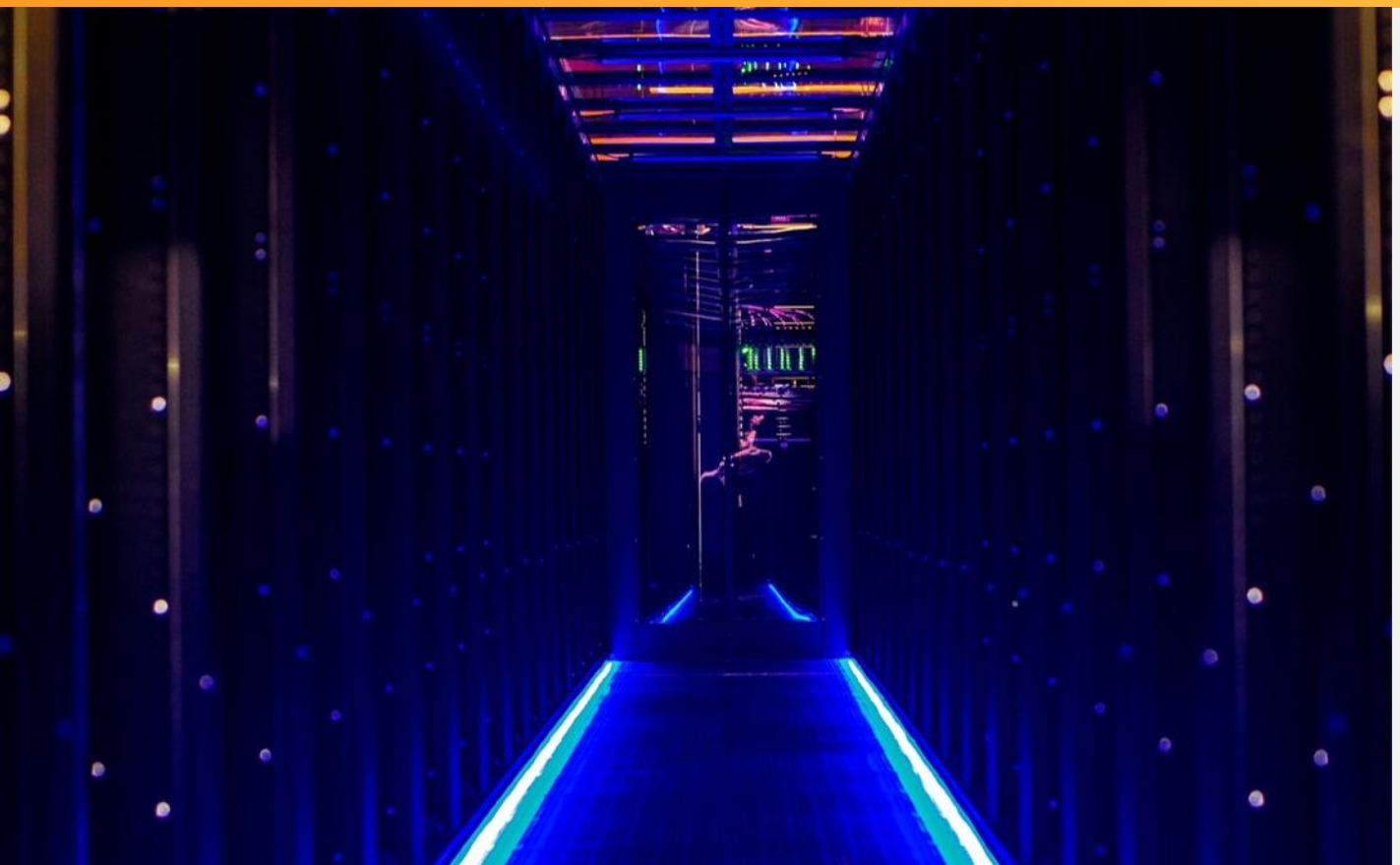
Operating Mesos-powered Infrastructures

Operating 600+ servers on 7 DCs @ Criteo : sharing some insights



Pierre Cheynier

@pierrecdn

Operations Engineer, SRE Division

October 27, 2017

criteo

Company

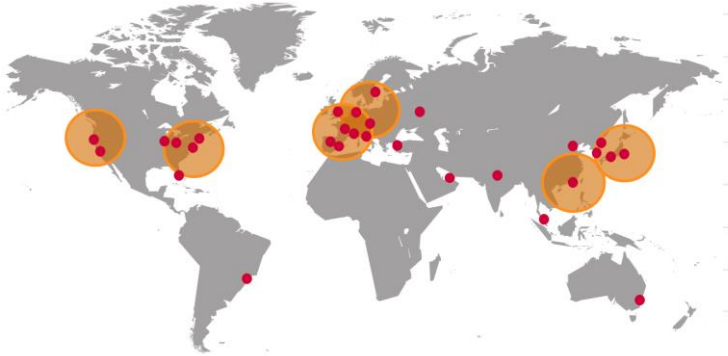
- **2,700 employees** (600 R&D engineers), **30 offices**
- **1.2B distinct users**/month
