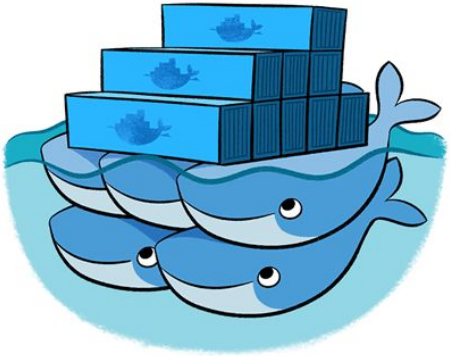


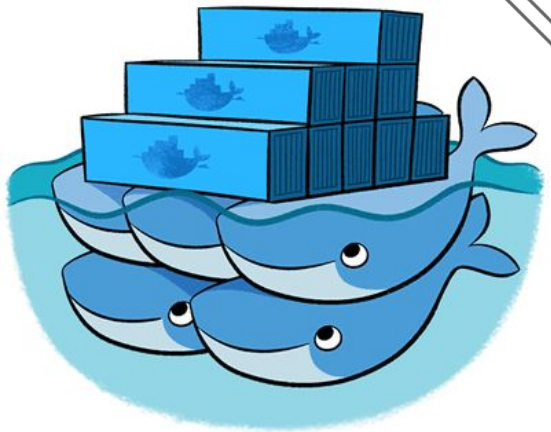
Swarm2K:

Build the world's largest Docker Swarm cluster

ชาญวิทย์ แก้วกสิ
Docker Captain & Swarm Maintainer
ห้องปฏิบัติการวิจัยไอยราคลัสเตอร์
มหาวิทยาลัยเทคโนโลยีสุรนารี

ACRL - Aiyara Cluster Research Laboratory

- Research and Development a low-power cluster
- Collaborate with Docker to develop Swarm since 0.1
- Docker Swarm has been downloaded more than 20 million times
- Been used to build clusters by developers world-wide



Software Container ?

simple tools, not big compl

IV

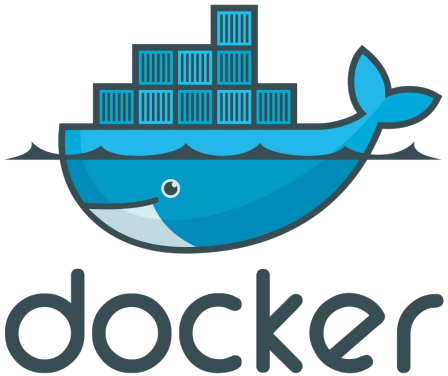
Define s interfaces f

assemb r systems.

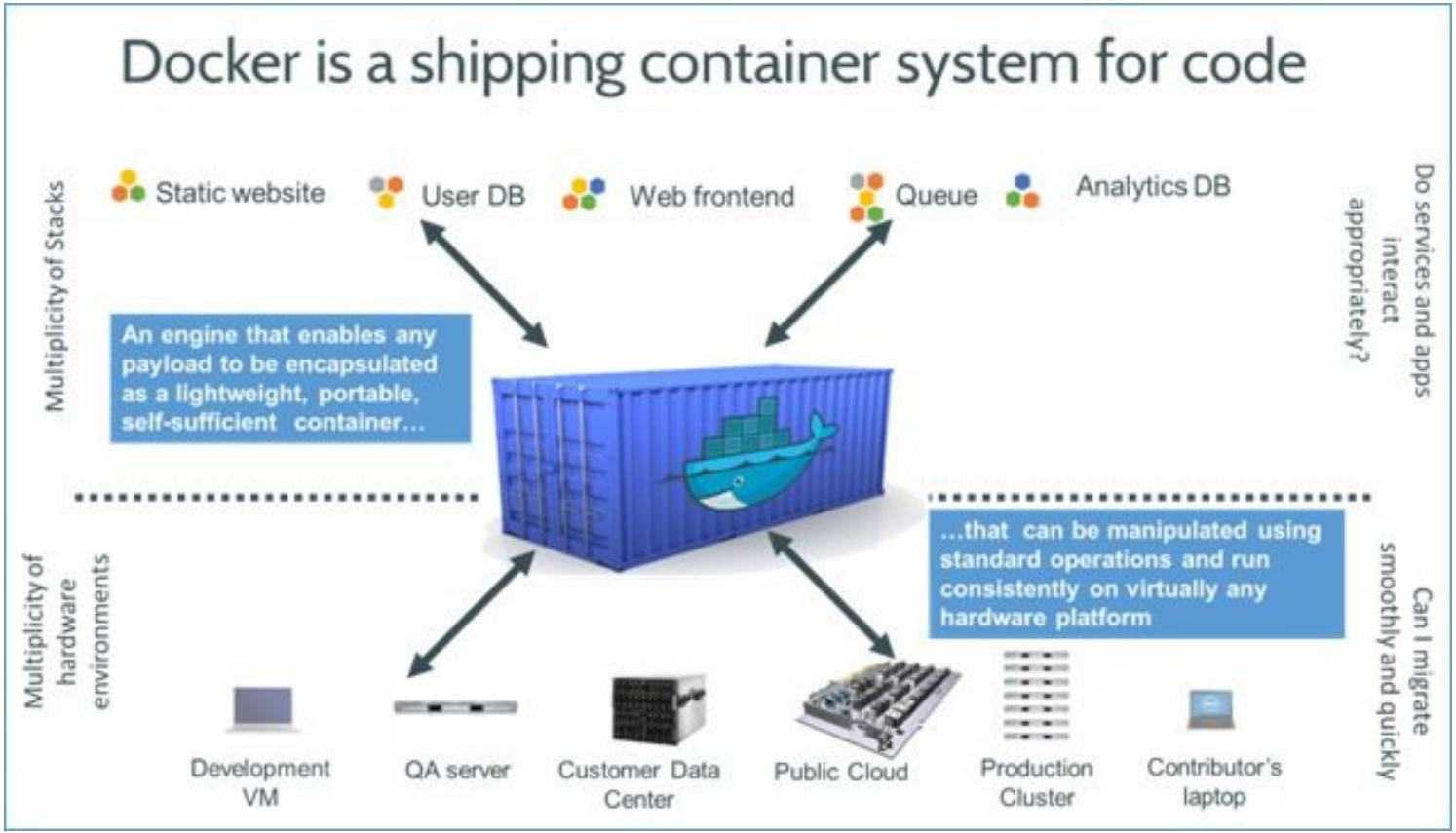
Software Container ?

