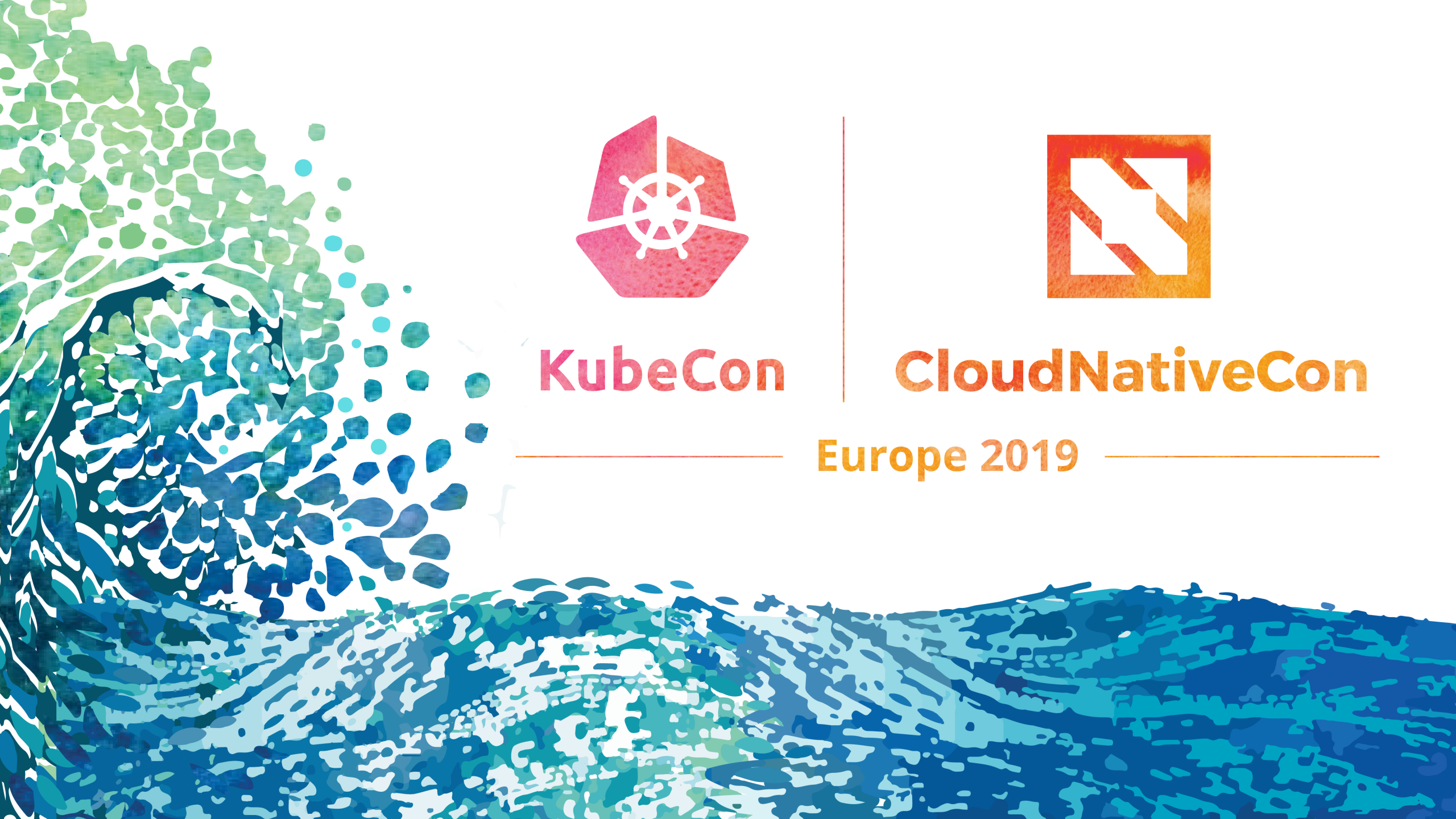

KubeCon
CloudNativeCon
Europe 2019

ebay

KubeCon | CloudNativeCon

Europe 2019

Running high-performance workloads at scale with k8s

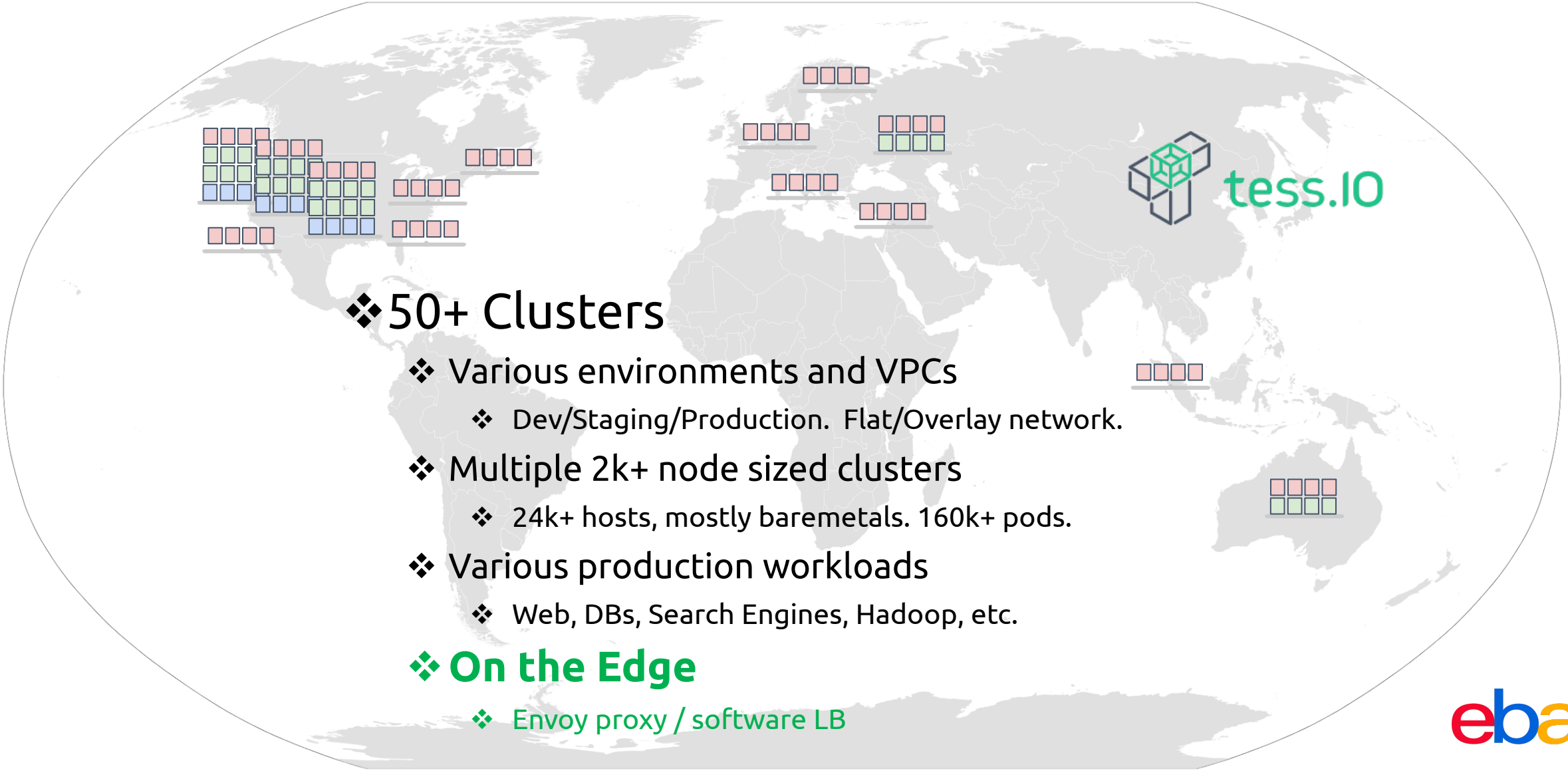
eBay's best practices + lessons learned.

Xin Ma *xima@ebay.com*

Agenda

- eBay's k8s deployments
- Build, Run, and Manage high-performing k8s clusters
  - Build k8s with k8s, at scale (eBay's fleet management system based on k8s)
  - k8s control plane performance
- Running high-performance workloads in k8s
  - Containers and Pod specs
  - Host Runtime performance
  - Network performance
- Summary, what's next for us, Q&A

eBay's k8s deployments



- 50+ Clusters
  - Various environments and VPCs
    - Dev/Staging/Production. Flat/Overlay network.
  - Multiple 2k+ node sized clusters
    - 24k+ hosts, mostly baremetals. 160k+ pods.
  - Various production workloads
    - Web, DBs, Search Engines, Hadoop, etc.
- **On the Edge**
  - Envoy proxy / software LB