- **Billions of ads served & transactions analyzed / day**

2005 - CREATION DATE

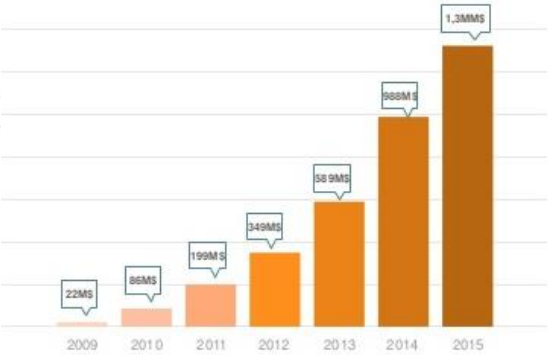
2013 – NASDAQ IPO

- **7 datacenters + 15 network PoPs**
- **20K servers** (Linux/Windows mix)
- **3M RPS** at peak time
- **Real Time Bidding: ~ 10 ms**
- **Hadoop: 171 PB storage** (+600TB per day)

2009 – GOING ABROAD

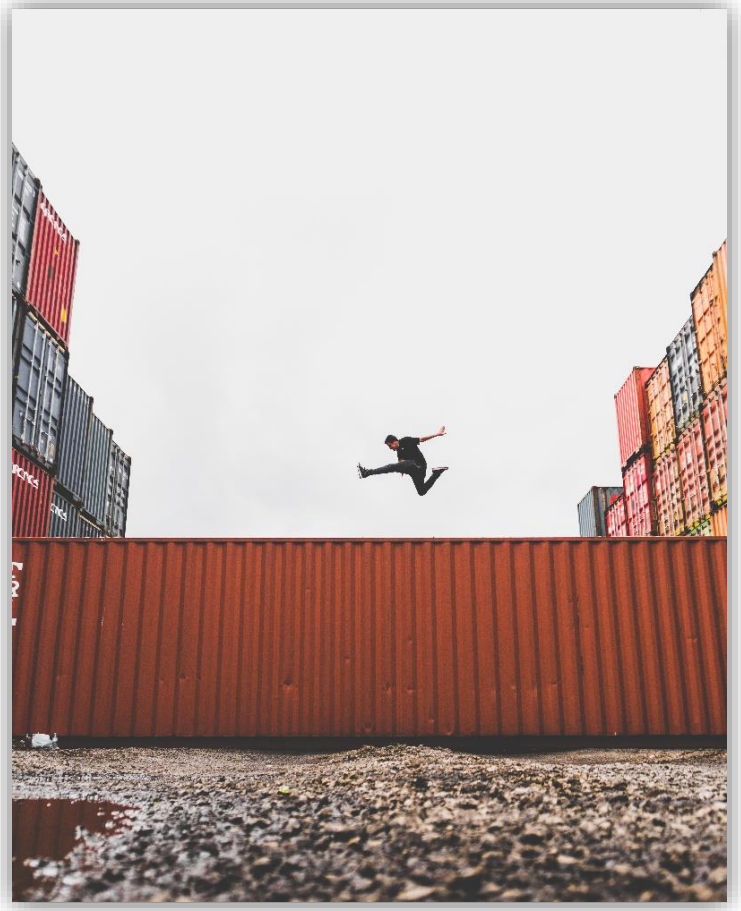


2016 - +1B REVENUE !