- Solaris Zones
- LXC - Linux Containers

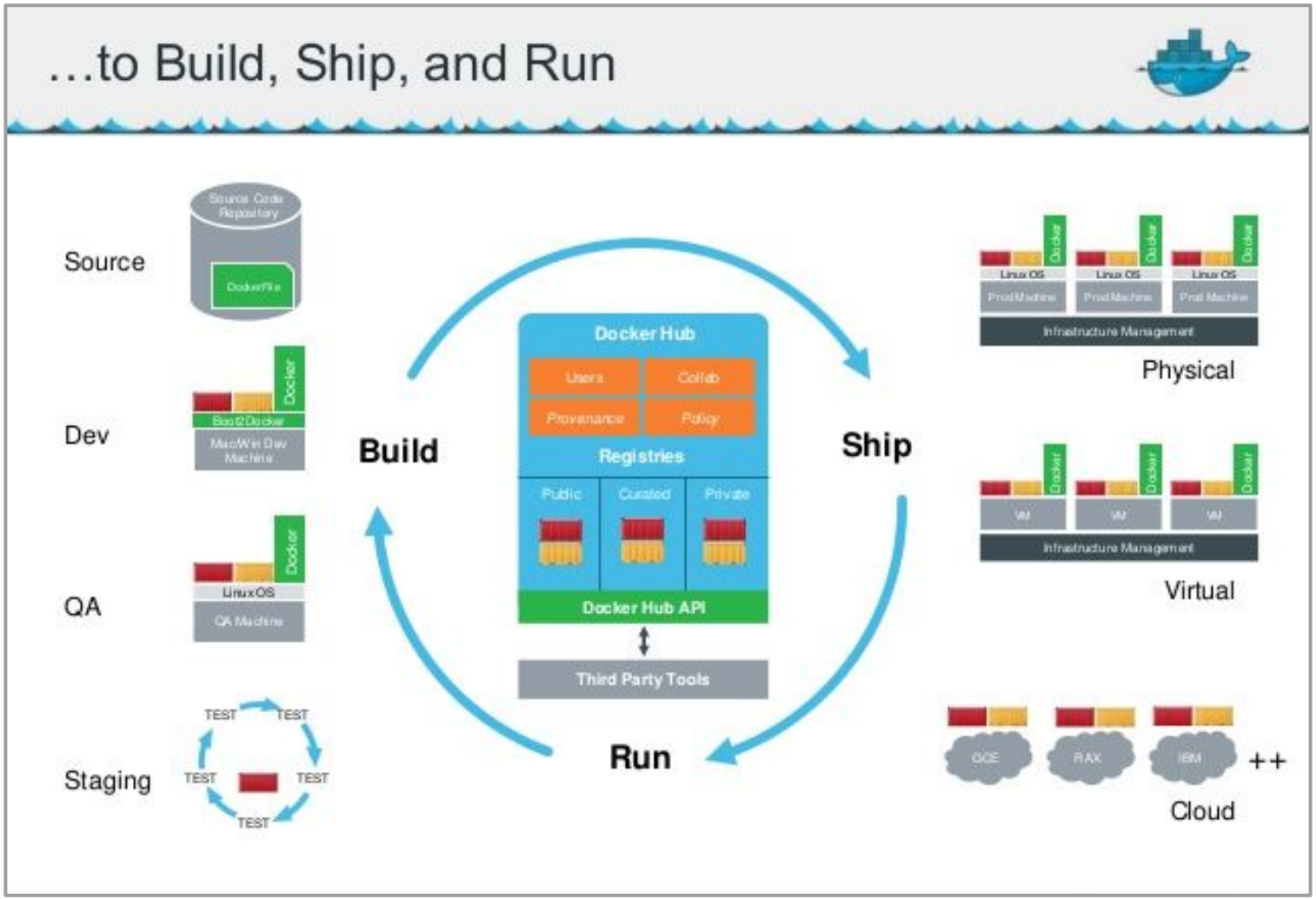
- Docker makes container commodity



Container and Docker

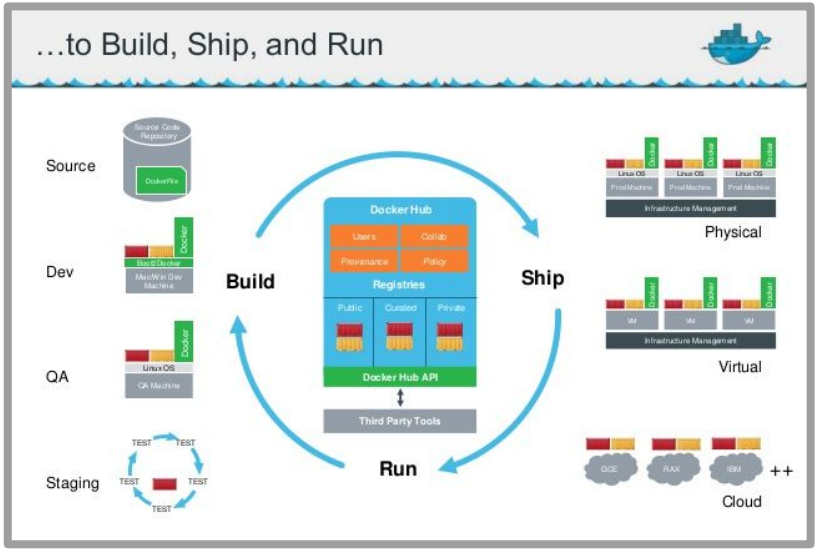


…to Build, Ship, and Run

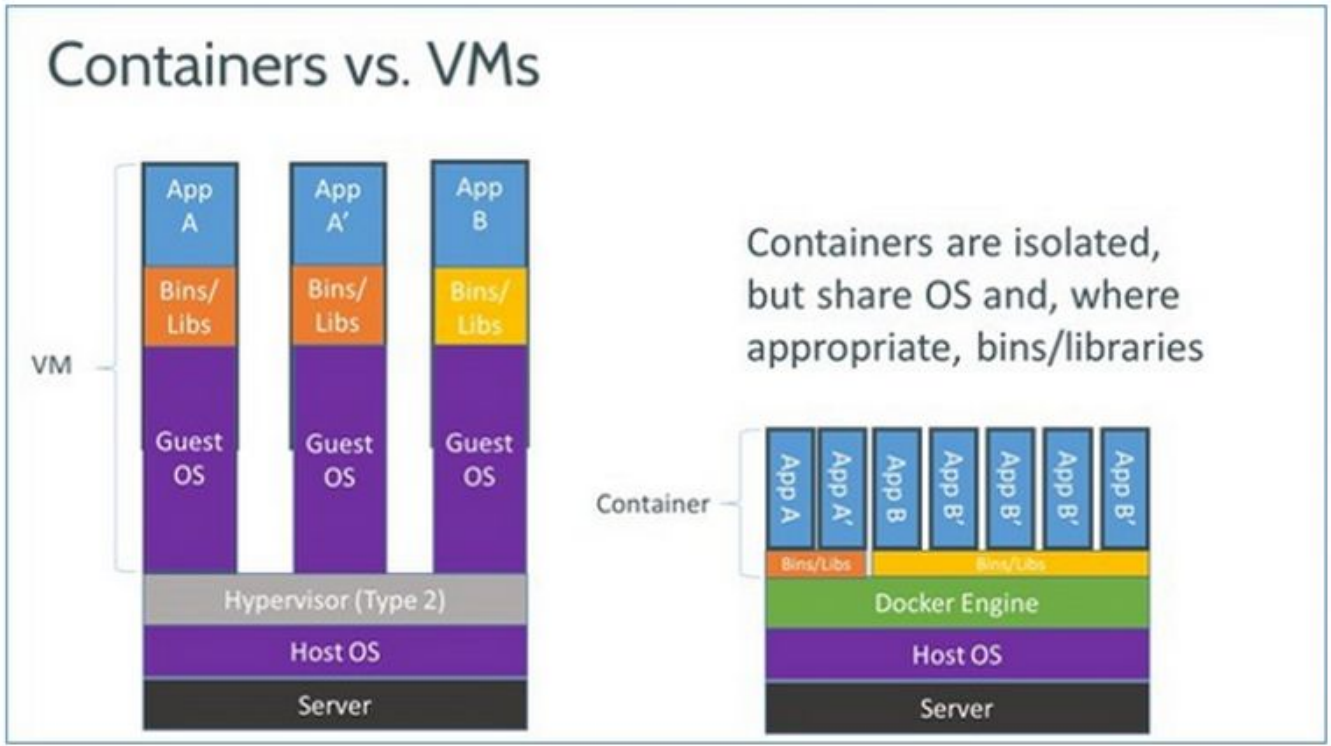


Tool that makes Container Commodity