ebay

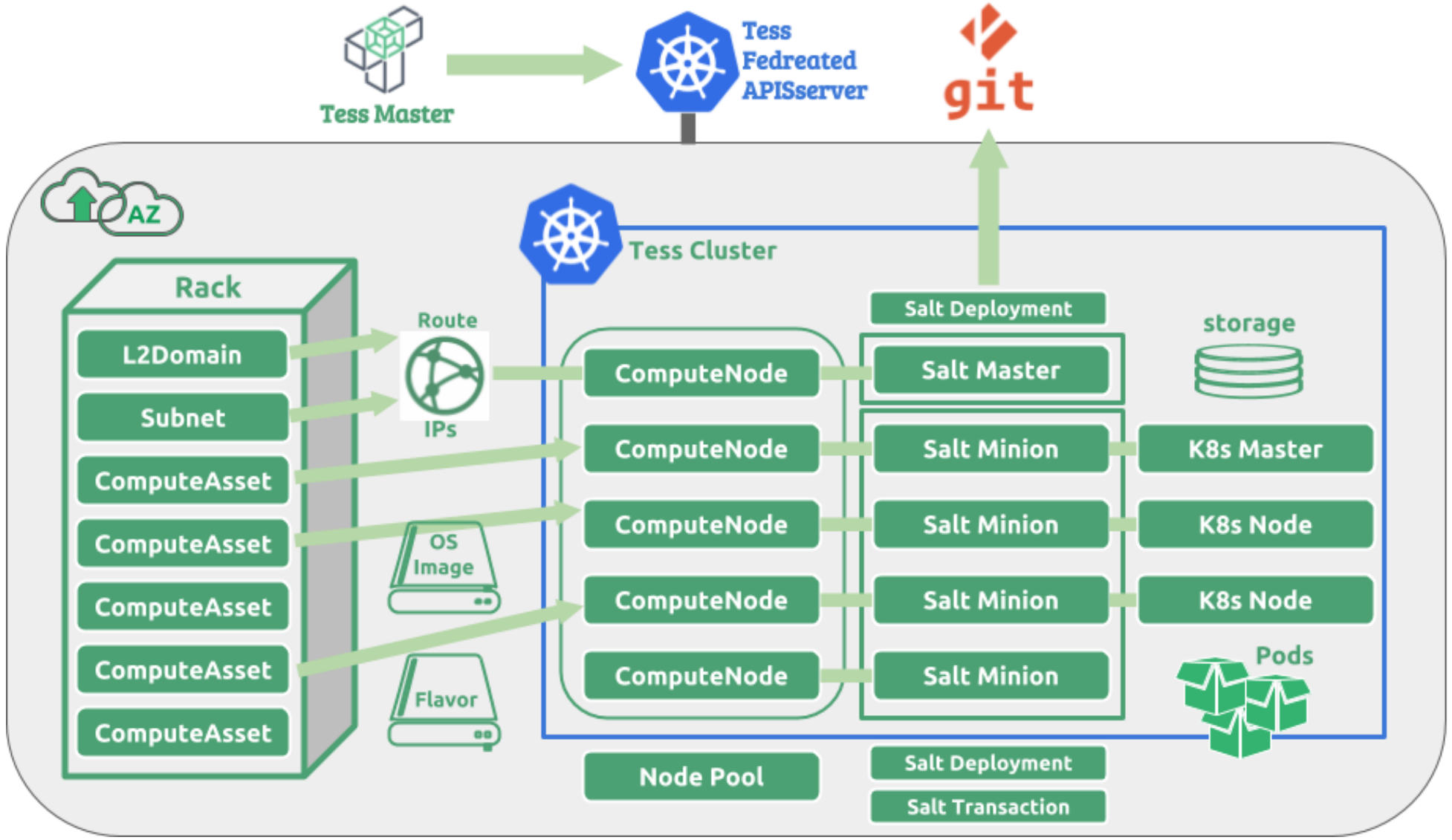

ebay
KubeCon
CloudNativeCon
Europe 2019
Build, Run, & Manage
high-performing k8s clusters

Build, Run and Manage k8s clusters

- Fleet management system
  - CRDs
    - Models the Datacenter. *Racks*/*Switches*/*Subnets*, etc.
    - Models the *OSImages*, *ComputeNodes*, k8s nodes, and *k8sClusters*.
  - Controllers
    - Provision *ComputeNodes* like pods, create *NodePools* like deployments
    - Create and install *SaltMaster* on top of a ComputeNode, from git
    - Install k8s at computenodes, with *SaltMinions*, just like creating pods
    - Install and manage multiple kube nodes with *SaltDeployments*, just like deployments
    - Transactions, scheduler and rolling update strategies, etc.
- GitOps

Deep Dive at the upcoming Shanghai KubeCon.

