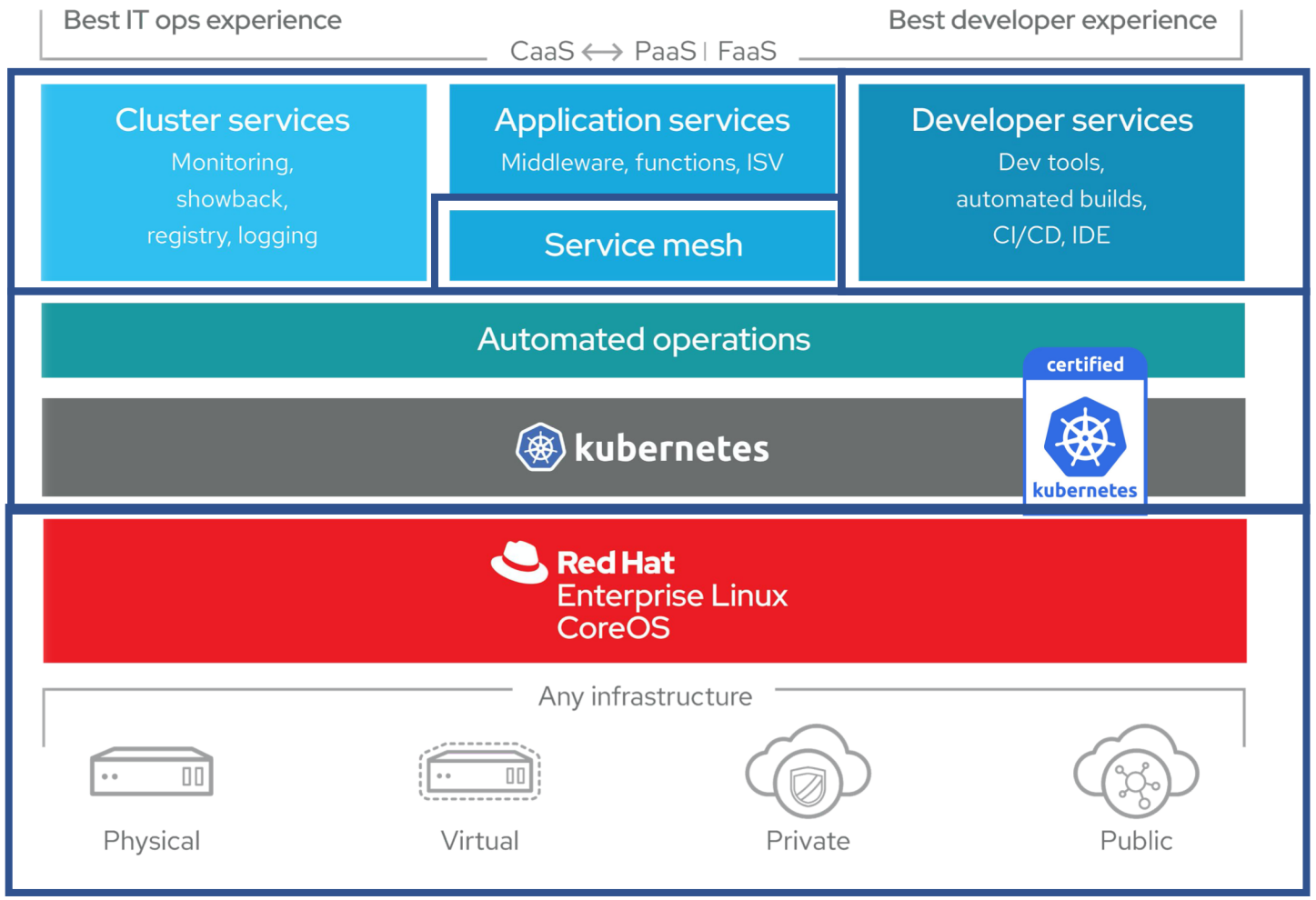
Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

OpenShift Container Platform in a Nutshell



… of cloud applications & infrastructure (automatically)

OpenShift Container Platform in a Nutshell



- Manage
  - resources, .e.g. public/private cloud
  - infrastructure, e.g. network
  - operating system, e.g. Linux performance settings
- Development
  - of applications using CI/CD pipeline
  - … in different programming languages
  - … and code repositories
- Deployment
  - of applications in containers
  - … to be scaled
  - … and made reliable
- Operation
  - Monitoring, logging
  - Middleware, operators
  - Network topology

Why considering OCP Networking Performance?

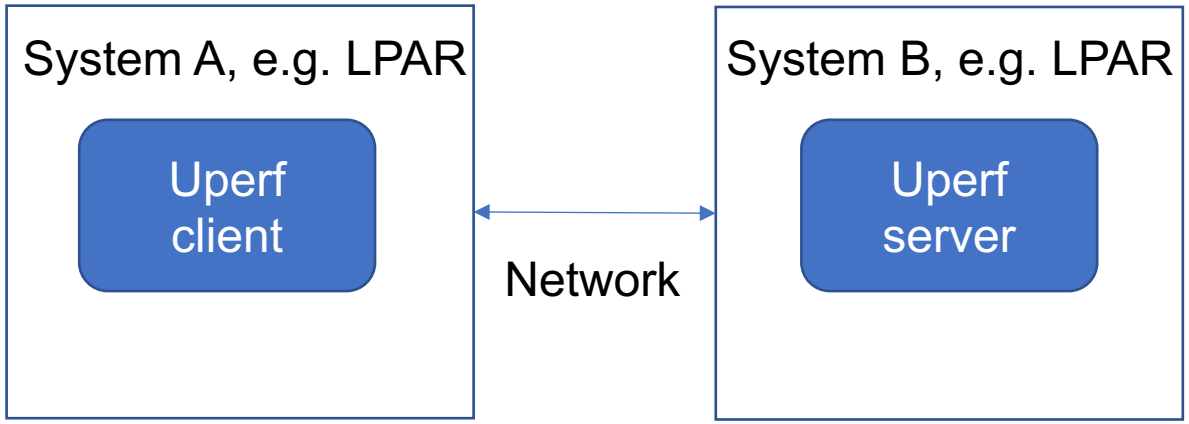
- Business perspective
  - Networking performance, i.e. latency, throughput **business critical** quality attribute
  - Network Performance **critical** in microservice architectures through interdependencies
- Technical perspective
  - OCP networking architecture **complex system**
  - Several **new technologies**
  - **Not much documentation** publicly available
  - **Limited experience** with intertwinement of technologies

Container technology **more difficult** to analyse compared to LPAR
- More layers, e.g. software-defined network (SDN)
- Limited insights and monitoring capabilities (“grey box“)

What‘s the **influence** of the OCP architecture such as SDN on networking performance?

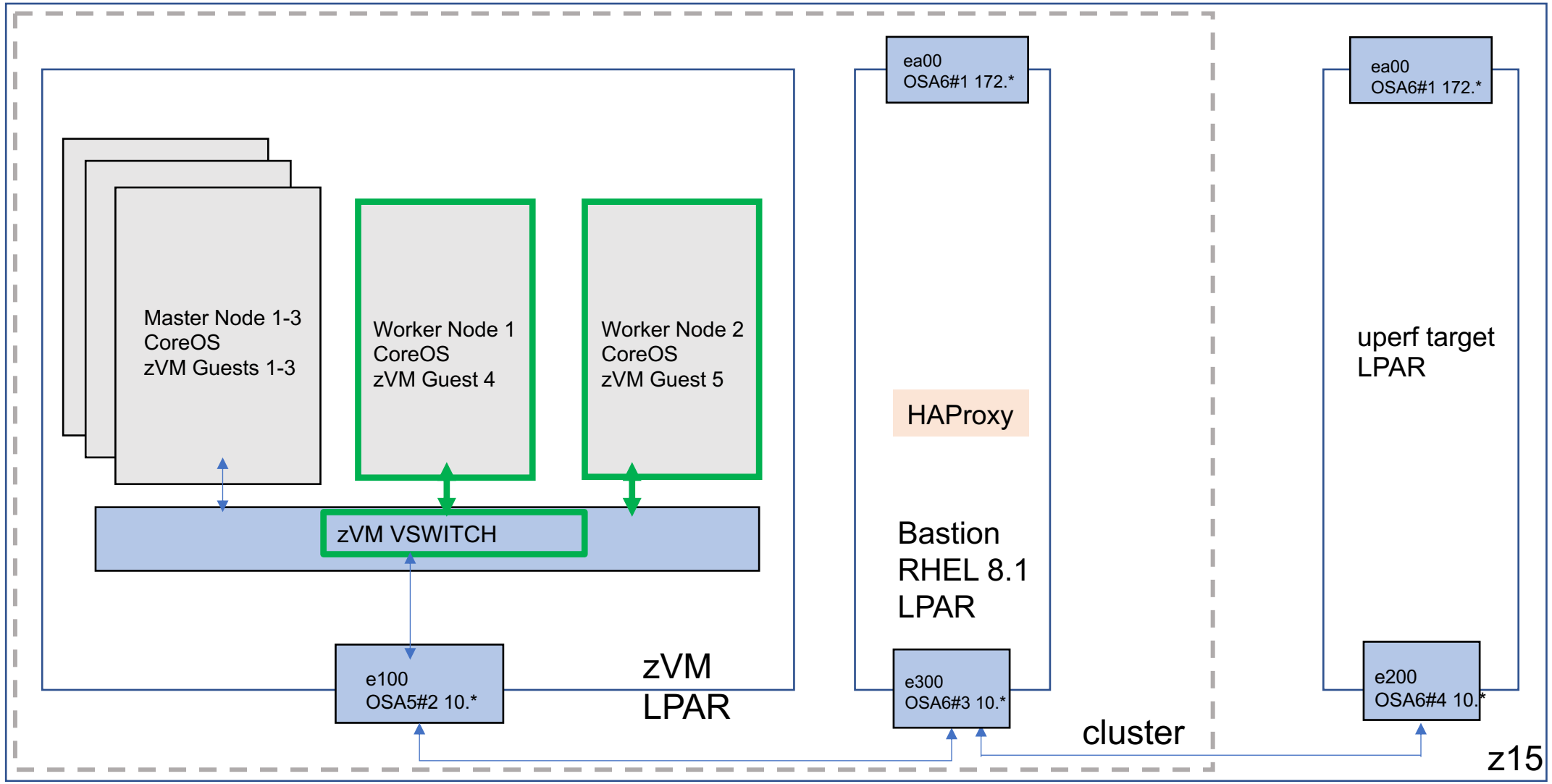
How to Benchmark the Network: uperf & workloads

Uperf: A network (micro) benchmark
- Two sets of workloads
  - **Request Response** (latency)
  - **Streaming** (throughput)
- Several numbers of simultaneous **connections** (1-50-250)
- Different **request sizes** (1x1-200x1000-200x30000 B)
- Typically used in distro regressions
- (Distro results used as OCP baseline to be compared to)



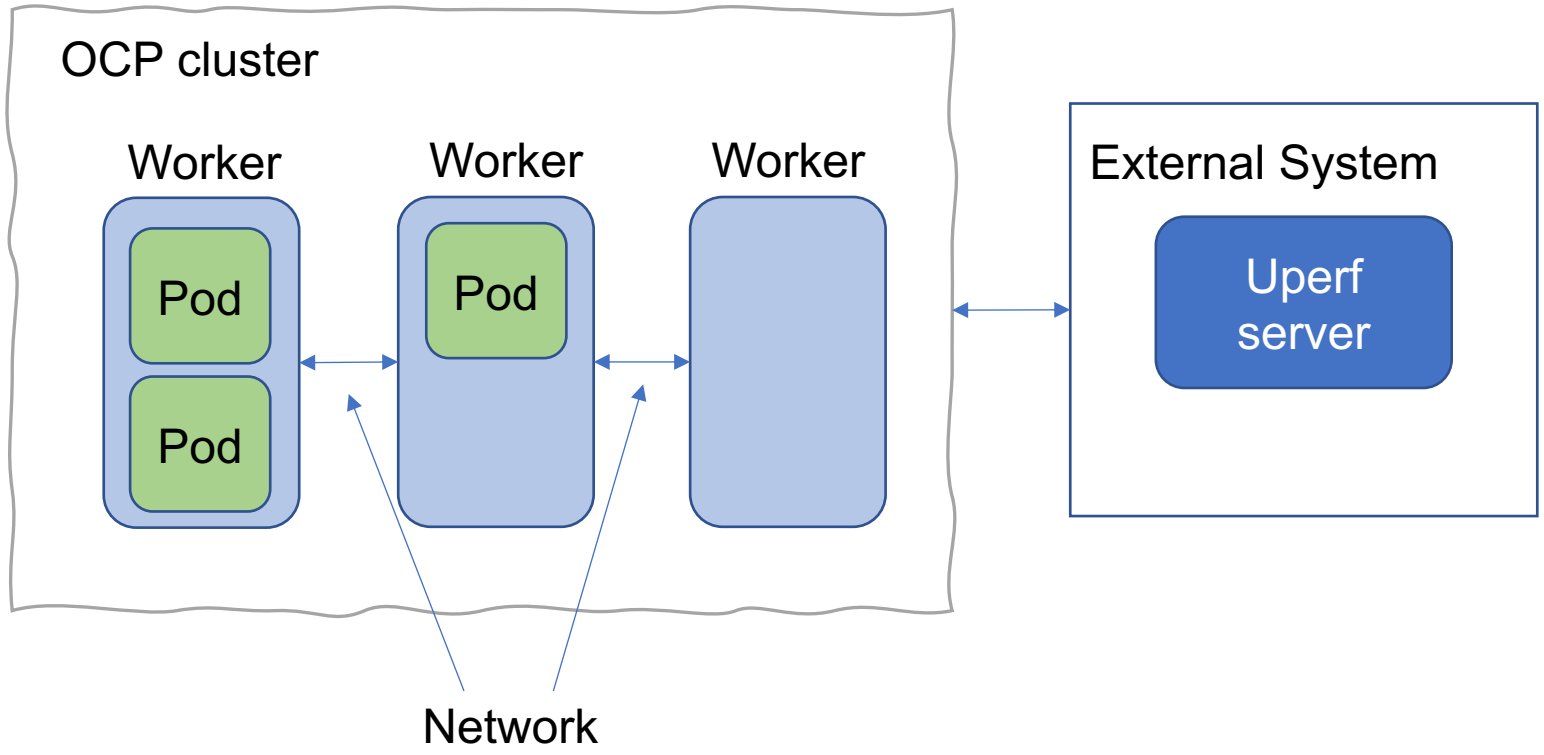
Results:
- Latency in us/ms
- Throughput in MiB/s