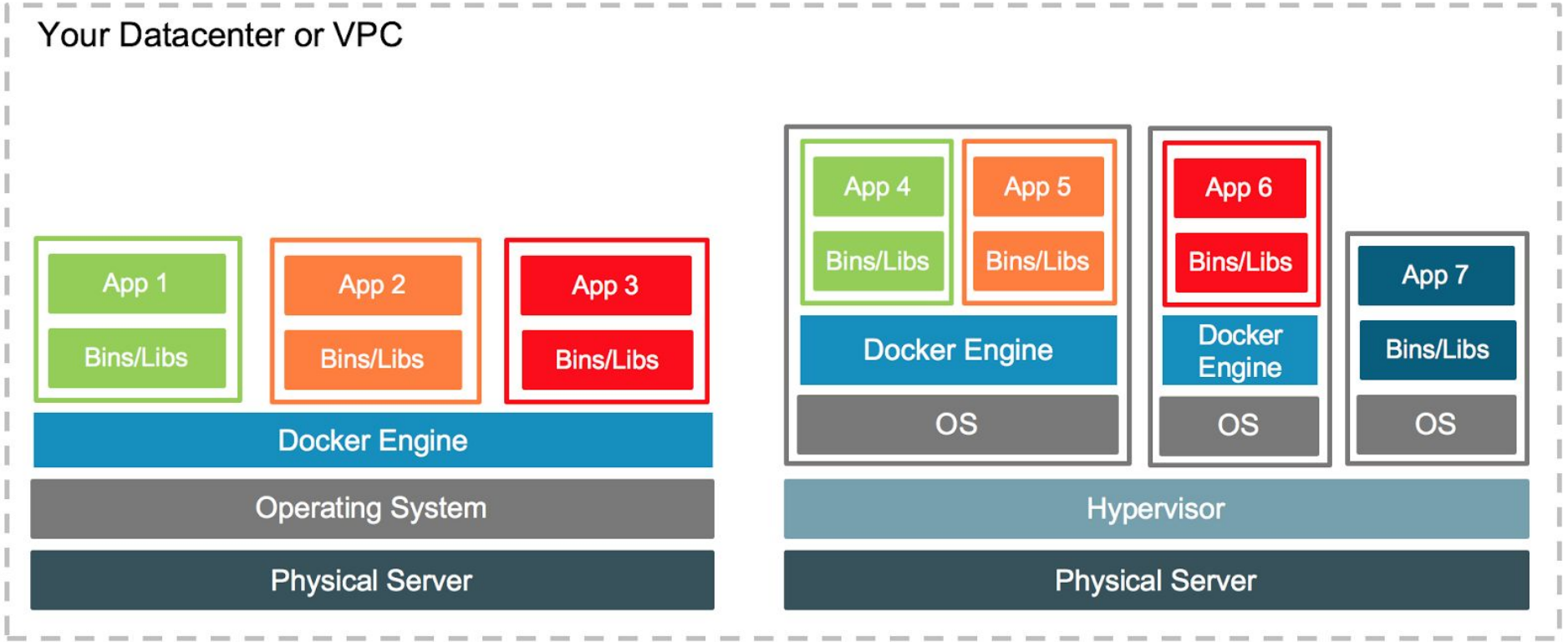
- **Build** - easily build and destroy containers
- **Ship** - easily move containers around because the help of union file systems
  - AUFS
  - Device Mapper Thin Provision
  - BTRFS
  - Overlay File System
- **Run** - easily run containers
  - $ docker run nginx