Build and Manage k8s, with k8s.



Control Plane Performance

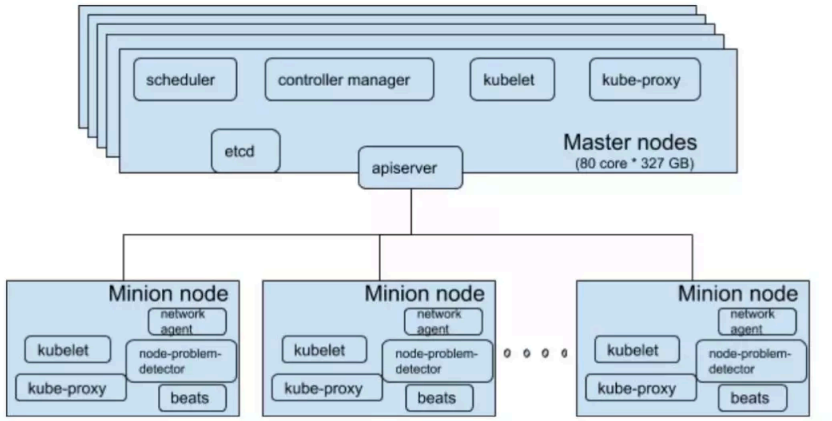
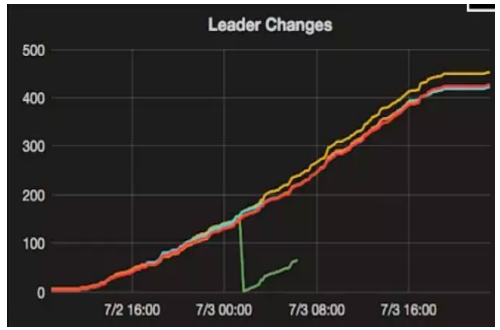


Figure 1. Tess.IO cluster architecture



- k8s Control plane architecture
  - 5 Master nodes. active-active as an etcd cluster
    - Apiserver, local etcd, KCM, and scheduler
  - Kube Nodes
    - kubelet & kube-proxy
    - Daemsonsets / add-ons for cni, storage, monitoring, etc.

- performance challenges (running 5k node with 100k+ pods)
  - Heavy LIST and Watcher API calls
    - etcd
      - High memory %
      - Frequent leader election changes
    - scheduling: delay
    - apiserver frequent crashes & restarts.

ebay

Control Plane Performance

- Benchmarking
  - Kubemark simulating 5k nodes.
  - Test cases.
    - creating + deleting 10k pods in parallel
    - get to know the cluster's limits - max # of lists, watchers, etc.
- eBay's performance practices to run large k8s clusters
  - apiserver & etcd
    - evenly distribute load to 5 apiservers
    - dedicated ssd for etcd – guaranteed iops
      - separate drives for etcd data and etcd snapshots
    - split events
    - Increase *max-mutating-requests-inflight*
    - rate limiting
  - Writing good controllers
    - List option & resource version - apiserver cache or directly hit etcd
    - Use informers
    - Build-in metrics to measure controllers performance
  - cluster lifecycle management
    - Bad nodes and terminating pods – slow down scheduling.
      - Enhanced *FilteredList*
      - Clean up bad node and terminating pods
        - Fleet management system – node lifecycle management