OpenShift (on z) Container Platform: System Architecture



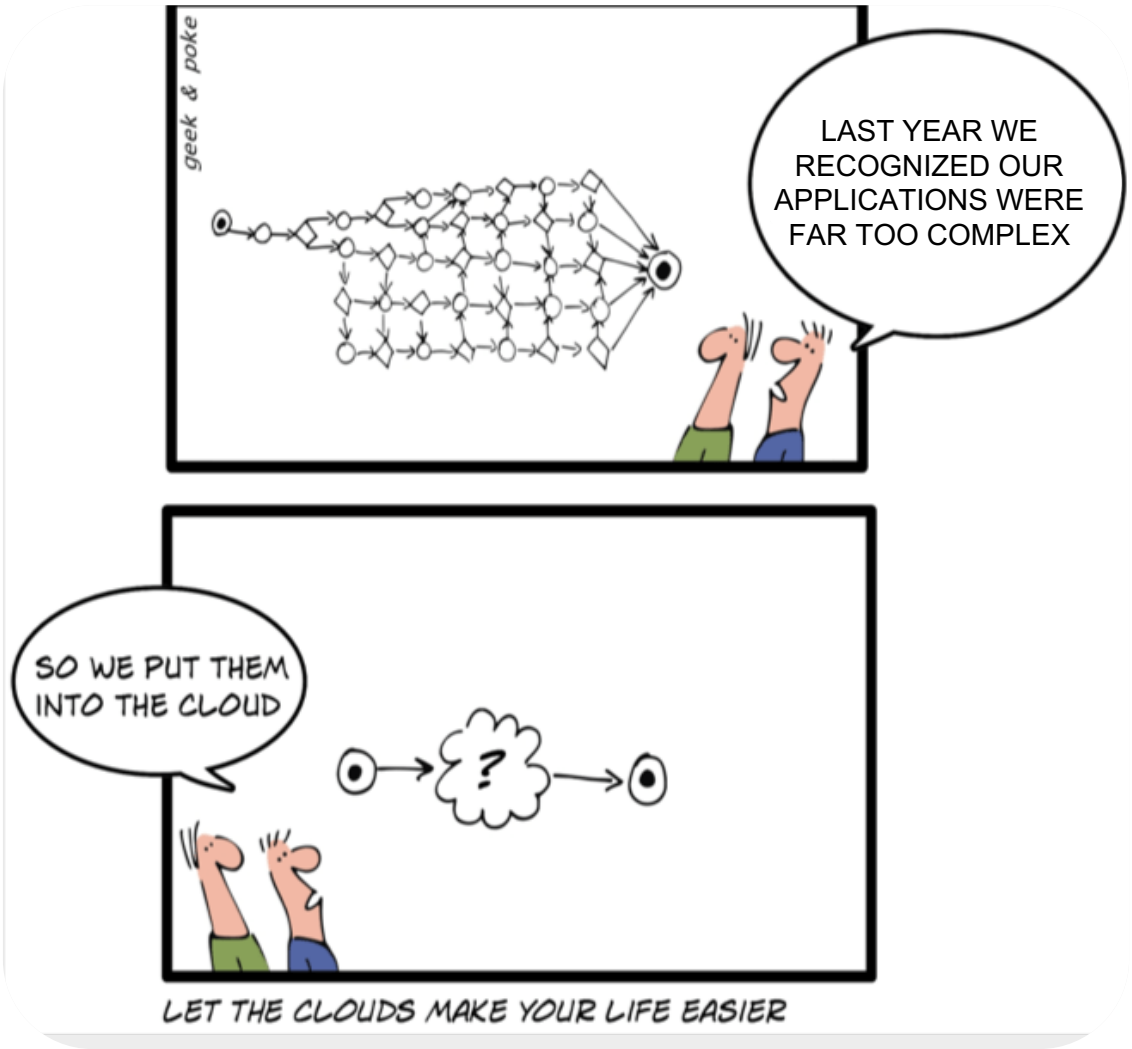
How to Benchmark the OCP Network: scenarios

1. Worker 2 Worker performance
2. Pod 2 Pod performance
3. Pod to external service performance



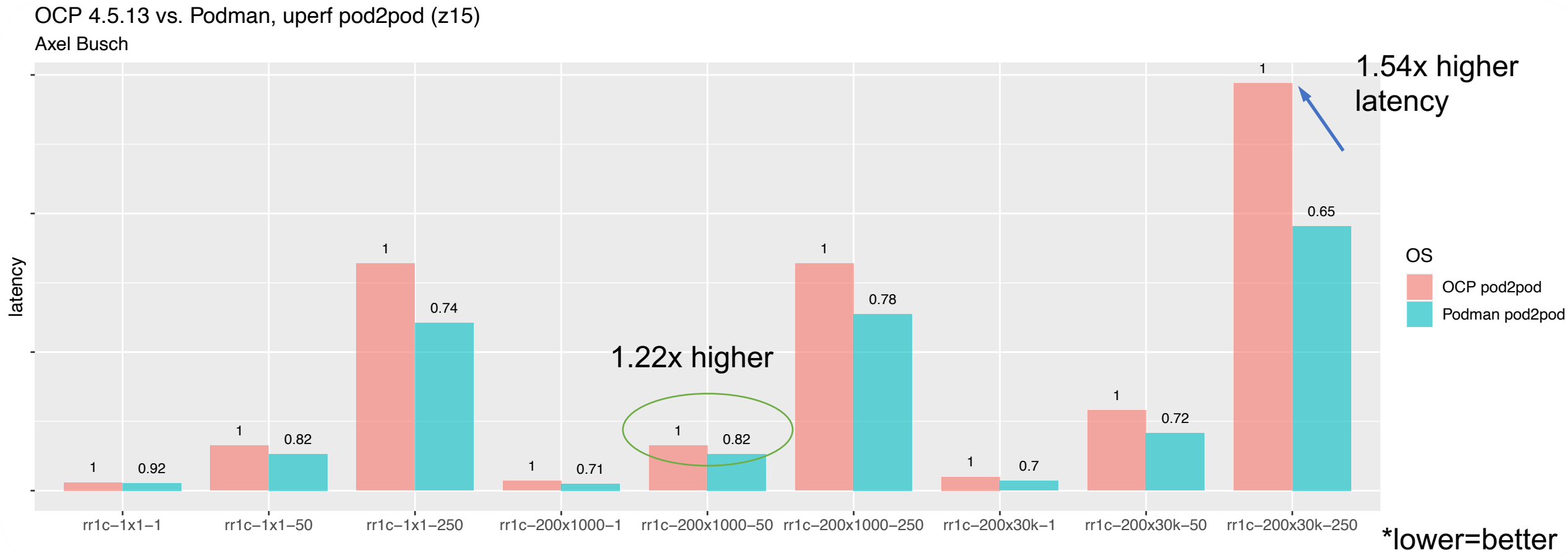
Focus on pod 2 pod and pod 2 external configuration!

Complexity

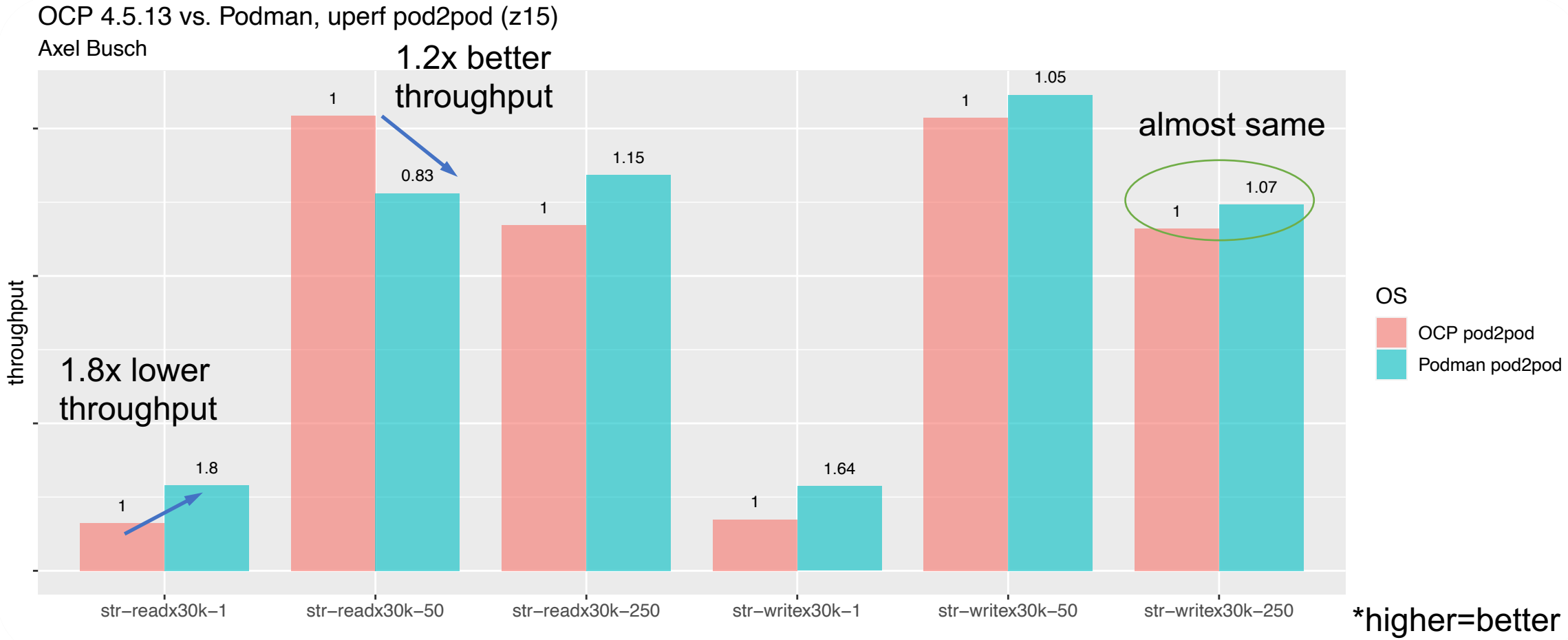


http://geek-and-poke.com/

Scenario results: Pod to pod performance (1)

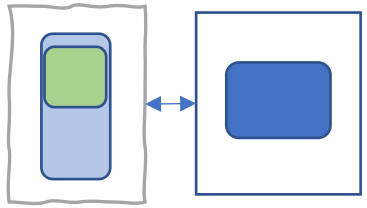


Scenario results: Pod to pod performance (2)



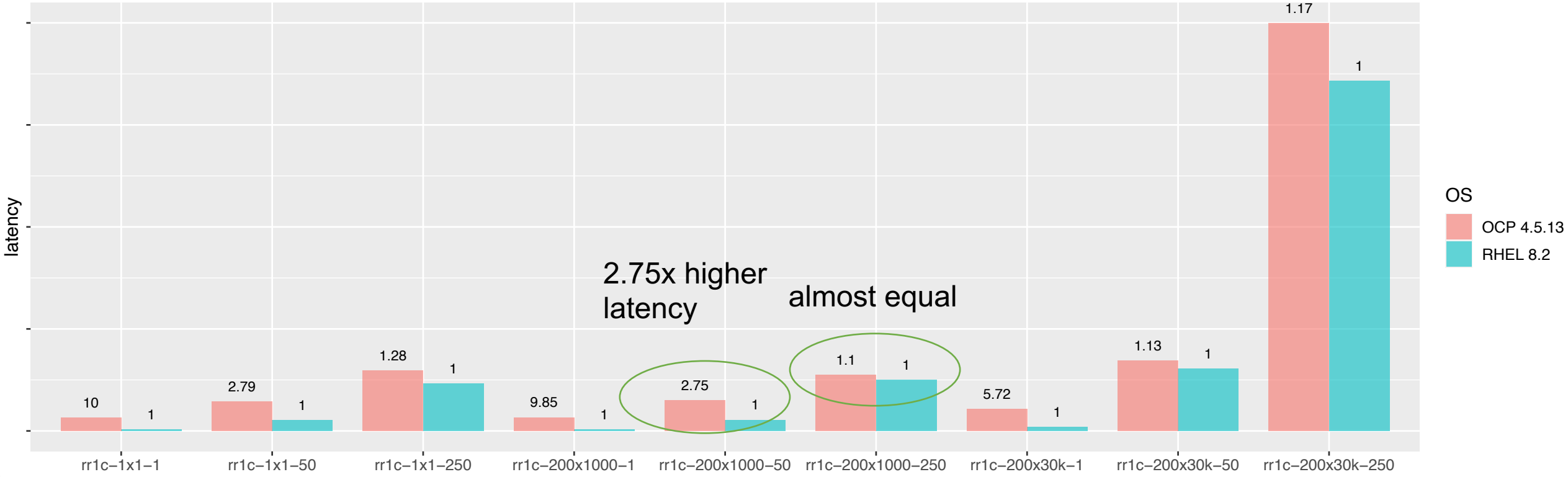
*higher=better

Scenario results: Pod to external performance (1)



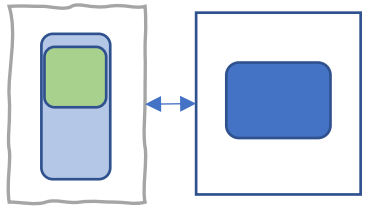
RHEL 8.2 vs. OCP 4.5.13, uperf LPAR (server) to OCP guest (client), z15 z/VM vswitch

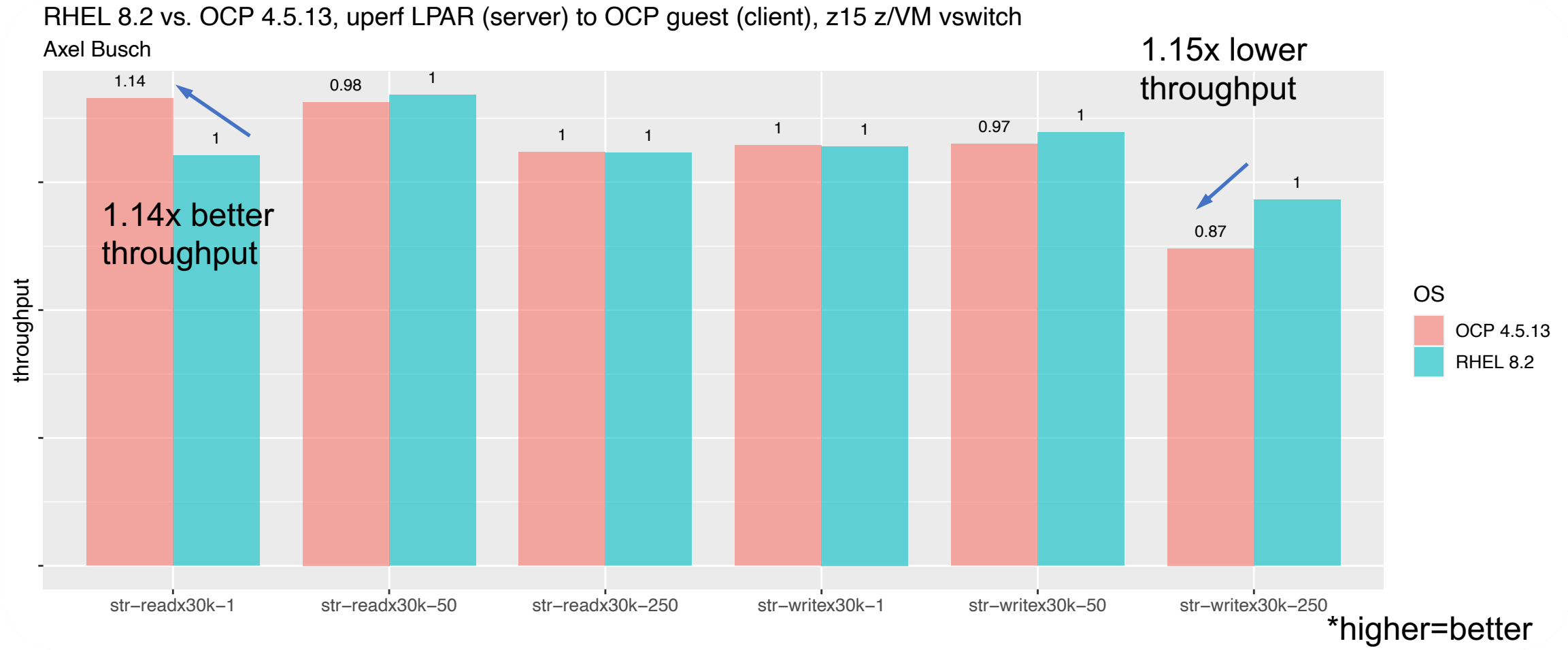
Axel Busch



*lower=better

Scenario results: Pod to external performance (2)