Containers vs. Virtual Machines



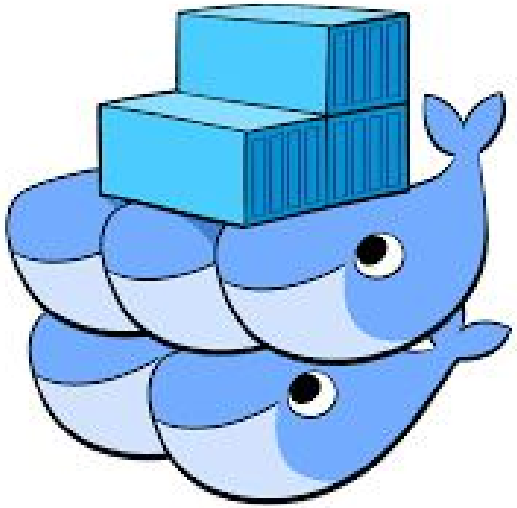
Containers co-exist with Virtual Machines



Swarm2K - Motivations

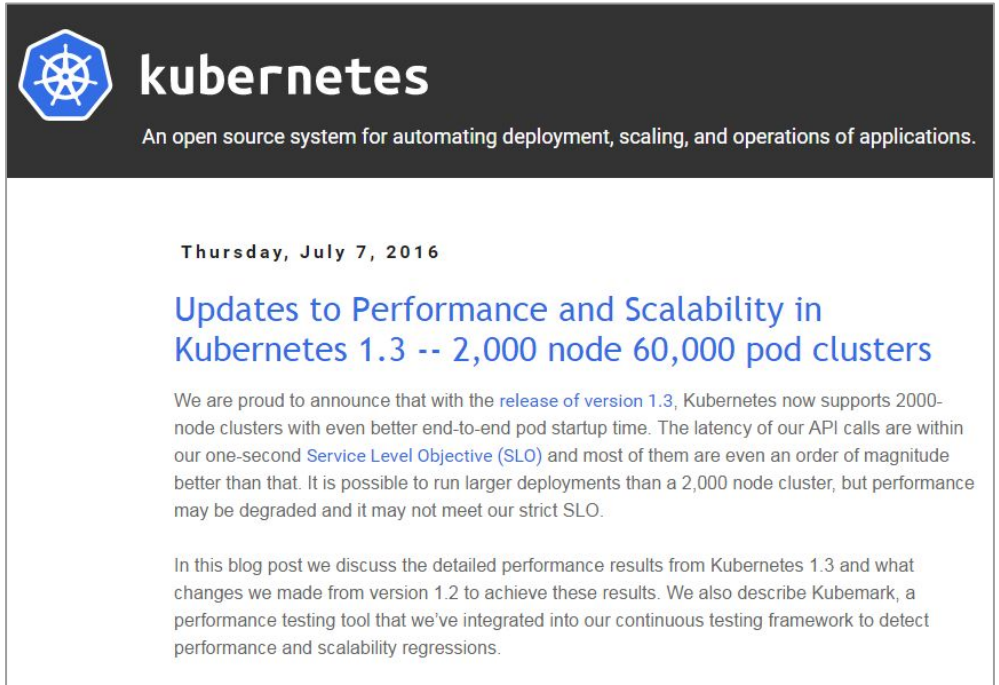
Since the announcement of Docker 1.12 Swarm mode at DockerCon 16, many people including me got the same questions:

- How large does the Swarm mode scale?
- Where is the Swarm limitation?



Motivations

- Early July 2016, A Google engineer responsible for testing Kubernetes
- He announced that Kubernetes was capable enough to form **2,000** nodes with **60,000** pods



kubernetes

An open source system for automating deployment, scaling, and operations of applications.

Thursday, July 7, 2016

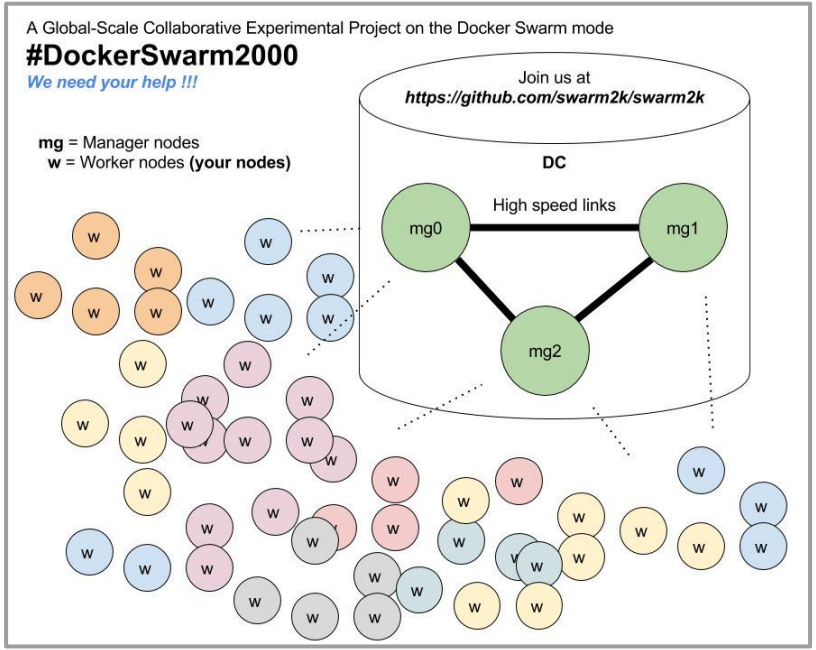
Updates to Performance and Scalability in Kubernetes 1.3 -- 2,000 node 60,000 pod clusters

We are proud to announce that with the release of version 1.3, Kubernetes now supports 2000-node clusters with even better end-to-end pod startup time. The latency of our API calls are within our one-second Service Level Objective (SLO) and most of them are even an order of magnitude better than that. It is possible to run larger deployments than a 2,000 node cluster, but performance may be degraded and it may not meet our strict SLO.

In this blog post we discuss the detailed performance results from Kubernetes 1.3 and what changes we made from version 1.2 to achieve these results. We also describe Kubemark, a performance testing tool that we've integrated into our continuous testing framework to detect performance and scalability regressions.