Transitioning…

- **Hardware : reducing the Total Cost of Ownership**
  - Filling racks on premises ➔ fully populated cabinets, repeatable process
  - Fully secured (RAID, 2 x power, ...) COTS ➔ commodity hardware
- **O/S : maintainability**
  - Windows ➔ Linux
- **Runtime : diversity**
  - .NET Framework ➔ CoreCLR (.NET Core Runtime) & JVM
- **Platform deployment : flexibility, self-service**
  - IT automation ➔ Tasks/Job Orchestration



Transitioning…

criteo

- **Stable & Maintainable system => Simple & Modular**

Why Mesos ?

criteo

- **Small and Extensible project**
  - A highly-available distributed system kernel, abstracting and isolating resources in less than 250k LoC
  - Concrete primitives and interfaces, extensibility through Modules
  - Implementing industry standards (such as CNI, CSI & OCI soon)

- **Self-sufficient**
  - Mesos Containerizer
  - UCR

- **Where are we ?**
  - Started a small PoC during 2015 S2
  - 1.5 year later: 600 agents, 150+ production apps, 250K QPS
  - 2 generalist frameworks, ML-oriented & GPU-based workloads coming.

The long journey of setting up production-grade infrastructures

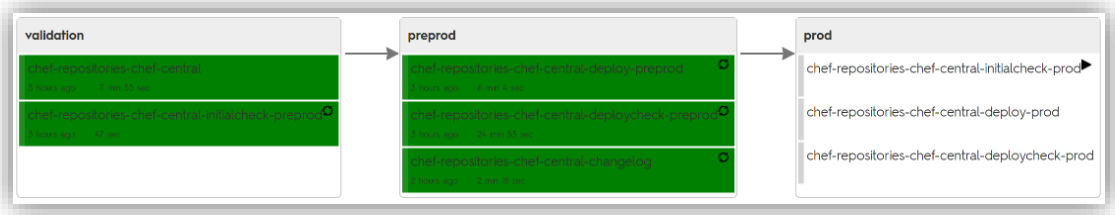
- 1 - Automate everything
- 2 - Configure defensively
- 3 - Discovering services and more
- 4 - Provide visibility to the end-users
- 5 - Networking is hard

1 - Automate everything

- **Chef: our all-purposes config management tool**
- **Automate everything:**
  - address hardware scale up/down operations in minutes.
  - Choregraphie: perform complex ops using lock-based resource protection
- **Reliability > CI pipelines:**
  - perform tests in VMs
  - deploy in preproduction environment

```
$ ./scale_tla mesos_agent ssd TY5 10
OK! 10 ssd servers in 7 distinct racks and 3 pods
Pick 10 ssd servers from the TY5 inventory...
Waiting for them to report in Chef...
Assign mesos_agent role...
```

```
48      - default['mesos']['version'] = '1.3.0'
    48  + default['mesos']['version'] = '1.4.0'
```



2 - Configure defensively

- **Identify fault-domains**
  - Placement constraints
- **Take care of user secrets**
  - Authenticate everything
  - Encryption channel provided through asymmetric crypto & key distribution
  - Mesos Secrets available now (1.4.0) - **`SecretResolver`**
- **Enforce limits**
  - CPU: for predictability use **`--cgroups_enable_cfs`**
  - Mem: turn off swap (hi OOM-killer !)
  - Disk: turn on disk quotas / unbounded by default on Marathon / understand GC.
  - User: mandatory (forbid root usage and grant frameworks through Mesos ACL).
- **Perform backups**
  - And try to restore ! (beware of API consistency / versioning)