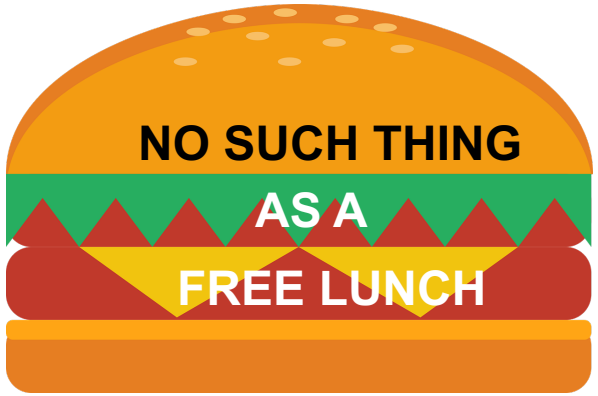




Almost same throughput for most workloads!

Scenario results: Discussion

- Pod 2 Pod performance
  - 1.09 - 1.5x higher latency
  - From 1.2x higher - 1.8x lower throughput compared to podman2podman
- Pod to external service performance
  - For typical workloads only from 10% to 2.75x higher latency, but can be up to 10x higher latency
  - Almost same throughput, up to 1.15x lower throughput compared to RHEL8.2 in z/VM



High automation…

Scalability…

Availability…

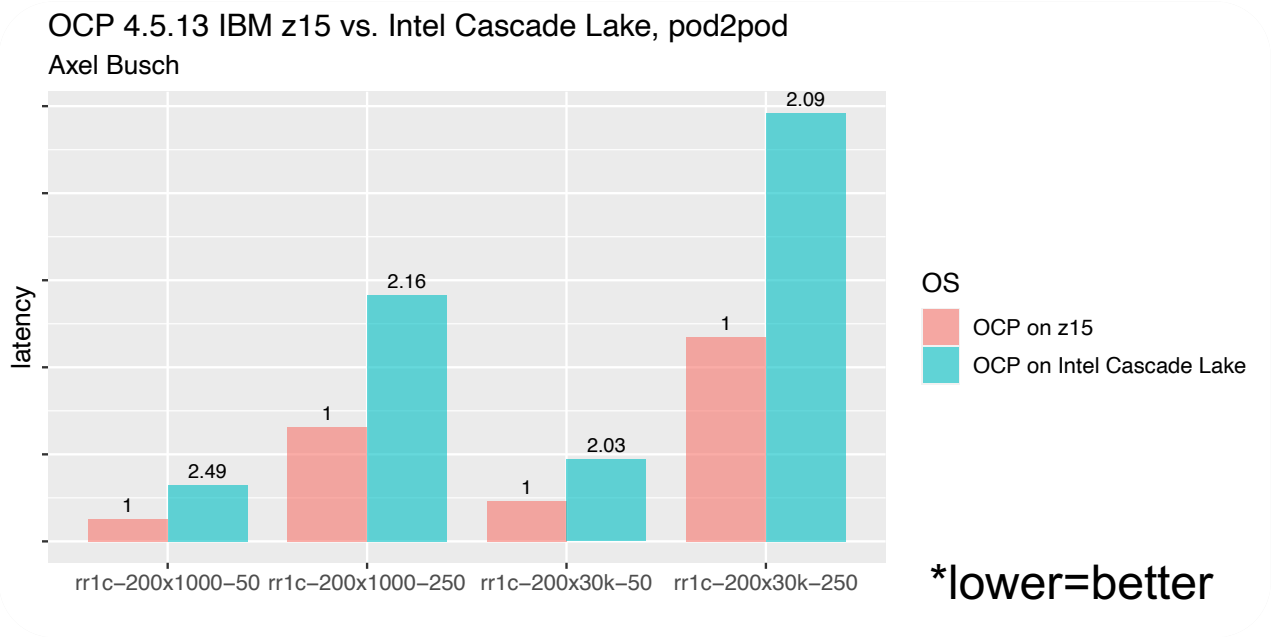
Flexibility…

… does **not** come for **free**…

… but has **improved significantly** and we are **working to improve it further**

Scenario results: Comparison to competitors

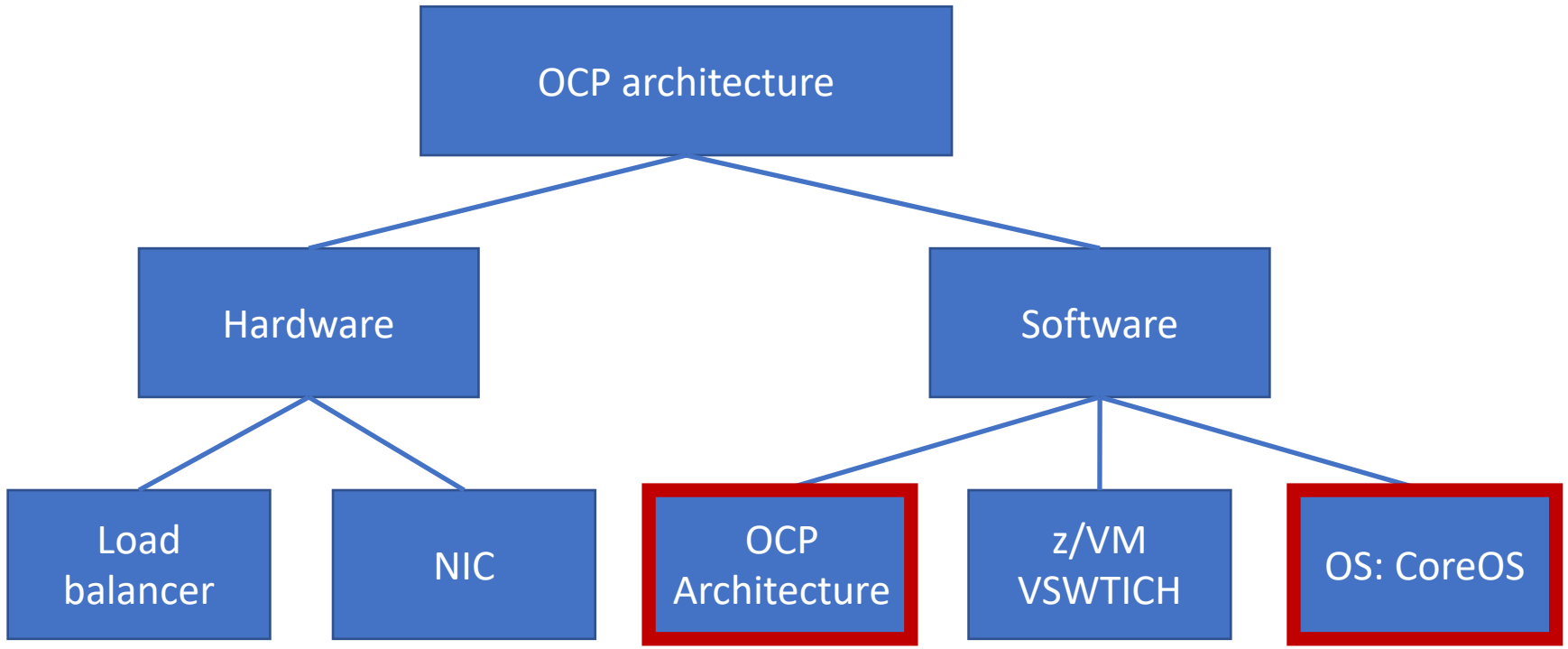
Pod 2 Pod performance on Intel cascade lake cluster up to 1.7x higher latency compared to podman2podman

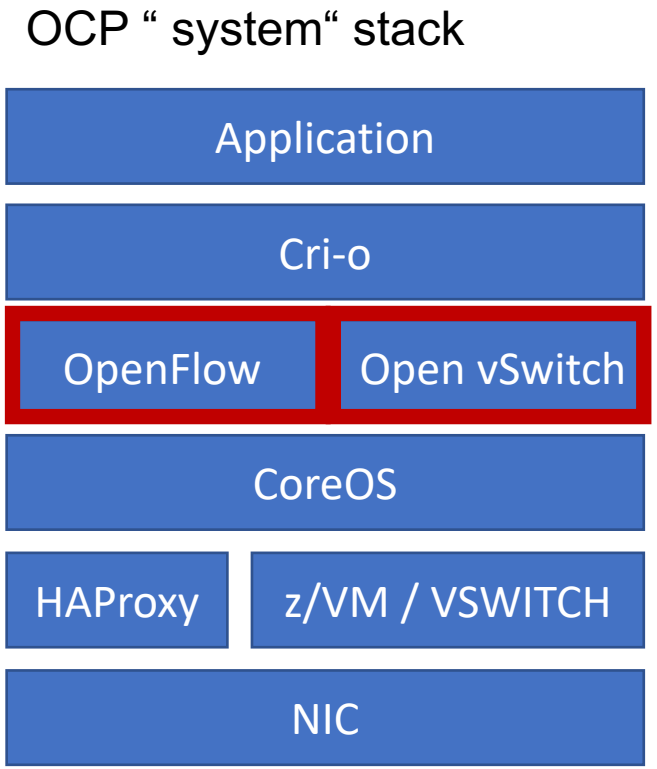


- Up to **2.49x better response times** on IBM z15
- Up to **2.00x better throughput** on IBM z15

Possible reasons

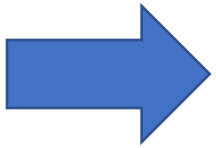
Software & hardware architecture highly influences quality attributes such as performance!





OCP Architecture: Software defined network (SDN)

- Used for (automated) **dynamically configurable network** in changing environments
  - **Physical resources** change according to load
  - **Pods** come and go dynamically
  - (Allowed) **Routes change** during runtime
- Pods (sometimes) need **connection to outside world**
- Reliability, Security, Scalability **QoS guarantees** supported by dynamic networks
- OCP 3.x - 4.6 mainly use open vSwitch, OpenFlow and openshift-node-agent
- OCP (on z) 4.7 uses Open Virtual Network (OVN)

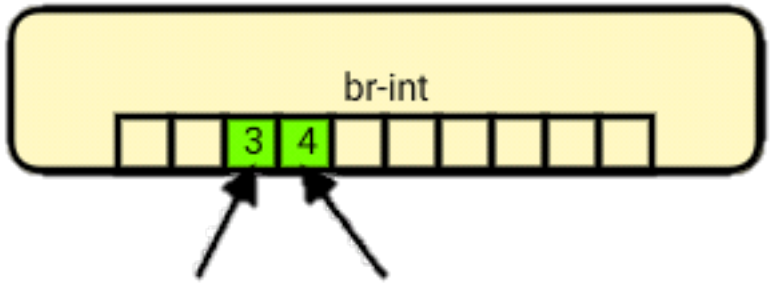


Open vSwitch and OpenFlow main components of OpenShift SDN

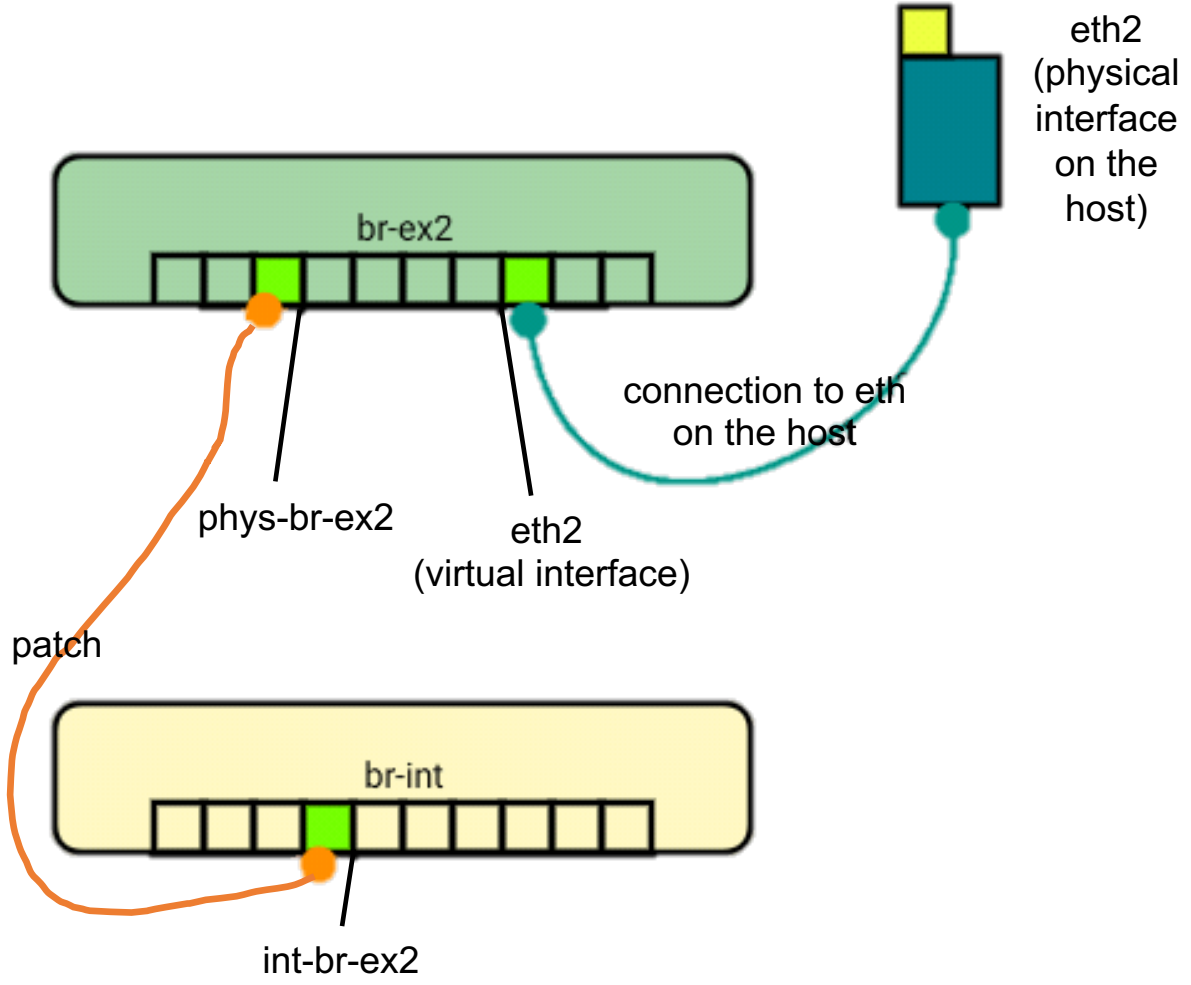
Open vSwitch (OVS)