Swarm2K

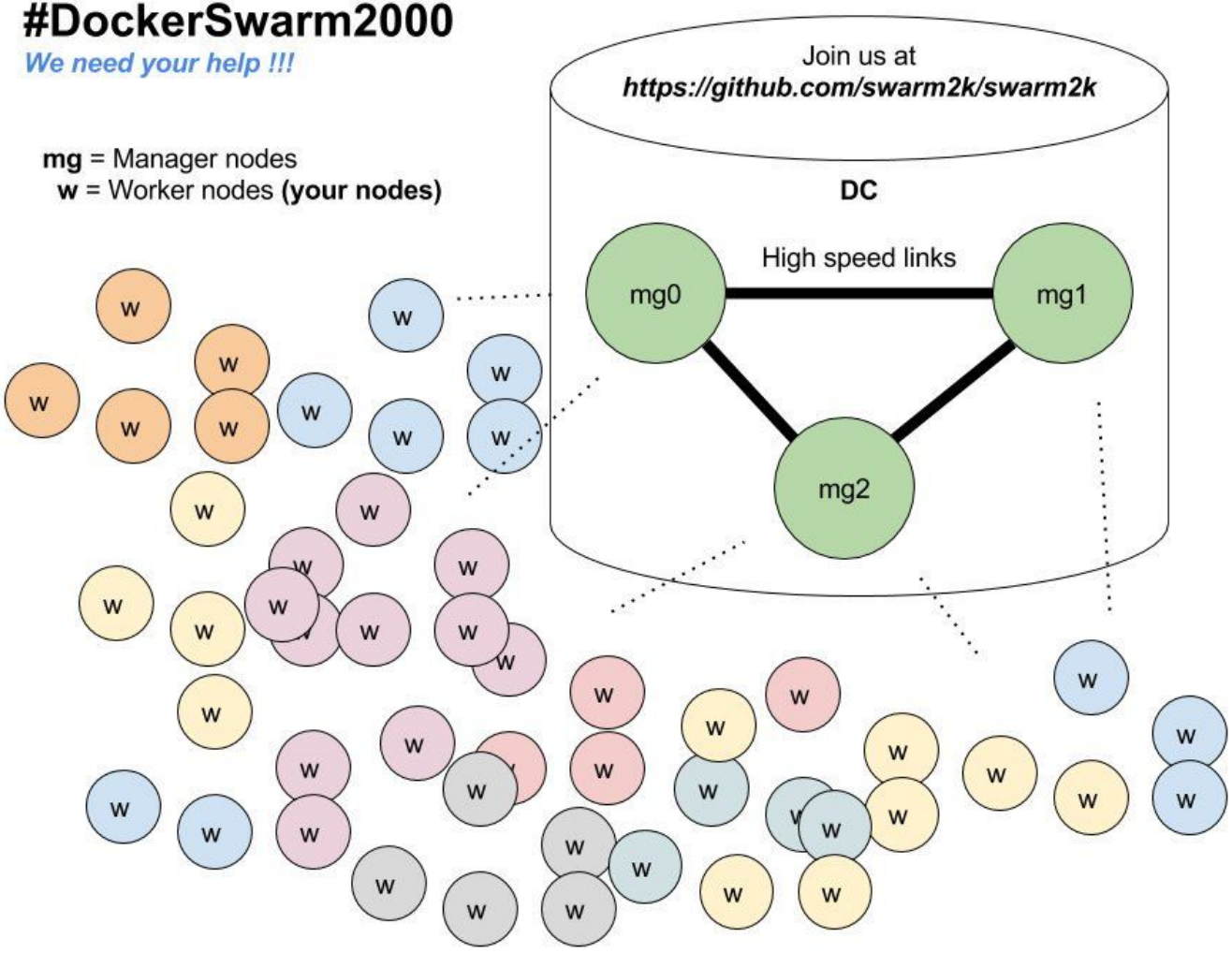
- Collaborative Project
- Setup to help Docker experiment its Swarm mode
- Crowdsourcing nodes
- Geo-distributed
- Contributions from many parts of the world
- Thailand: NIPA and 2 others



A Global-Scale Collaborative Experimental Project on the Docker Swarm mode

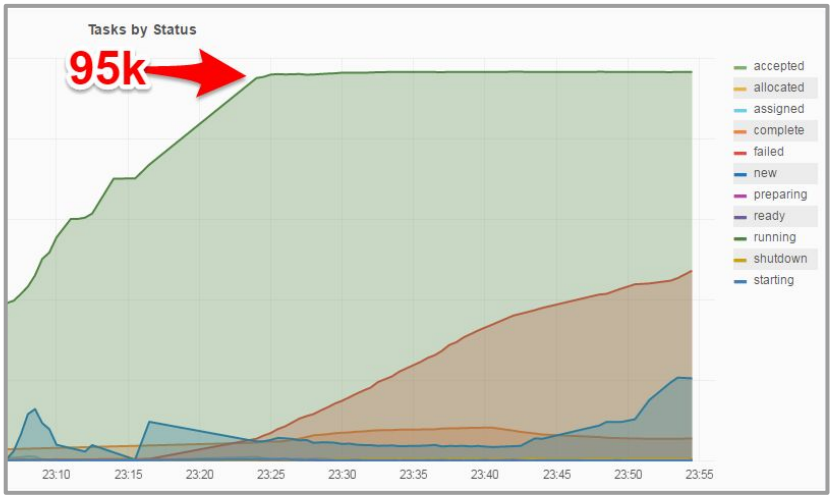
#DockerSwarm2000

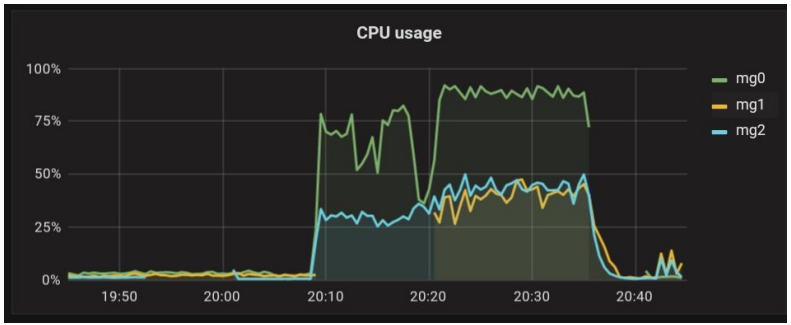
We need your help !!!