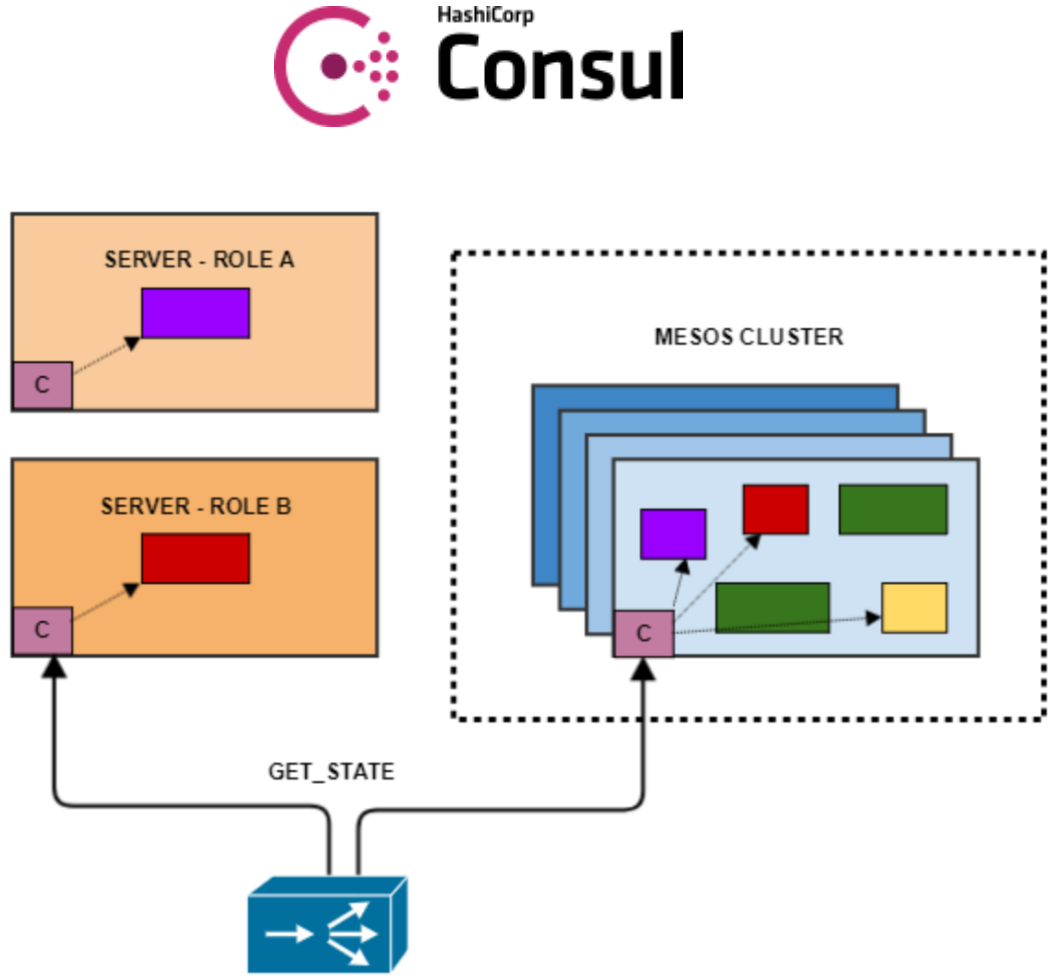
```
network=10g;
network_infra=L3;
platform=centos;
platform_version=7;
rack_name=807;
pod_name=6;
type=base;
```

```
"SECRET_PW":"nRRCP3B0Y…"
```

```
$ curl -XPOST -d @backup.json
(…)
HTTP/1.1 400 Bad Request
{"message":"Invalid JSON"}
```