OPEN vSWITCH
An Open Virtual Switch

- Needed when many virtual machines (or containers) run on one physical node
- Virtualizes network layer
  - Virtual ports
  - Virtual bridges
  - Used for connecting several virtual machines
- Connects virtual machines to physical network

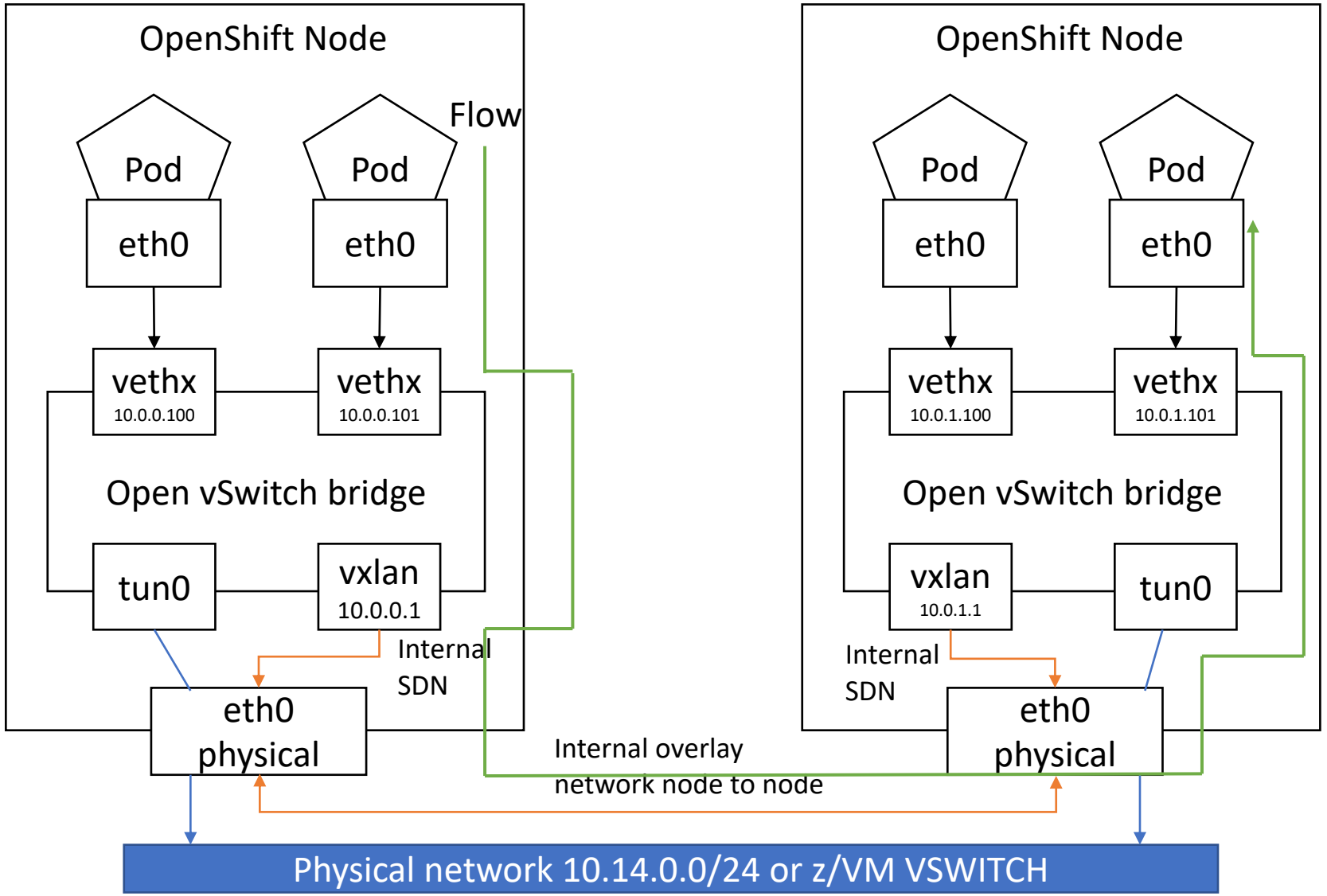


Virtual bridge with its corresponding ports. Each port has a corresponding name and number



see https://superuser.openstack.org/articles/openvswitch-openstack-sdn/

OpenShift software defined network



vethx: virtual interfaces

vxlan: forwards traffic to internal (overlay) network

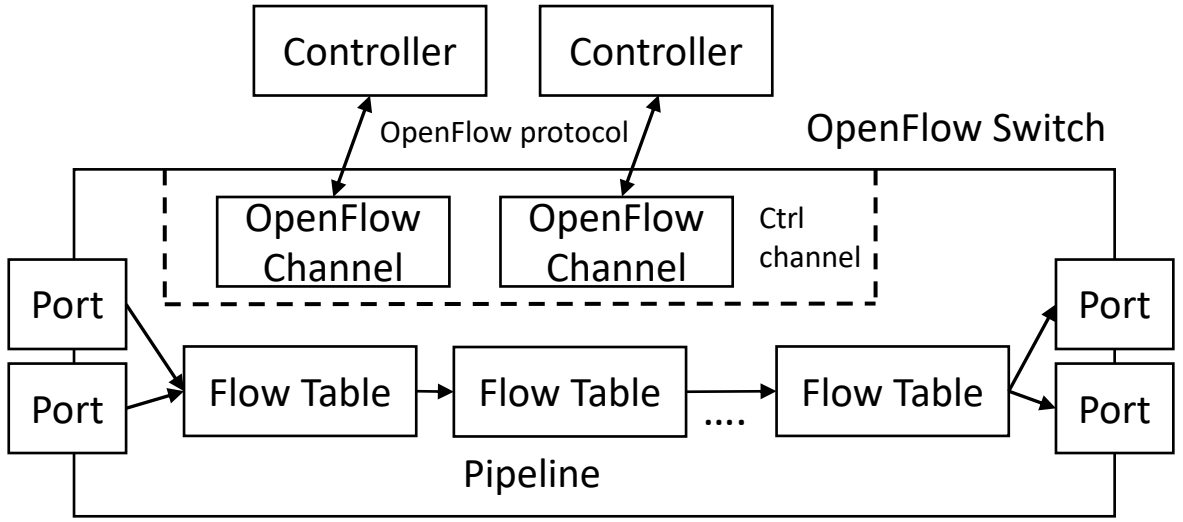
tun0: forwards traffic to external network

Open vSwitch and OpenFlow

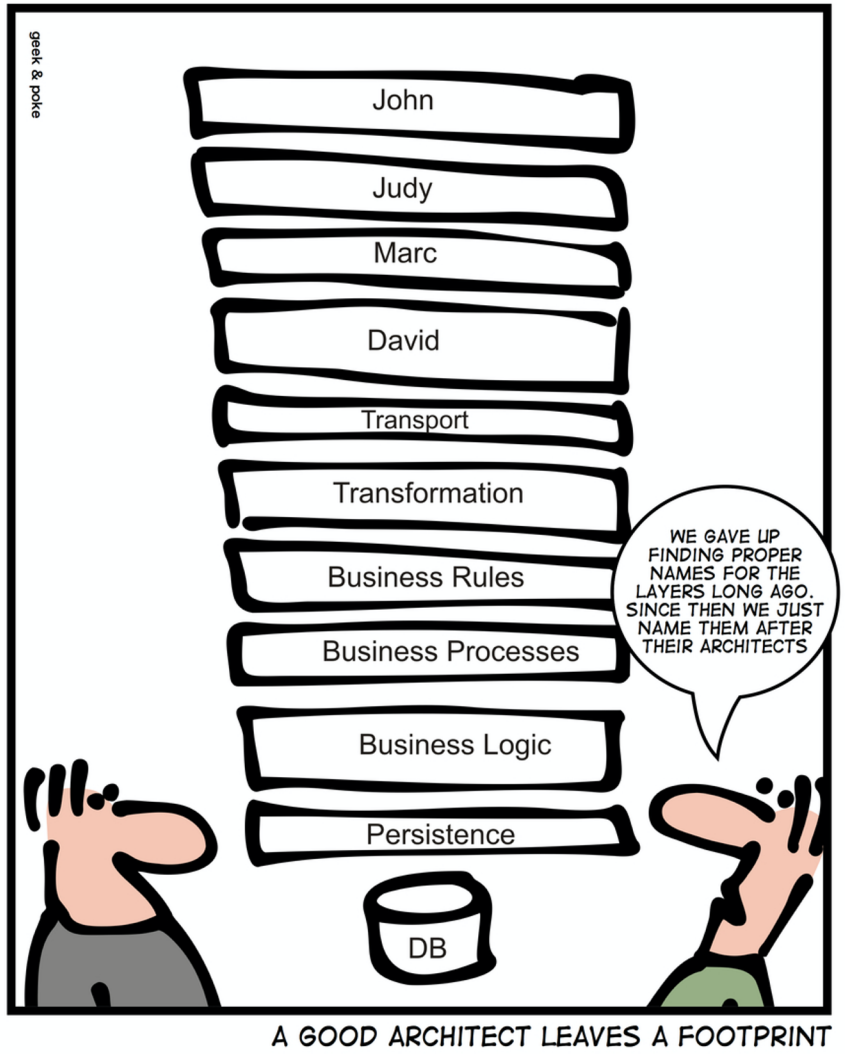
How to define which pod can access other pods and/or external network?

OpenFlow rules control communication between ports on virtual bridges

- OpenFlow separates control of packet flow from packet forwarding
- Integrates functions directly in the network (e.g. firewall)
- Channel controllers configure and manage the switch

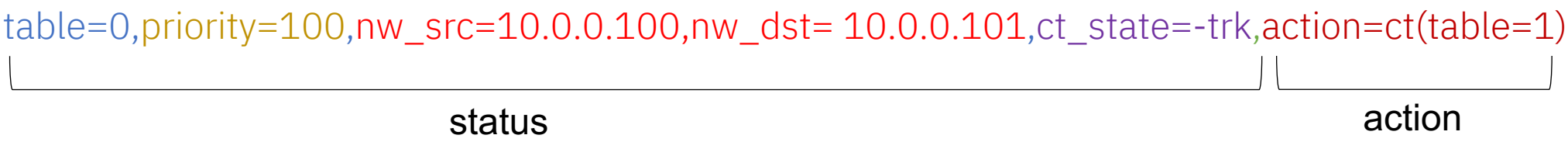


Hang on…



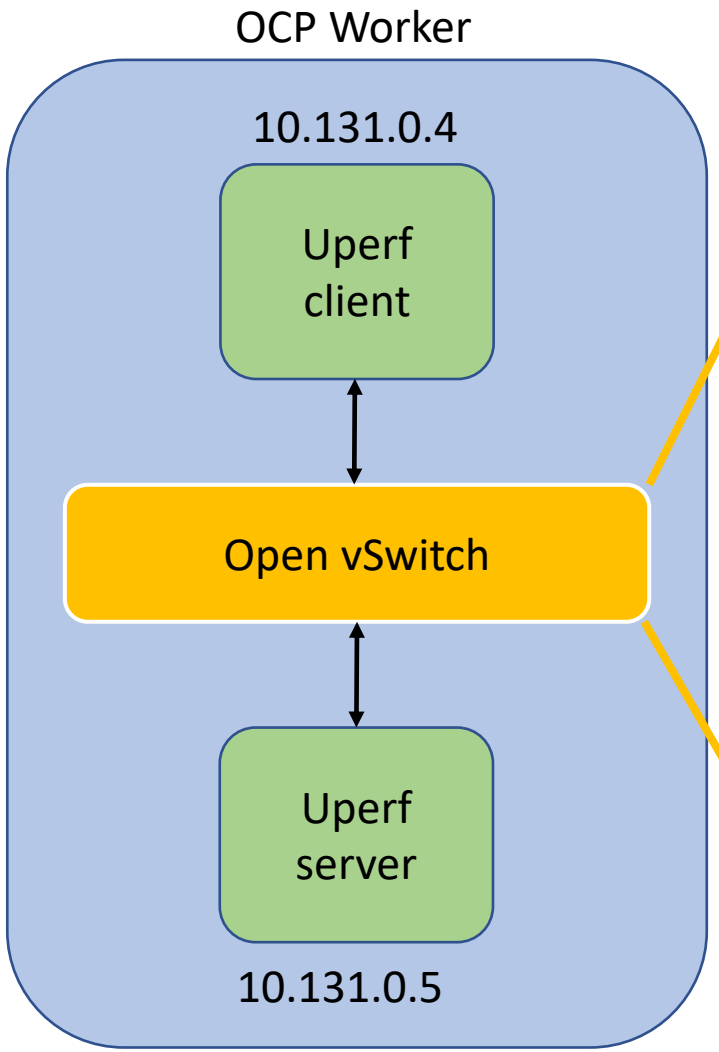
http://geek-and-poke.com/

OpenFlow rules architecture



- Connection tracking (contrac or ct) keeps track of connection states of individual TCP sessions
- ct allows to control packet flows by using ACLs
- Can be used to implement (stateful) firewall
  - Selectively commits some traffic
  - Matches ct states to allow established connections but deny new connections
- Works with tables that realizes pipeline

OVS & OpenFlow architecture OpenShift



Pipeline

Table=0

1. priority=300, ct_state=-trk, ip, actions=ct(table=0)
2. priority=100, ip, actions=goto_table:20

Table=20

1. priority=100,ip,in_port=5,nw_src=10.131.0.4 actions=load:0->NXM_NX_REG0[],goto_table:21
2. priority=100,ip,in_port=6,nw_src=10.131.0.5 actions=load:0->NXM_NX_REG0[],goto_table:21