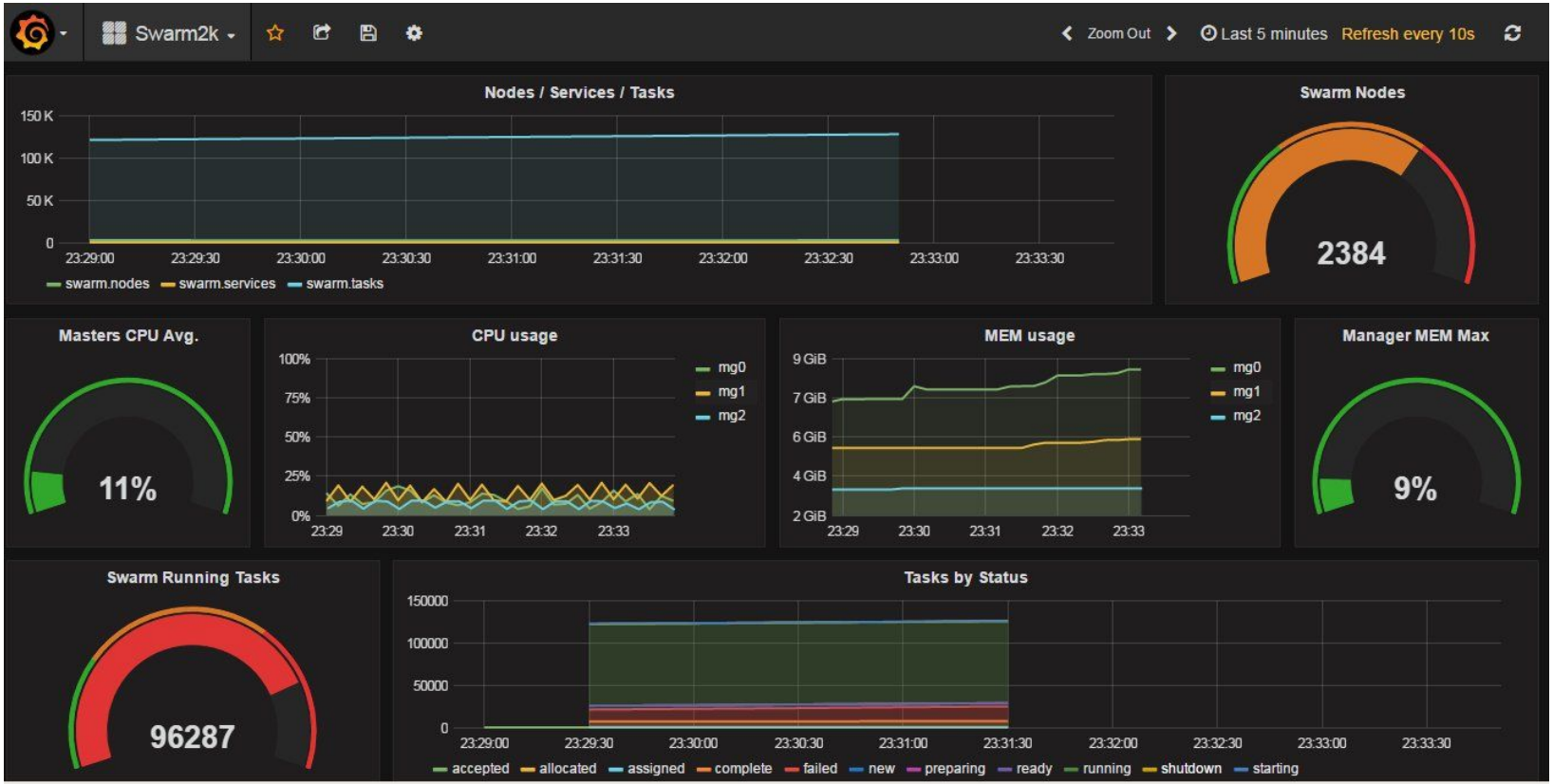
What is it?

- Largest Swarm cluster ever made
- 2,385 nodes
- 96,000+ concurrent tasks





Some stats


Swarm2k
Zoom Out
Last 5 minutes
Refresh every 10s
Nodes / Services / Tasks
150 K
100 K
50 K
0
23:29:00
23:29:30
23:30:00
23:30:30
23:31:00
23:31:30
23:32:00
23:32:30
23:33:00
23:33:30
swarm.nodes
swarm.services
swarm.tasks
Swarm Nodes
2384
Masters CPU Avg.
11%
CPU usage
100%
75%
50%
25%
0%
23:29
23:30
23:31
23:32
23:33
mg0
mg1
mg2
MEM usage
9 GiB
7 GiB
6 GiB
4 GiB
2 GiB
Manager MEM Max
9%
Swarm Running Tasks
96287
Tasks by Status
150000
100000
50000
0
accepted
allocated
assigned
complete
failed
new
preparing
ready
running
shutdown
starting

Side Effects

- No Kubernetes engineer tweets to me any more !!
- Mesosphere coins new term: Container 2.0

Together, we are shaking the world of software container !!

The Open Experiment

- Nothing to hide
- Transparent
- People joined the live chat while doing experiments