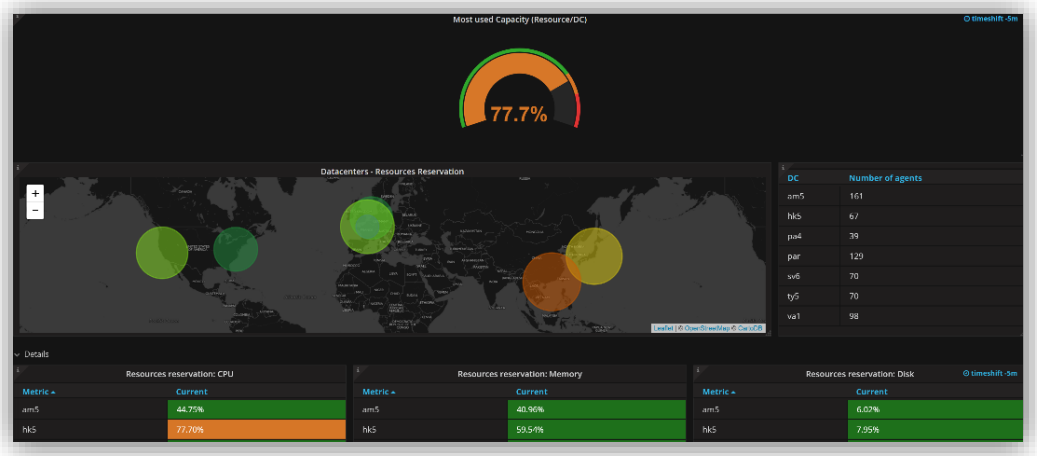
3 - Discovering services and more

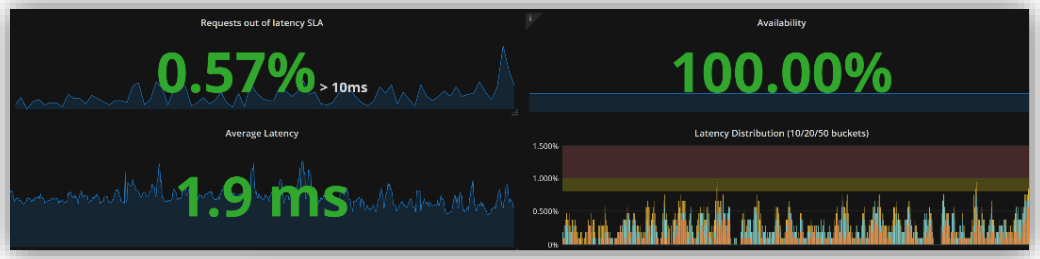
- **Flat Service Discovery model**
  - Don’t forget legacy !
  - Help managing the DC bootstrap case
  - Fallback to the nearest DC using “prepared queries”
- **Intra-DC communications** : 1 network hop
  - Consul API (DNS / HTTP)
  - CSLB library embedded in Criteo SDK
- **Consul as a DC, Services and State reference**
  - Tags and K/V used to store services metadatas
  - Consul health-check as a general state reference
  - Practical applications: automatically provision LBs, smooth transitions between legacy and Mesos.

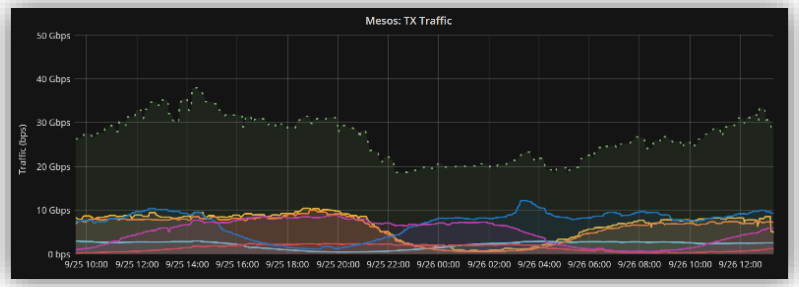


4 - Provide visibility to the end-users

- **Cultural changes**
  - App instances move continuously !
- **Metrology & Alerting**
  - Collectd, prometheus_exporter, etc.
  - Not well-known metrics, from `mesos.proto` :
    - Networking: `net_[rx|tx|tcp]*`, `[TrafficControl|Ip|Tcp|Udp]Statistics`,
    - Disk I/O: `CgroupInfo.Blkio.CFQ.Statistics`
    - Tracing: `PerfStatistics` (costly!)
- **SLAs**
  - Transparency about platform footprint
  - Report your ability to schedule – chaos monkey involved !
- **Debugging / Tracing**
  - The Mesos I/O Switchboard: remotely attach/exec
  - Introducing system tracing components such as LTTng