Table=21

priority=200,ip,nw_dst=10.128.0.0/14 actions=ct(commit,table=30)

Table=30

1. priority=300,ct_state=+rpl,ip,nw_dst= 10.131.0.0/23 actions=ct(table=70,nat)
2. priority=200,ip,nw_dst=10.131.0.0/23 actions=goto_table:70

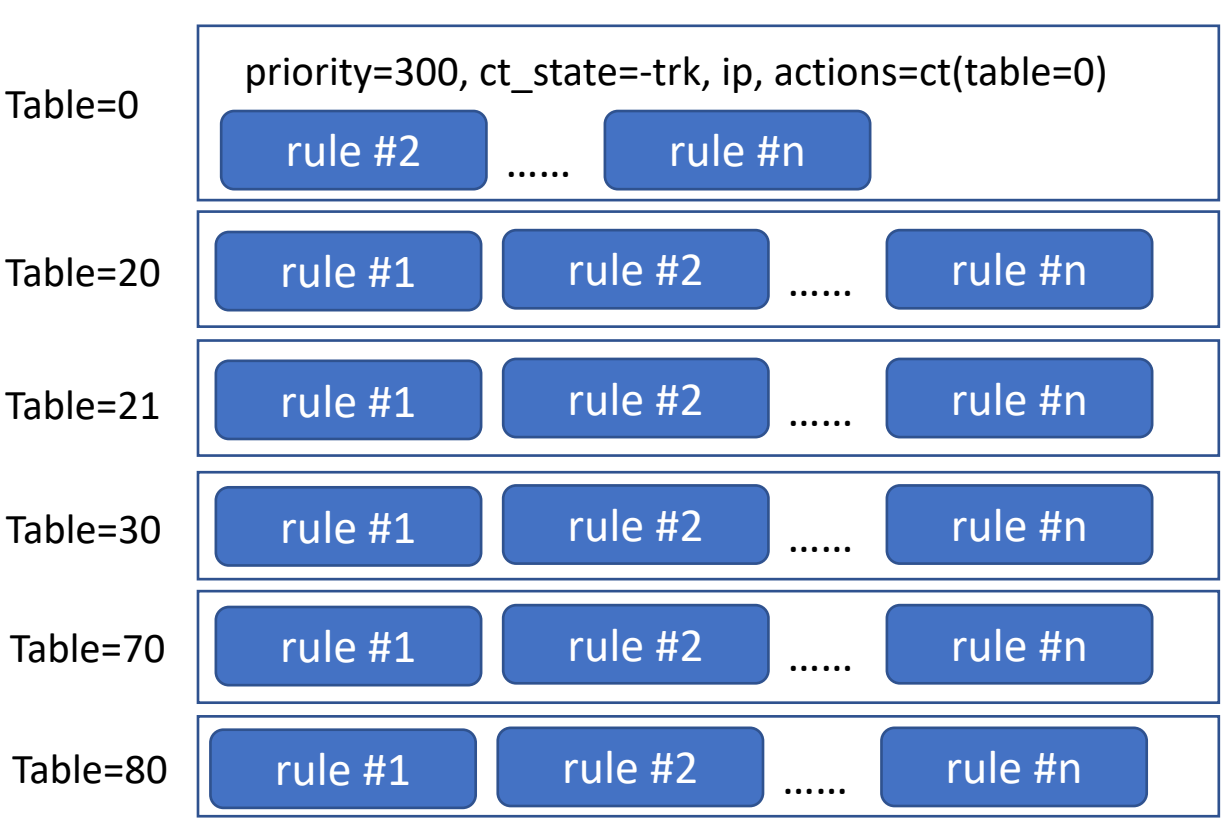
Table=70

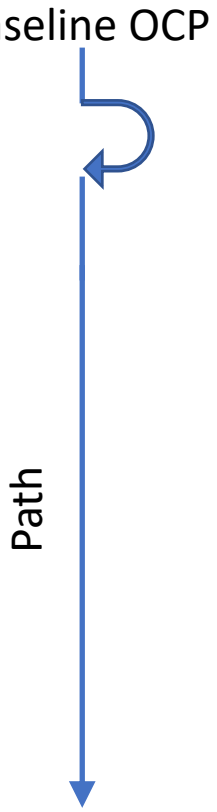
priority=100,ip,nw_dst=10.131.0.4 actions=load:0->NXM_NX_REG1[],load:0x5->NXM_NX_REG2[],goto_table:80

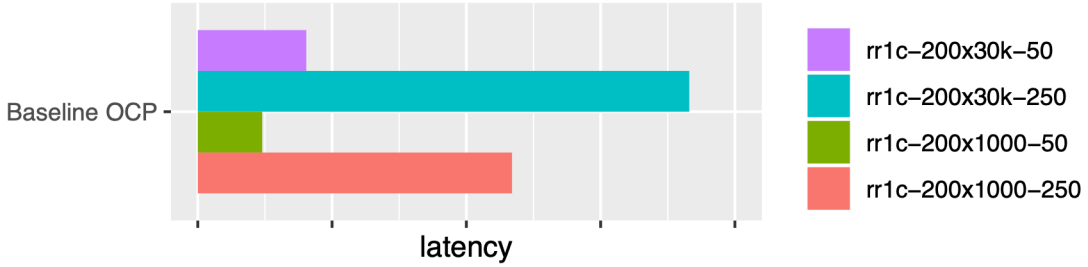
Table=80

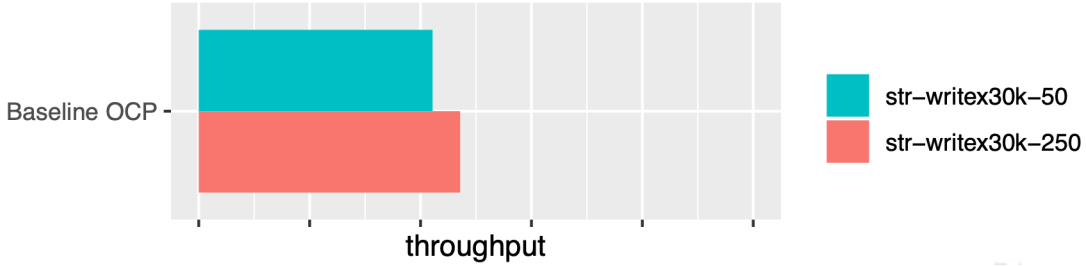
1. priority=200,ct_state=+rpl,ip actions=output:NXM_NX_REG2[]
2. priority=50,reg1=0 actions=output:NXM_NX_REG2[]