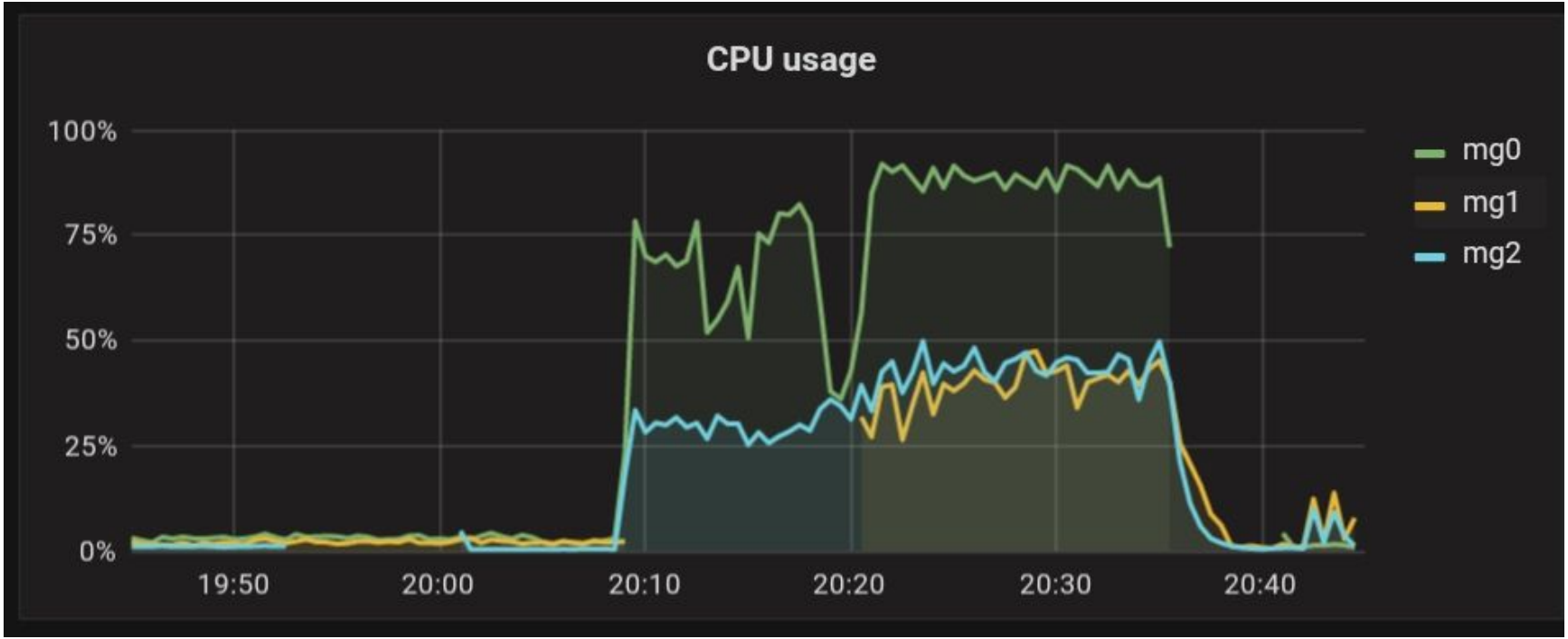
An Uncontrollable Environment

- Nothing in control
- 99% of nodes are not owned by me
- Me owned only 100 DigitalOcean nodes
- Nodes were joining and leaving
  - Some nodes died
  - Some nodes re-joined
- Live upgrade

Some numbers

- ~40 companies and individual participated
  - 3 teams from Thailand
- An 8-hour experiment
- 2,385 nodes of 512MB machines
  - Some are 1GB nodes
- ~96,000 concurrent containers
- 160% larger than the current Kubernetes limitation
- Produce ~700 MB Raft logs