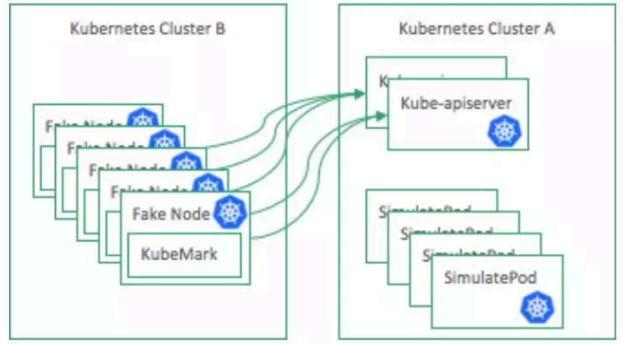


Figure 2. Tess.IO architecture

Running high-performance workloads in Kubernetes

Containers & Pod spec

- Build containers (containerization)
  - native first, and/or do it smart
    - Split services into multiple containers, and run sidecars
  - systemd?
    - It's ok to run systemd but it's a bit too much…
      - Lesson learned: #5795 systemd before v234 with fixed 65k *RLIMIT_NOFILE*
    - Use supervisord, dumb-init, or write your own init script
      - Logging: volume or stdout ?
  - Be careful about capabilities you give…
    - Lesson learned: imagine a container with *tuned* getting *SYS_ADMIN* cap
  - Container resource limits - Java's thread issues.
    - new jdk, or LXCFS with cpuset
- Pod Spec
  - Pass pod info into containers - *downwardAPI*
  - Burstable pods for over-commitment
  - Use *probes*
    - example: enable & disable traffic
  - *emptyDir* for ephemeral
    - Only option, not a perfect one. e.g. size limit concerns
    - CSI Inline Volume Support #596
  - Stateful pods
    - Statefulset with high-performance local volume

```
volumes:
- downwardAPI:
    defaultMode: 420
    items:
    - path: "podname"
      fieldRef:
        fieldPath: metadata.name
    - path: "annotations"
      fieldRef:
        fieldPath: metadata.annotations
    - path: "cpu_limit"
      resourceFieldRef:
        containerName: mycontainer
        resource: limits.cpu
    - path: "mem_limit"
      resourceFieldRef:
        containerName: mycontainer
        resource: limits.memory
  name: podinfo
```

```
    readinessProbe:
      exec:
        command:
          - /rprobe.sh
      initialDelaySeconds: 30
      periodSeconds: 30
```

```
volumes:
- emptyDir:
    sizeLimit: "40Gi"
  name: ebay
```

ebay

Host Runtime performance

- Kernel
  - Unify the platform and manage less kernel versions
    - Run latest kernel
    - drivers compatibility. ODM, GPU, etc.
- CPU & Power
  - p-state & c-state
    - Scaling governors: performance v.s. powersave
    - max-cstate: 0 or 9
      - Not absolute: if you need turbo
  - softirq & irqbalance/affinity
- Memory
  - THP?
    - Workload specific
  - Swap
    - avoid swap as much as possible, in k8s
      - Lesson learned: a small noisy daemon container could slow down the host
    - Overwrite *MemorySwappiness* default to 0 unless pod annotates

Host Runtime performance

- I/O
  - I/O scheduler: *cfq* v.s. *deadline*
- Storage
  - Storage classes
    - Local volume & CSI
      - Partitions
      - LVM
    - Stand-alone Cinder
- system config
  - Limit max_pid's
  - Others
    - vm.max_map_count (elastic search)
    - max sectors size / max_sectors_kb
    - vm.min_free_kbytes
    - etc.

Storage Class	IOPS/Throughput	Use Cases	Storage Solutions
Hot Tier	>50K IOPS 500 MB/s	NoSQL, eBay In-house App	Local SSD
Standard Tier	300 IOPS 150 MB/s	Archive	Ceph HDD based

Network performance

- RPS and RFS
  - Receive Packet Steering (RPS)
  - Receive Flow Steering (RFS)
    - set RPS and RFS at veth device
    - close to Baremetal throughput & retransmissions
- ipvlan
  - ovs bridge for common workloads
  - ipvlan for high-performance
  - ipvlan + ebpf ? upcoming
- ipvs caching
  - kube-proxy
- BBR
  - generally adopted by
    - search engine apps with high throughput
    - edge computing and software LBs
- tc qdisc - *fq_codel*

Manage the performance

- Monitor your performance
  - *API server and etcd metrics*
  - *Node Exporter and other DS -> Prometheus (for nodes)*
  - **Expose metrics from your controllers**
- Benchmarking
  - new (ODM) hardware, kernel, or driver, etc.
    - Burn-In tests
- OS image CICD
  - We build our own OS Image for host runtime
  - Workload certification is built-in with OS cicd

Tools and references

- http://www.brendangregg.com/linuxperf.html
  - perf
    - http://www.brendangregg.com/perf.html
- blktrace and btt
  - Block I/O Layer Tracing: blktrace. Alan D. Brunelle (Alan.Brunelle@hp.com). April, 2006.
  - btt User Guide. Alan D. Brunelle (Alan.Brunelle@hp.com). October 30, 2008
- fio and sysbench
- iotop
- iperf3
- dstat

ebay

KubeCon | CloudNativeCon

Europe 2019

❖ Summary

- ❖ Fleet management
  - ❖ Build and manage k8s with k8s (Shanghai kubecon)
- ❖ High performance k8s clusters control plane
- ❖ Run high-performance workloads in k8s

❖ What's Next

- ❖ cpuset and numa
- ❖ ipvlan + ebpf
- ❖ resource mgmt.
  - ❖ blkio cgroup
  - ❖ network qos

tess.IO