OpenFlow rules performance: uperf request response & streaming

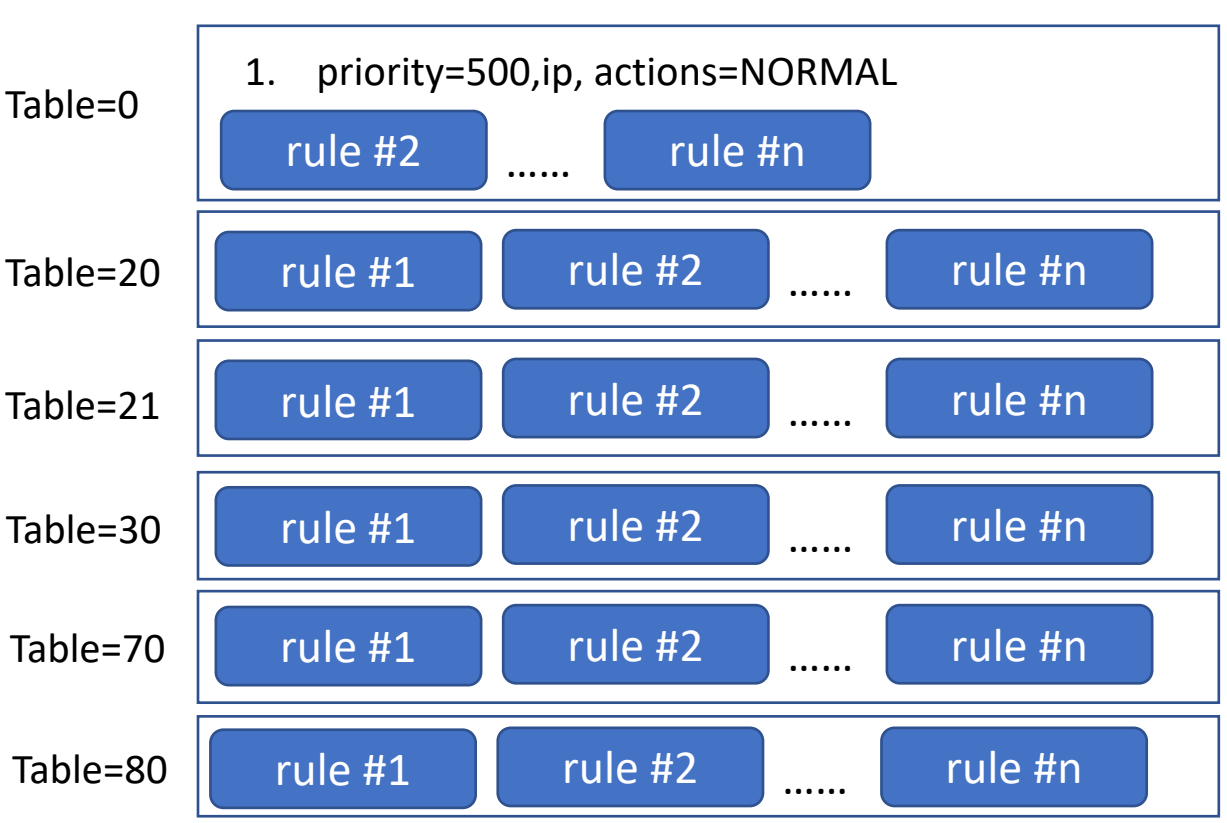


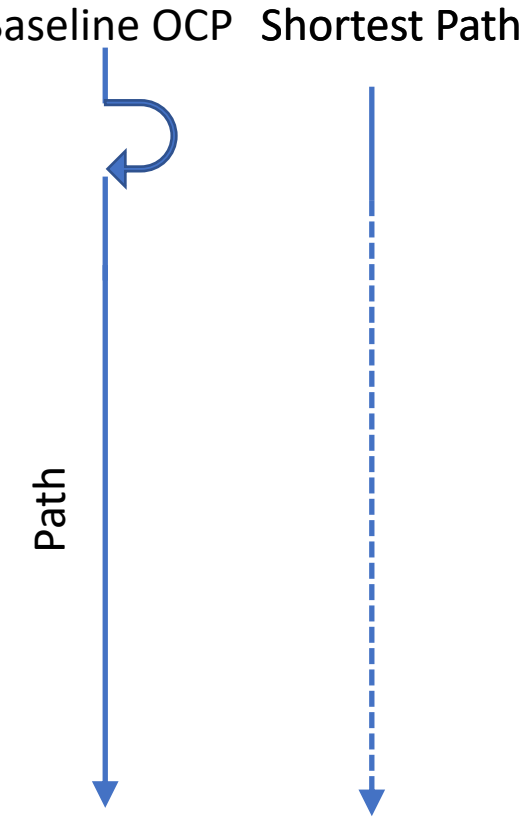


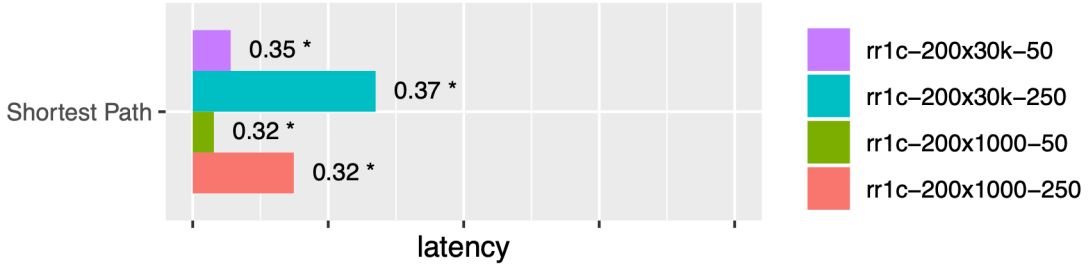


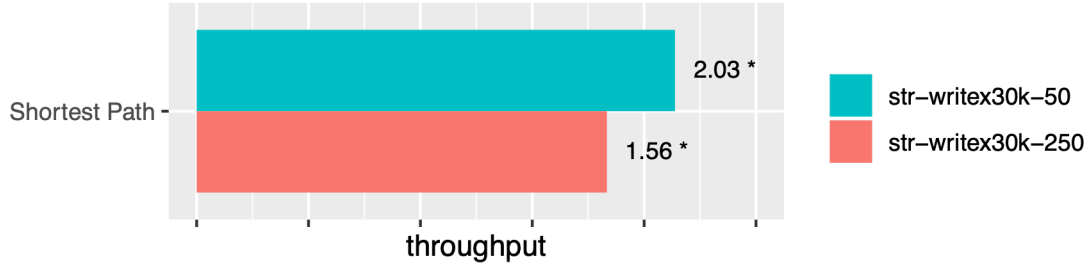


OpenFlow rules performance: uperf request response & streaming



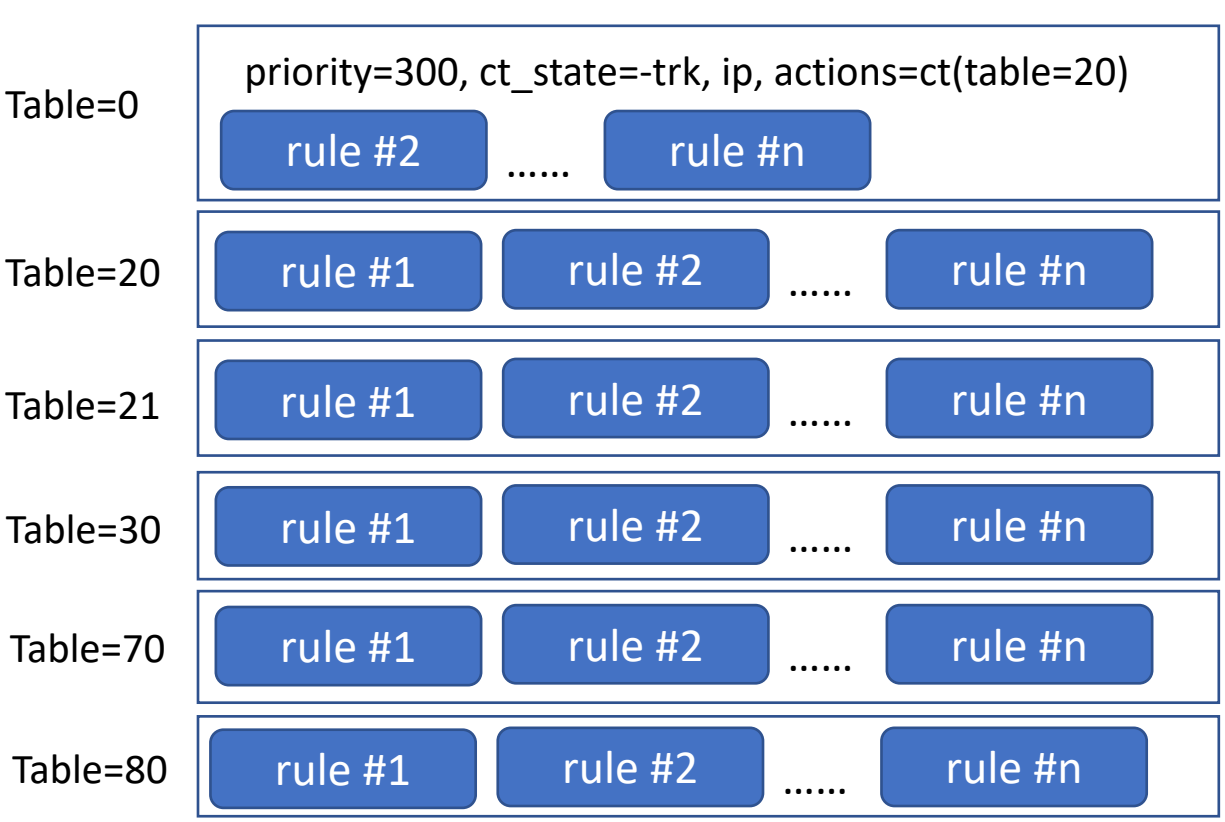


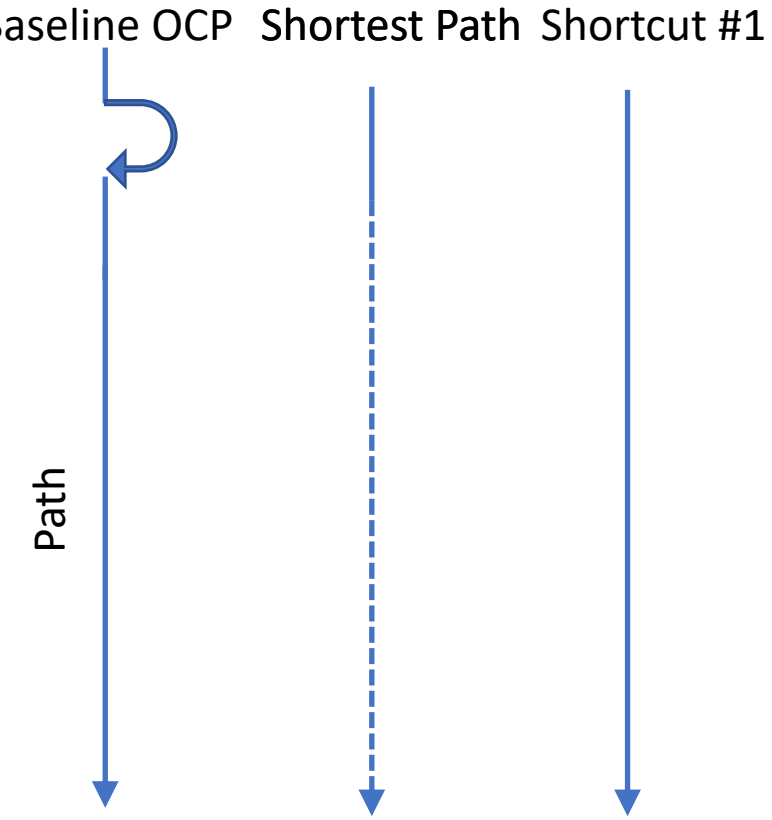


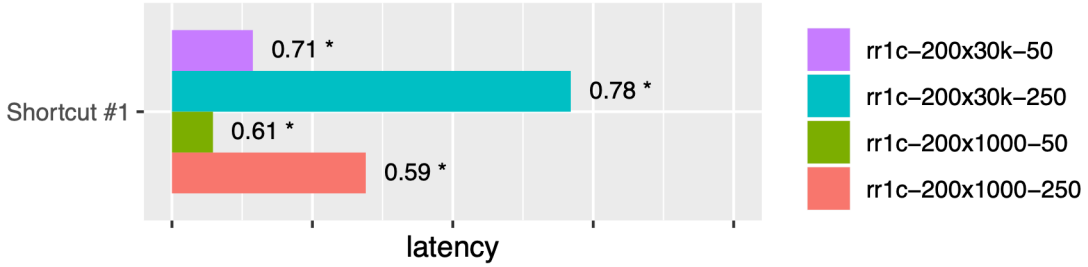


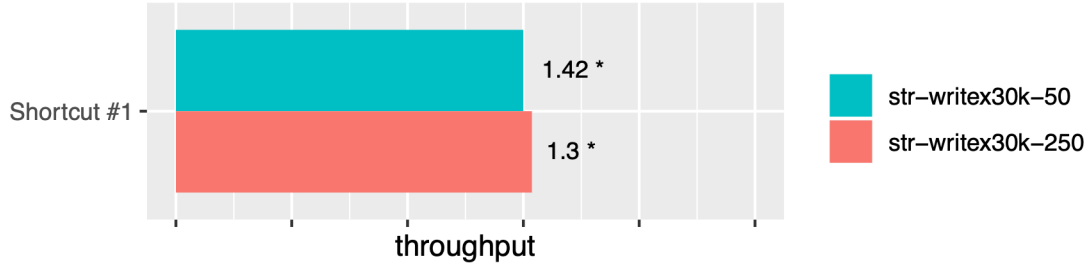
* relative to Baseline OCP

OpenFlow rules performance: uperf request response & streaming



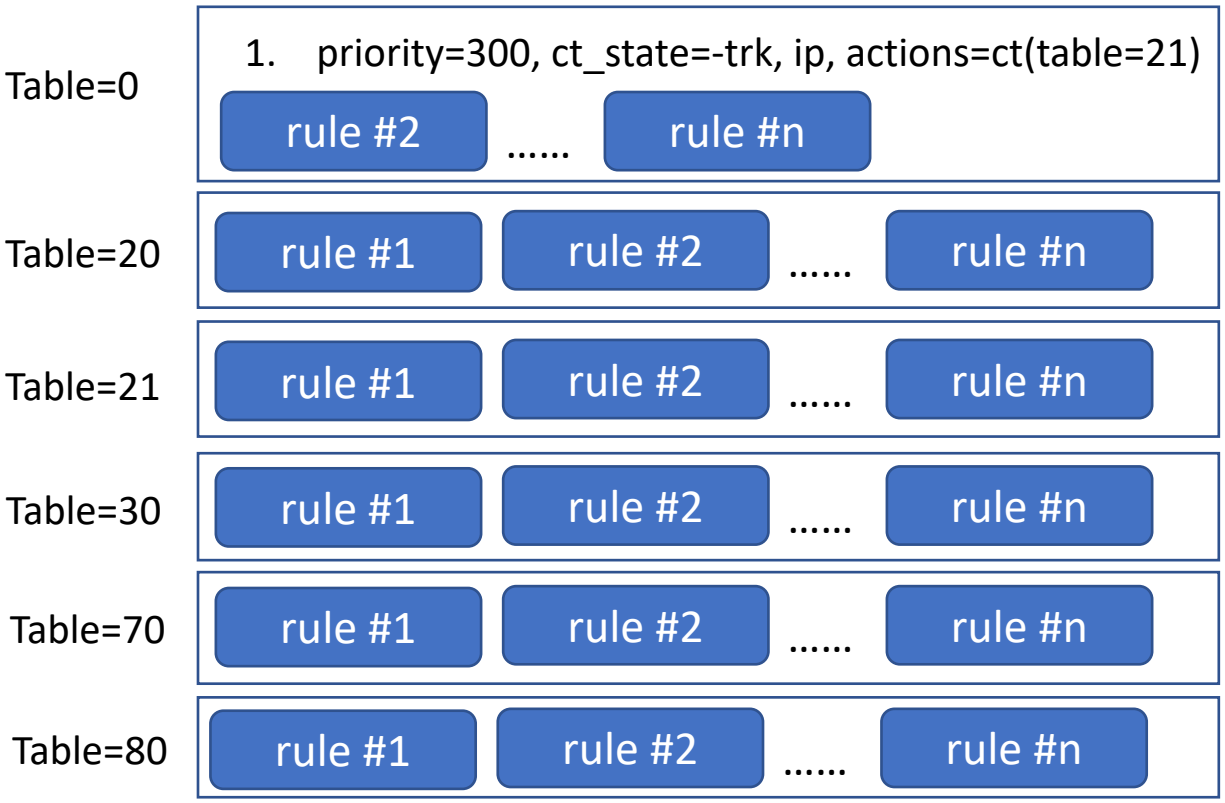


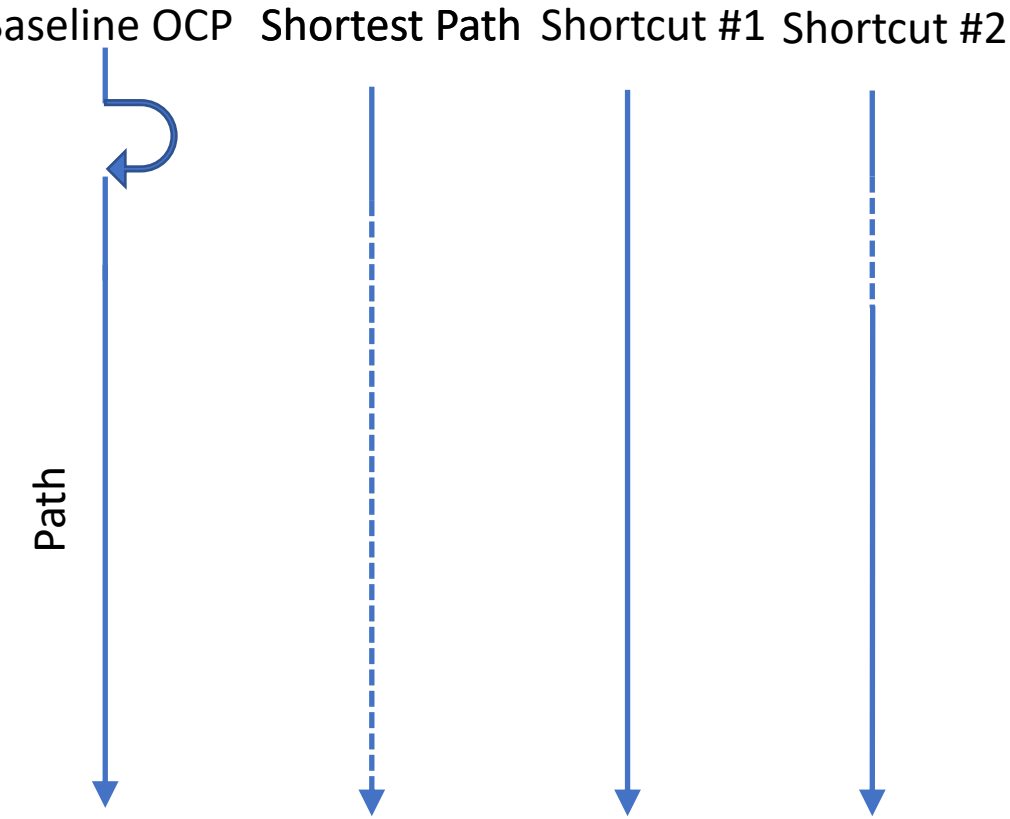


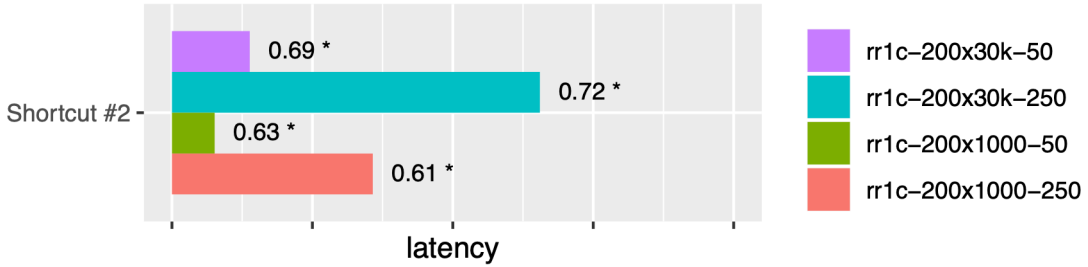