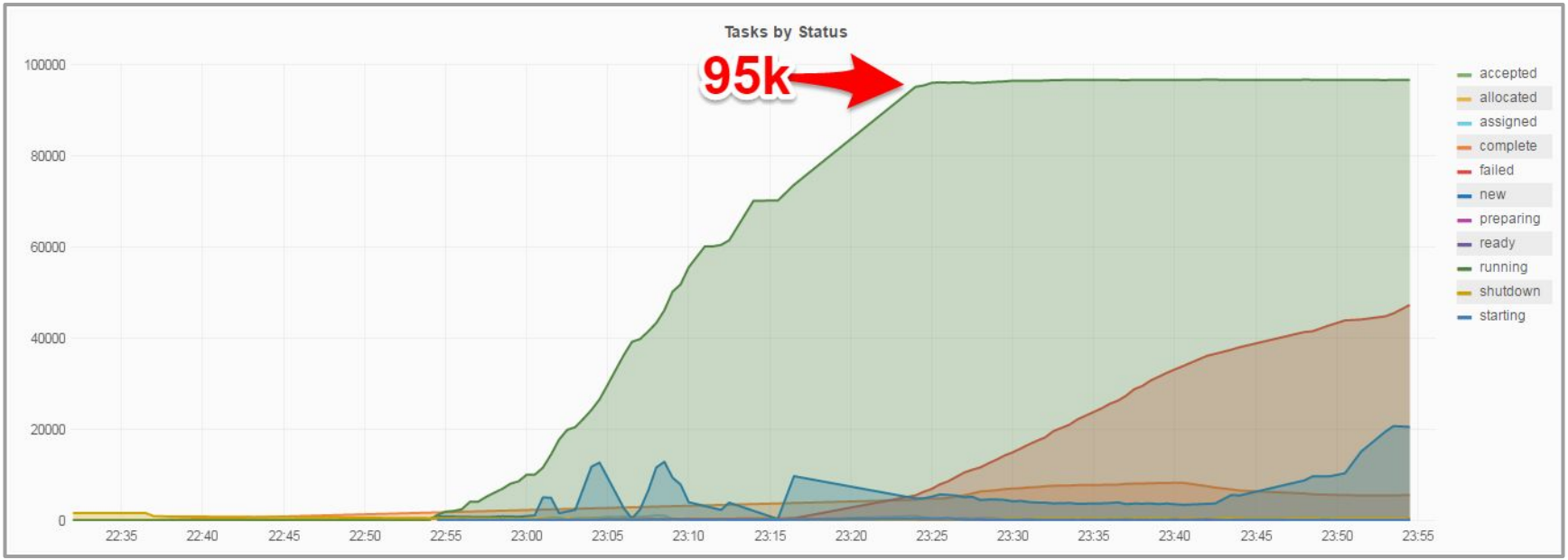
Live Upgrade Behaviour

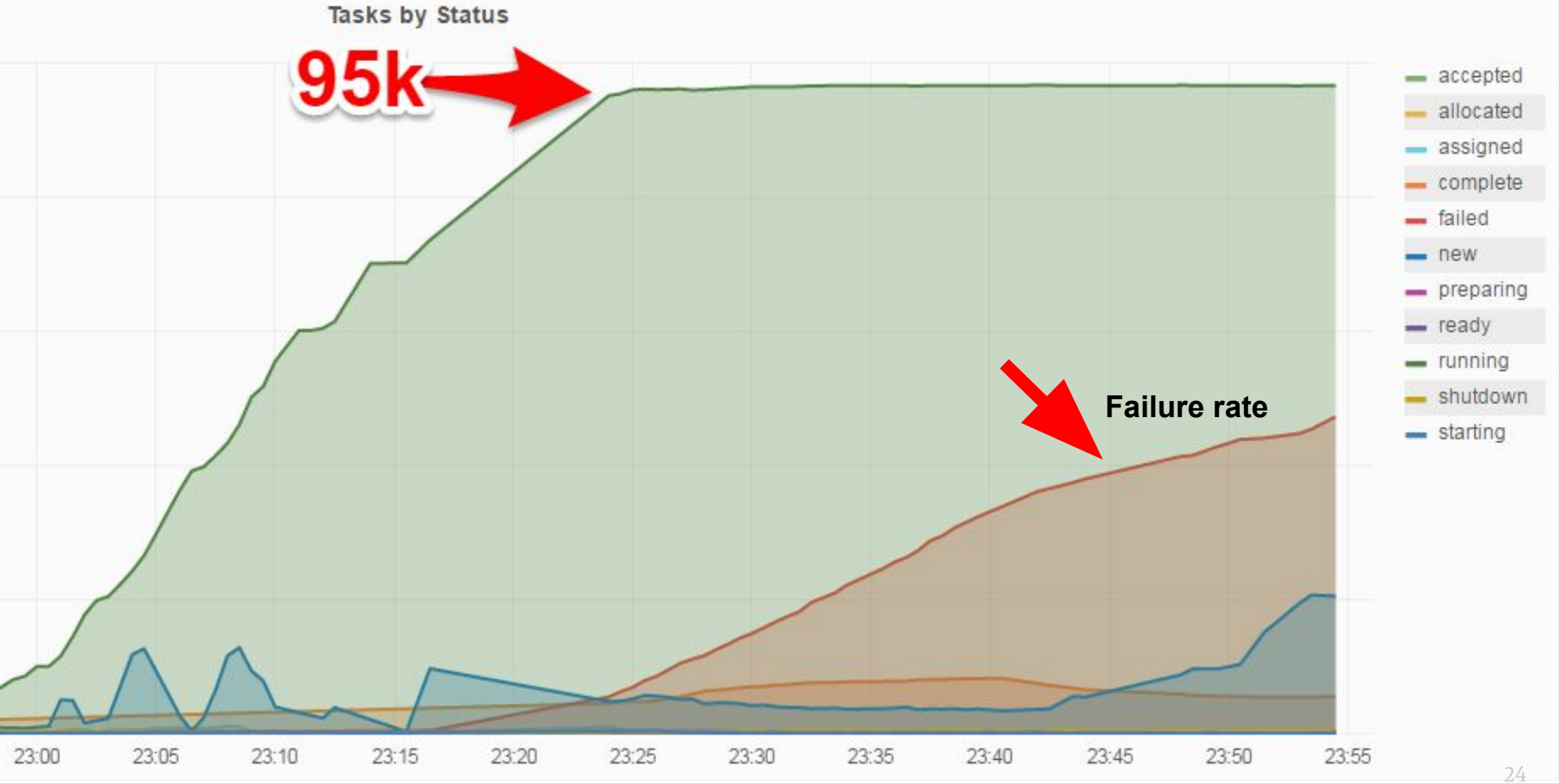


Software

- Ubuntu 16.04
- Linux kernel 4.4
- Docker 1.12-rc4
  - Swarm mode
- Grafana + InfluxDB + Telegraf for monitoring
- Alpine as containers
  - Smallest possible
  - ~ 70 containers per node during pre-test on an 512MB node

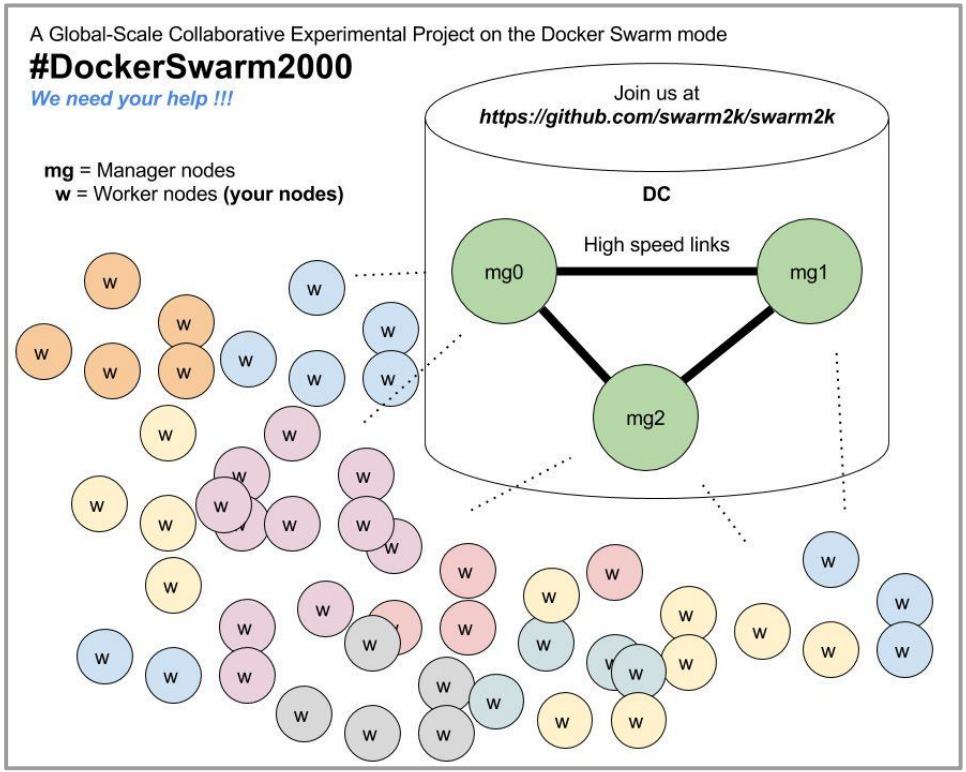
Limitation at the moment




Tasks by Status
95k
Failure rate
accepted
allocated
assigned
complete
failed
new
preparing
ready
running
shutdown
starting
23:00
23:05
23:10
23:15
23:20
23:25
23:30
23:35
23:40
23:45
23:50
23:55

Server Topology

- 3 Managers
- Formed Raft Quorum
- All managers in the same data center
  - We need high-speed links for cluster stabilization
- All managers in the Drain mode
  - Recommended by the Docker core team



Where are we going?

- Swarm3K
- 3,000+ nodes coming this year
- To test Docker an RC of 1.13

Join us !!

https://github.com/swarmzilla

Q & A