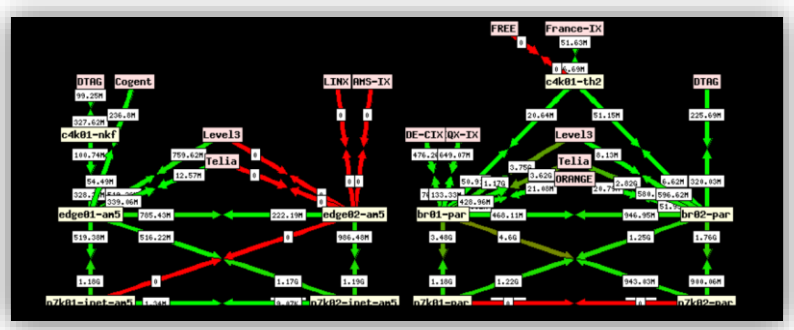
5 - Networking is hard

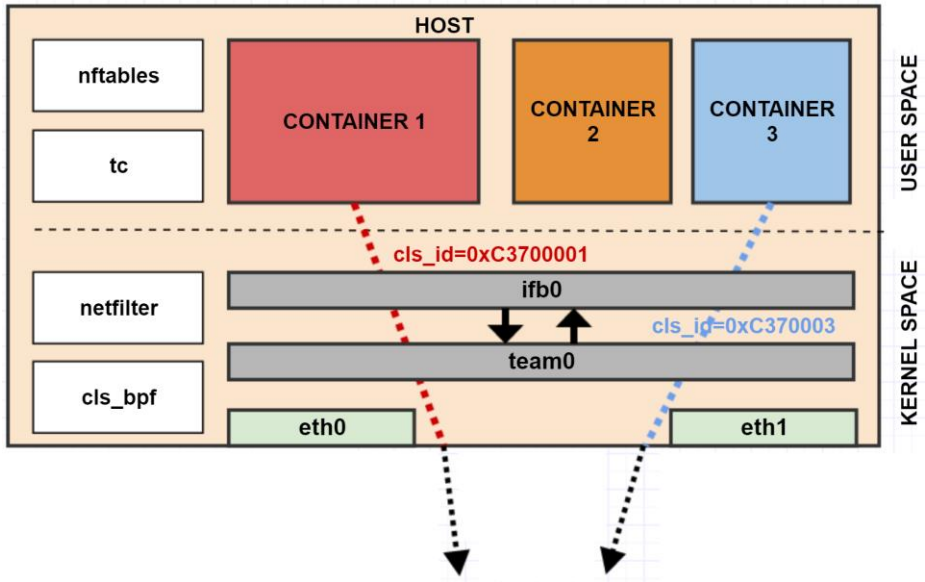
- **“The network is reliable”**
  - *The 8 fallacies of distributed computing* (L. Peter Deutsch - 1994)

- **Load-balancing**
  - Providing services such as: visibility, timeout profiles, sticky cookie, TLS...

```
"labels": {
  "DNS_ENTRY_AP": "mesoscon2017.crto.io",
  "DNS_ENTRY_AP2": "mesoscon.crto.io",
  "STICKY_COOKIE": "tasty_cookie"
}
```

  - Use the new “seamless reloads” feature (1.8-dev2).
- **`net_cls` cgroup : the simplest way to introduce basic QoS**
- **Noisy neighbours > which trade-off will you choose ?**