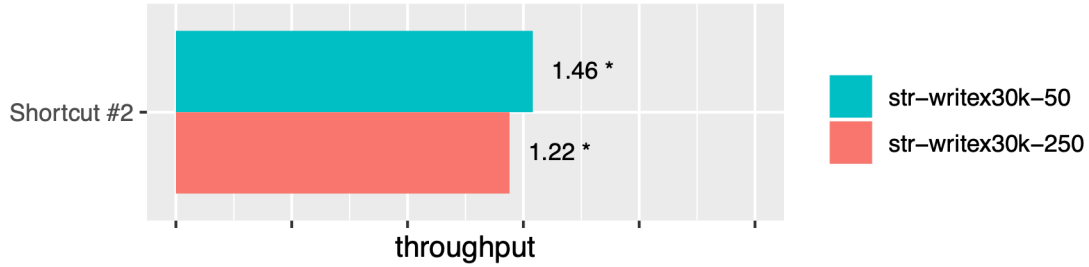
* relative to Baseline OCP

OpenFlow rules performance: uperf request response & streaming



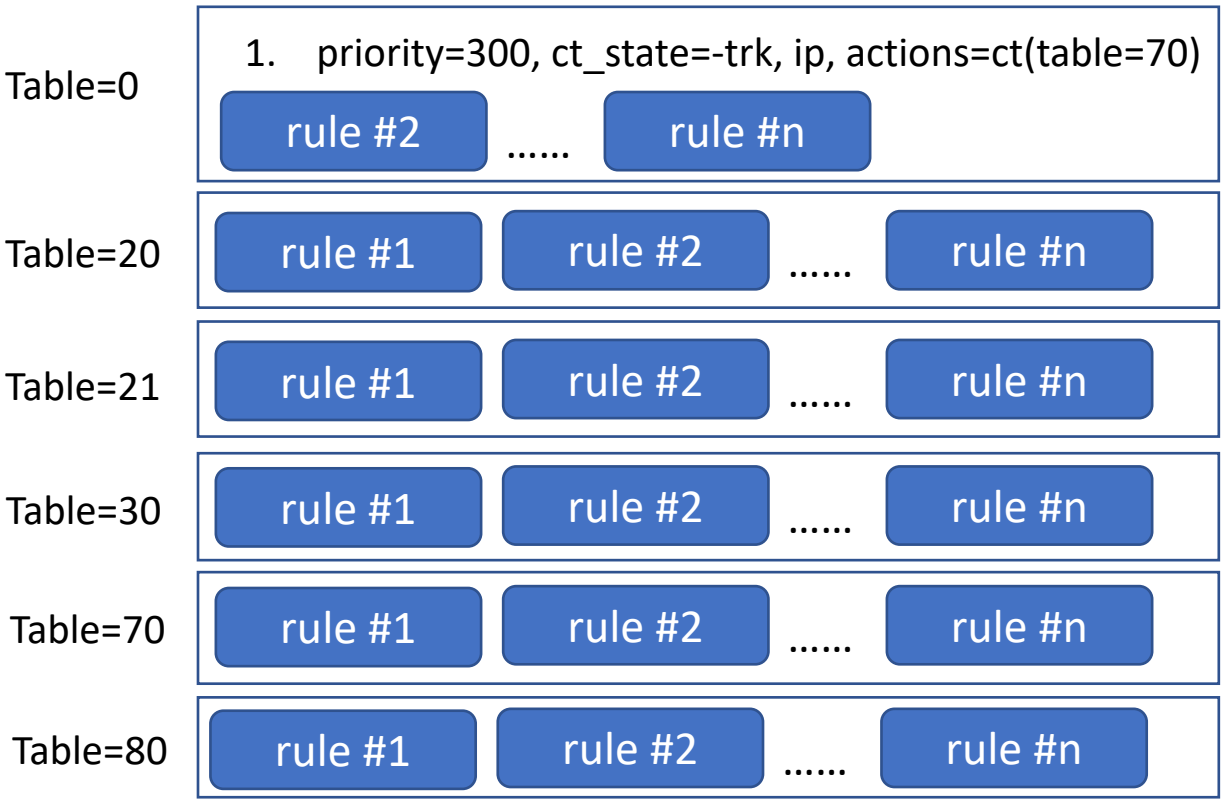


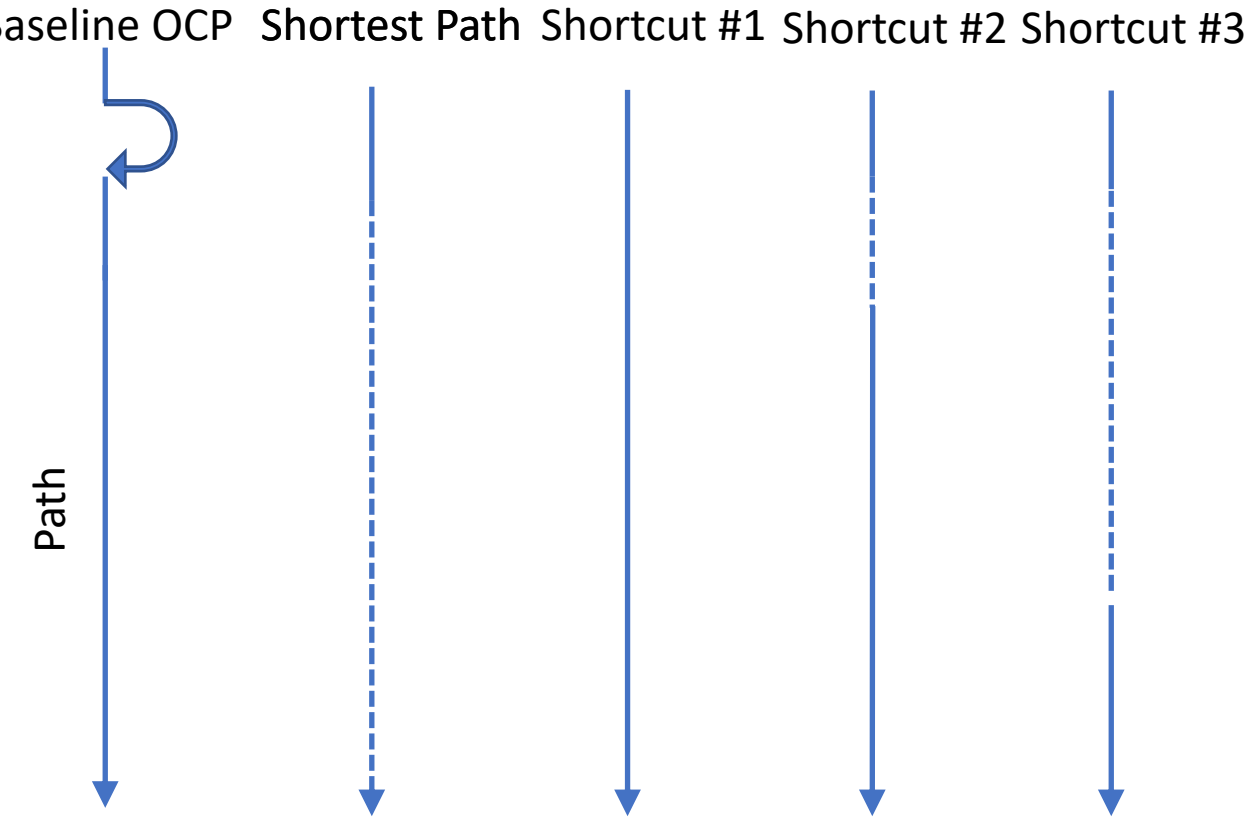


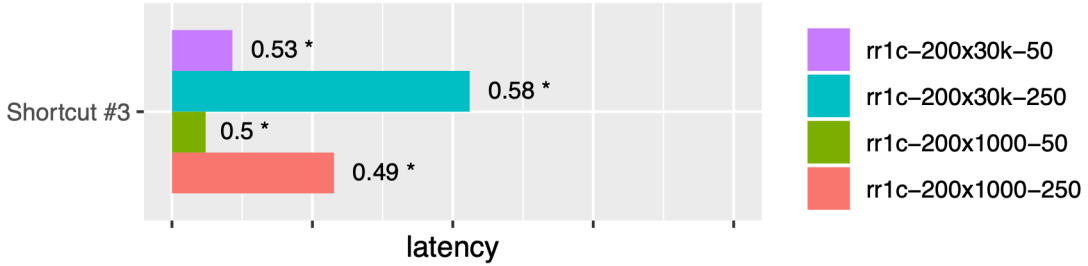


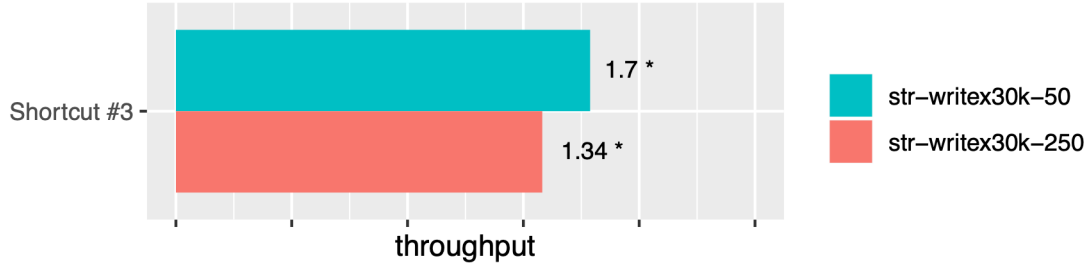
* relative to Baseline OCP

OpenFlow rules performance: uperf request response & streaming



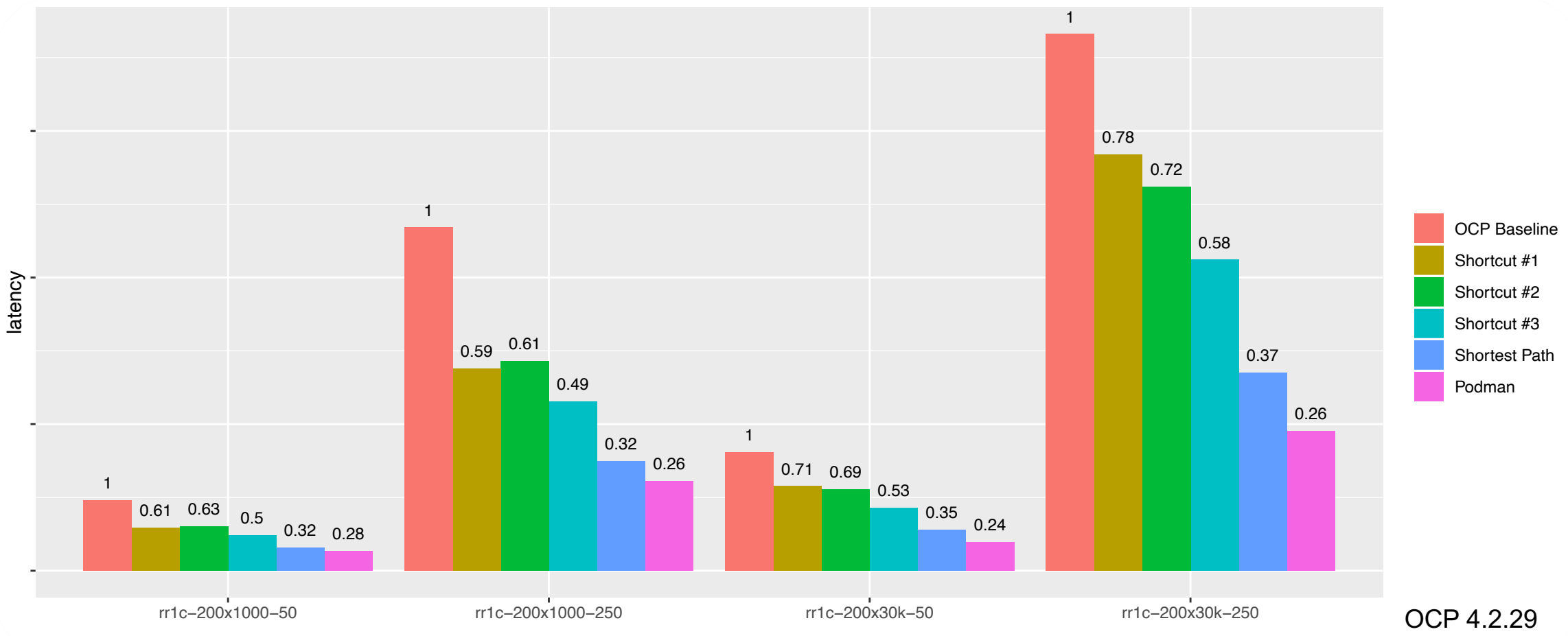






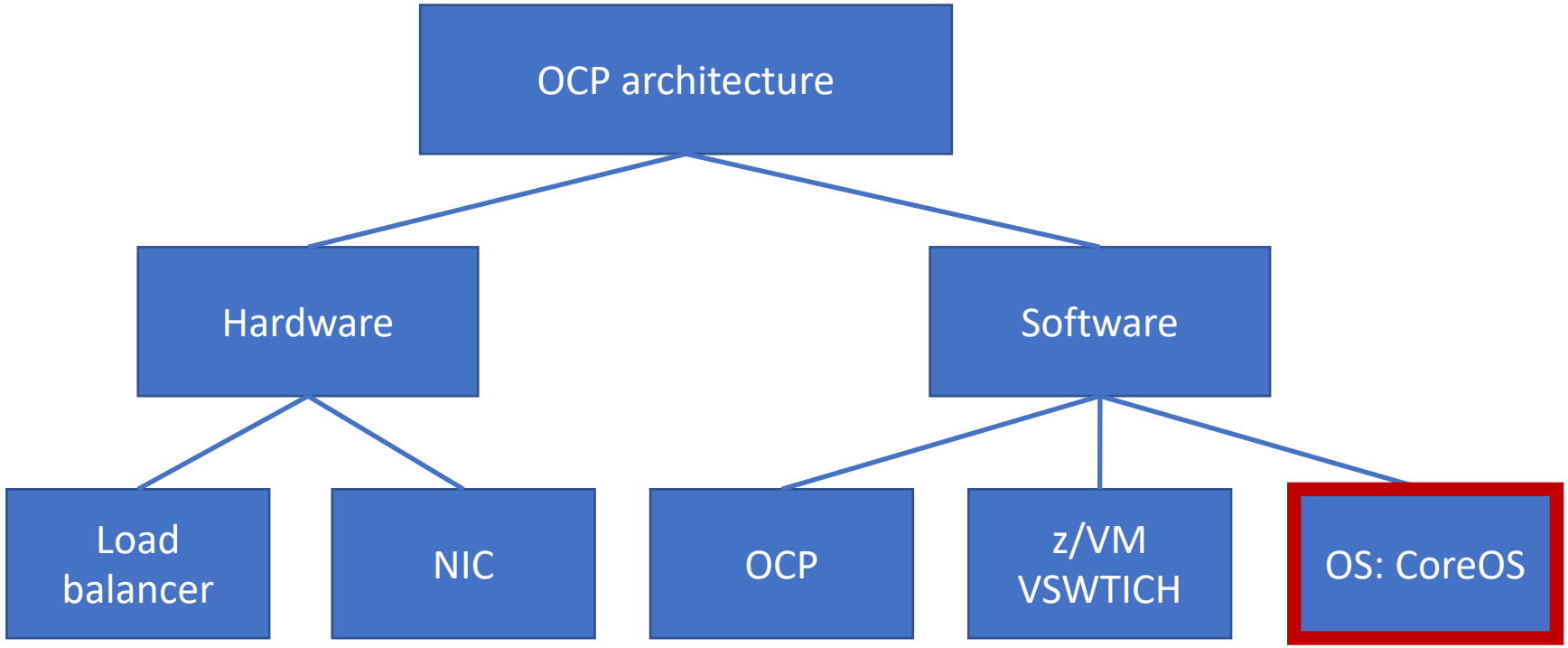
* relative to Baseline OCP

OpenFlow rules performance: Summary



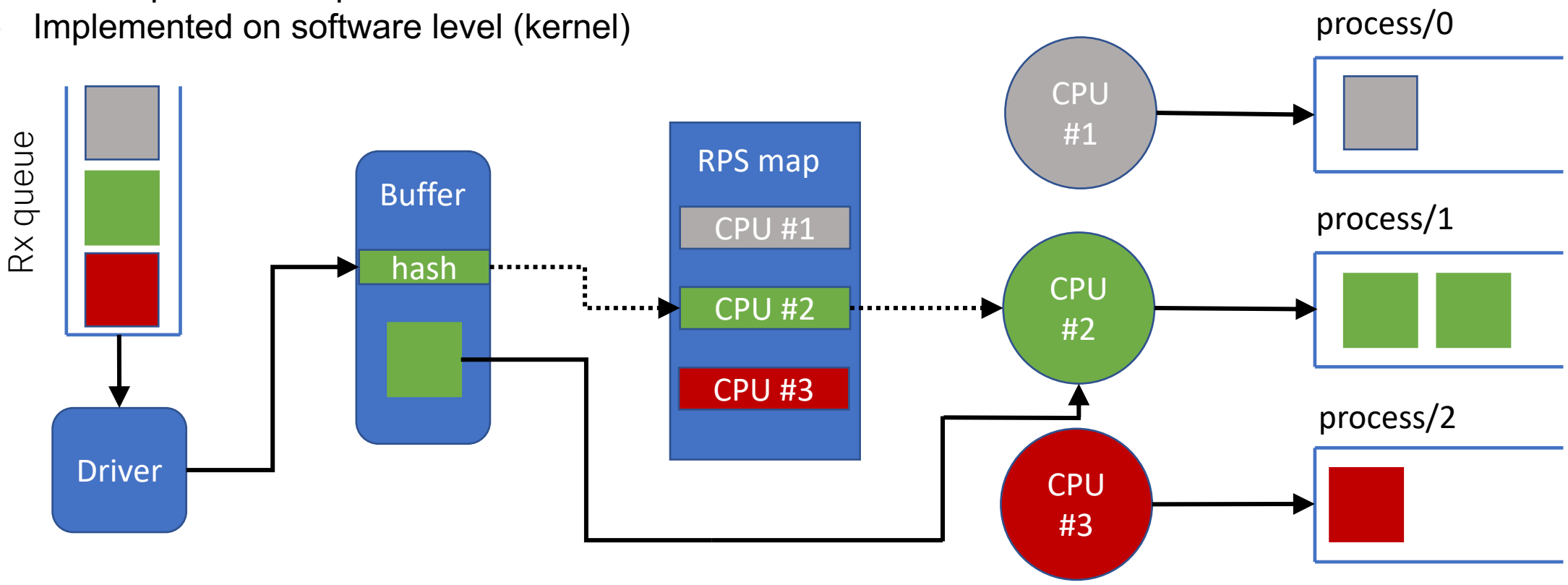
OCP 4.2.29

Performance Optimizations



CoreOS/OCP optimizations: Receive Packet Steering (RPS)