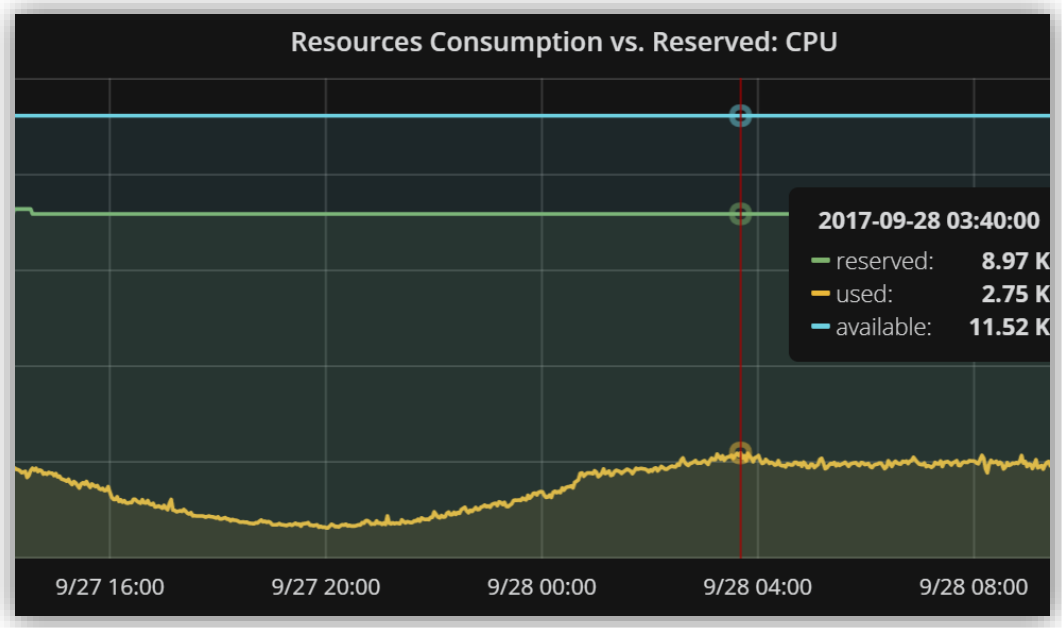


Incidents…

- **DC Outages**
  - Jul, 2017: “The **site has been evacuated and the Fire Department has been notified.** Every server basically got shutdown and restarted”.
- **Disaster recovery scenarios**
  - Apr, 2017: “**Marathon applications were deleted WW**”
  - Jun, 2017: “**Zookeeper does not accept connections anymore**, has been satured by Aurora, new task deployments are in pending state”
- **Noisy neighbours**
  - “Network latencies on 1 instance increased a lot (average, 95pctl)”
  - “In 1 cabinet row, switches backplanes are currently saturated”

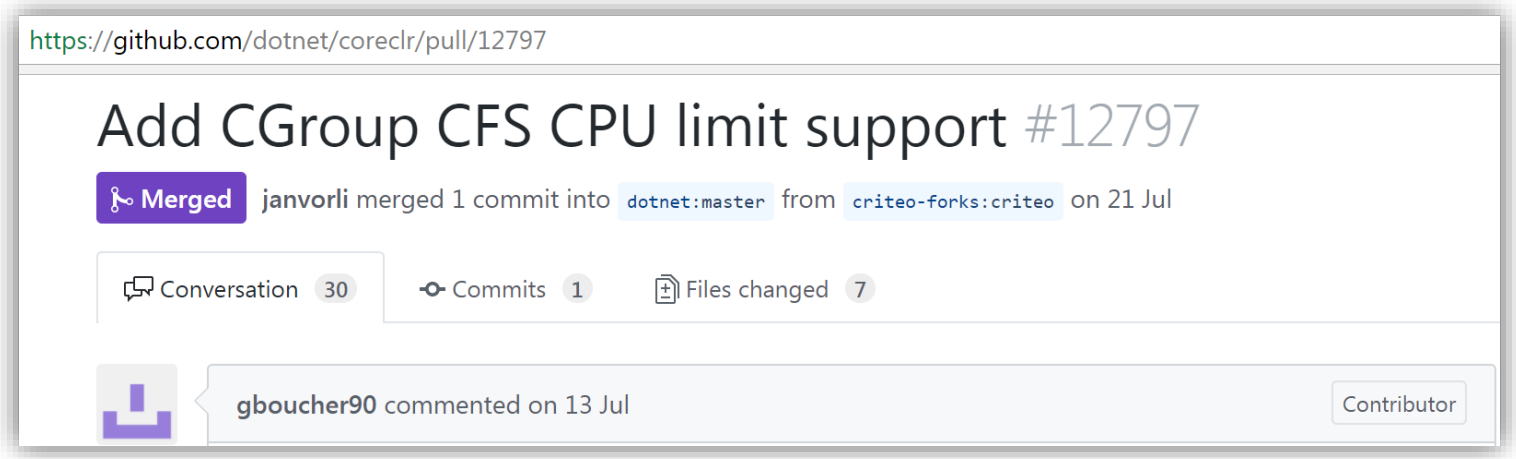
What's left to answer ?

- **Isolation, isolation, isolation**
  - Network and I/O bandwidth as a first-class resource ?
  - Latency critical apps: combine with `cpu_set` ?
- **Efficiency**
  - Revocable resources for non-latency critical tasks (jobs) ?
  - Quotas + Oversubscription ?
  - Bin packing (= reclaim hardware … & electrical power !)
- **Maintenance Primitives**
  - Anticipate more complex operations by reclaiming resources and not allocating new tasks.



Happy users !

- **Providing support and sharing knowledge leads to great contributions**



Thank you.

Do you want to know more ?

We’re hiring !

criteo