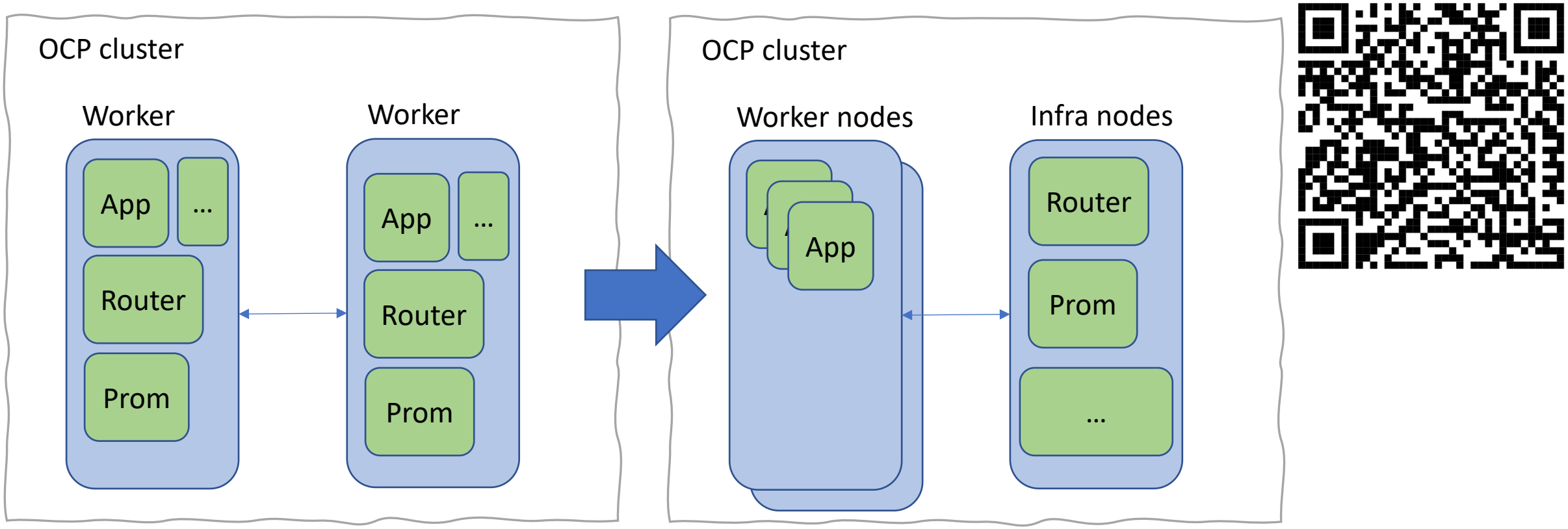
- Prevents hardware queue of network card beeing bottleneck
- Directs packets to specific CPUs
- Implemented on software level (kernel)



Latency can be improved significantly with RPS setting on (depending on workload).

CoreOS/OCP optimization results: Using infrastructure nodes

How to setup infra nodes:

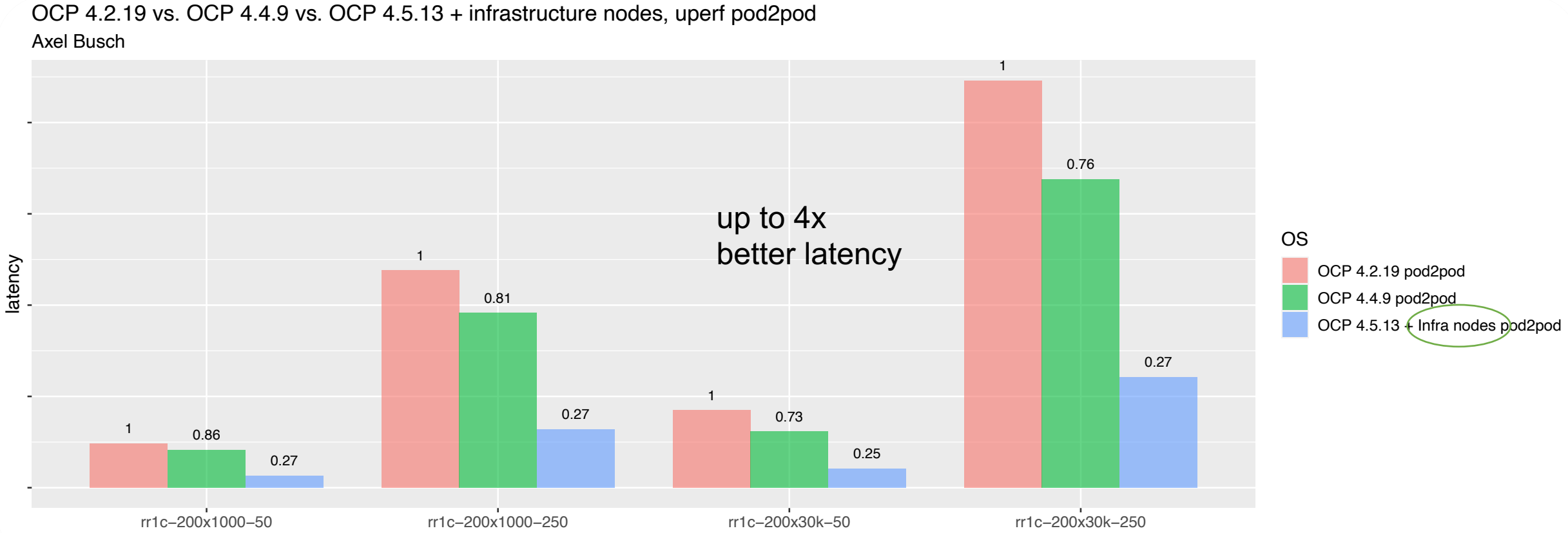


- Move all infrastructure services to infrastructure nodes to keep worker for application workload exclusive
- Consume resources of worker and slow down applications

- Move all infrastructure services to infrastructure nodes to keep worker for application workload exclusive
- Consume resources of worker and slow down applications

https://www.linkedin.com/pulse/boosting-performance-using-infrastructure-nodes-your-cluster-miranda/

CoreOS/OCP optimization results: update on OCP 4.5.13 and use of infra nodes



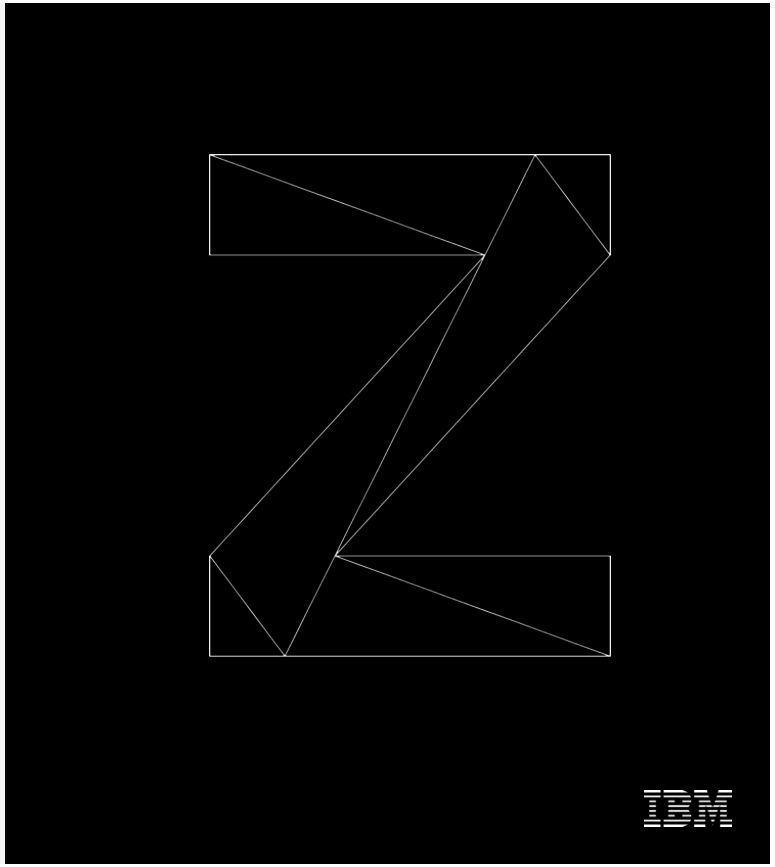
Further optimizations

Updates on
- Receive Packet Steering (RPS)
- Static routes
- Infrastructure nodes

Red Hat OpenShift on IBM Z - Performance Experiences, Hints and Tips

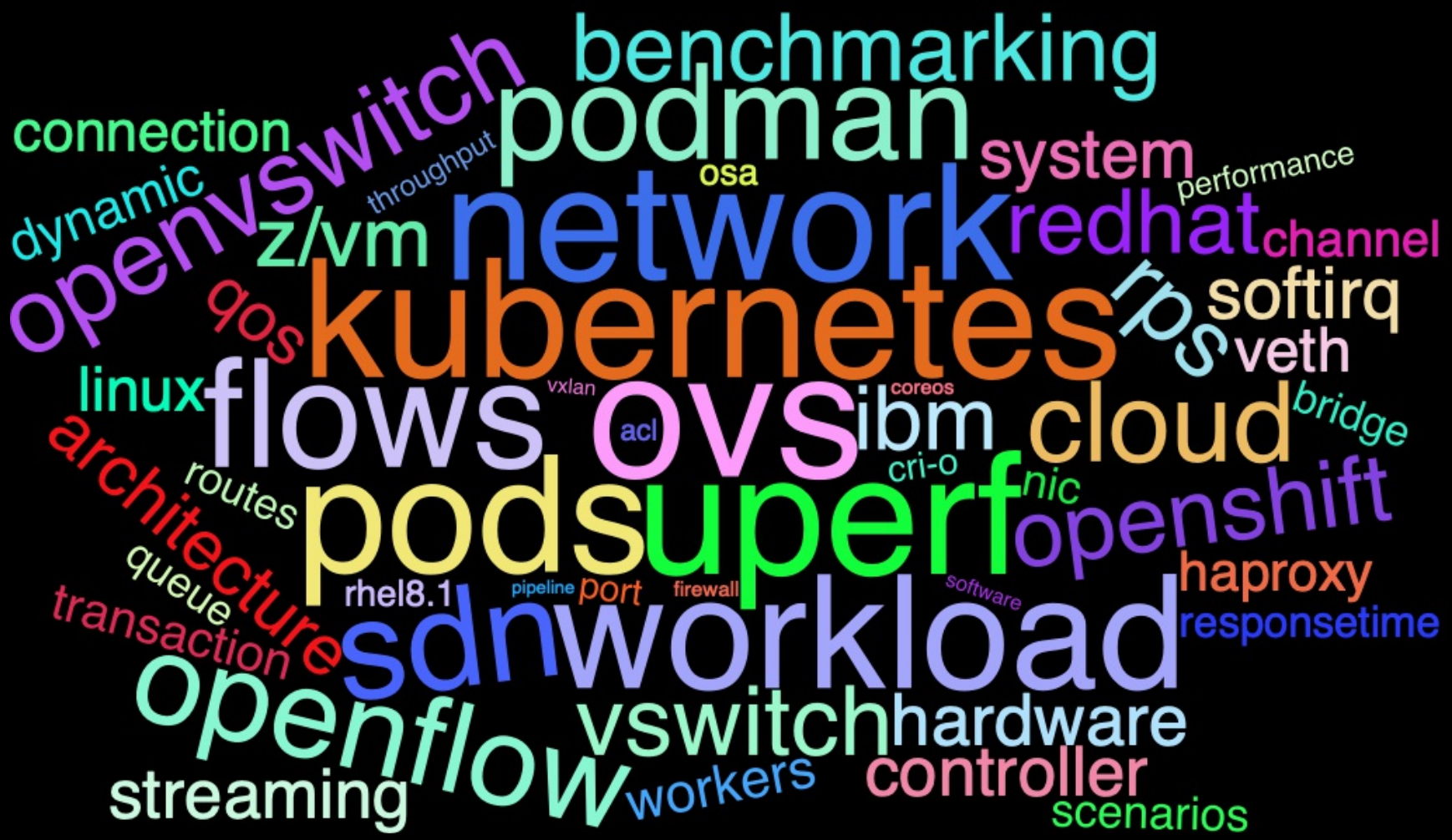
Marc Beyerle (marc.beyerle@de.ibm.com)
Senior Java Performance Engineer, IBM Mainframe Specialist

Document version: 1.1
Document date: 2020-10-20



http://public.dhe.ibm.com/software/dw/linux390/perf/OpenShift_on_IBM_Z_-_Performance_Experiences_V11.pdf

Summary


benchmarking
podman
openvswitch
connection
dynamic
throughput
osa
system
performance
network
z/vm
redhat
channel
qos
kubernetes
rps
softirq
veth
linux
flows
ovs
vxlan
coreos
ibm
cloud
bridge
acl
cri-o
nic
architecture
routes
podsuperf
openshift
haproxy
queue
rhel8.1
pipeline
port
firewall
software
responsetime
transaction
sdnworkload
openflow
vswitch
hardware
streaming
workers
controller
scenarios

Thank you!

Dr.-Ing. Axel Busch
Linux & OCP Performance Analyst
axel.busch@ibm.com
Linux on IBM z Performance

ibm.biz/perfradar – the Performance Radar Blog