CCTC 2016 中国云计算技术大会
Cloud Computing Technology Conference 2016
CSDN

谷歌容器集群管理系统实践

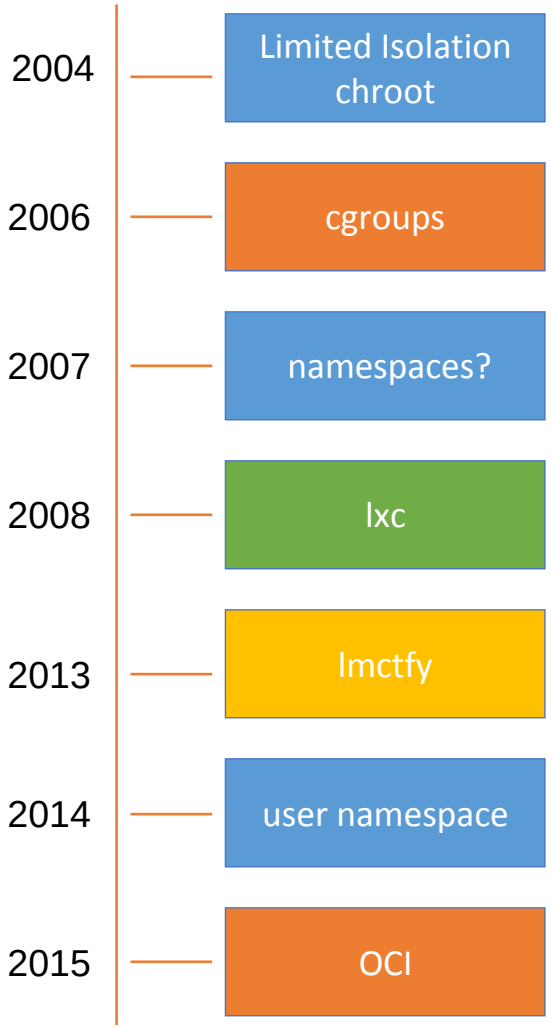
邓德源
才云科技

- Development
  - 20% time side project
    - work on any project you want
    - dashboard for posting jobs
    - you can do more than you think!
  - Be SRE for 2 or 3 months (20% more salary!)

- SRE (Site Reliability Engineering)
  - 50% time for development
    - to automate routine tasks
    - scale nicely
  - SRE consists of two categories: 55% SDE, and 45% Ops
  - Say no to development team
    - error quota, e.g. 99.95% slo monthly means 4hrs impact
    - both SRE and Development team mange the risk
  - Ability to change code base
  - Postmortem
    - Not to blame, but to find problems and fix it

Container

- Reason
  - primary goal: save money - VM is heavy
  - high-density and performance
  - fast to start
- Start Container Journey
  - 2004 - ?
- Everything runs in container
  - use container for decade
  - >2B container a week



- cgroup
  - resource isolation (cpu, memory, blockio, etc.)

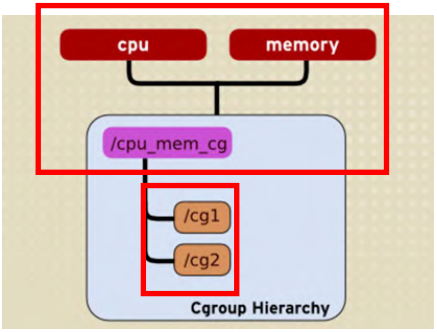
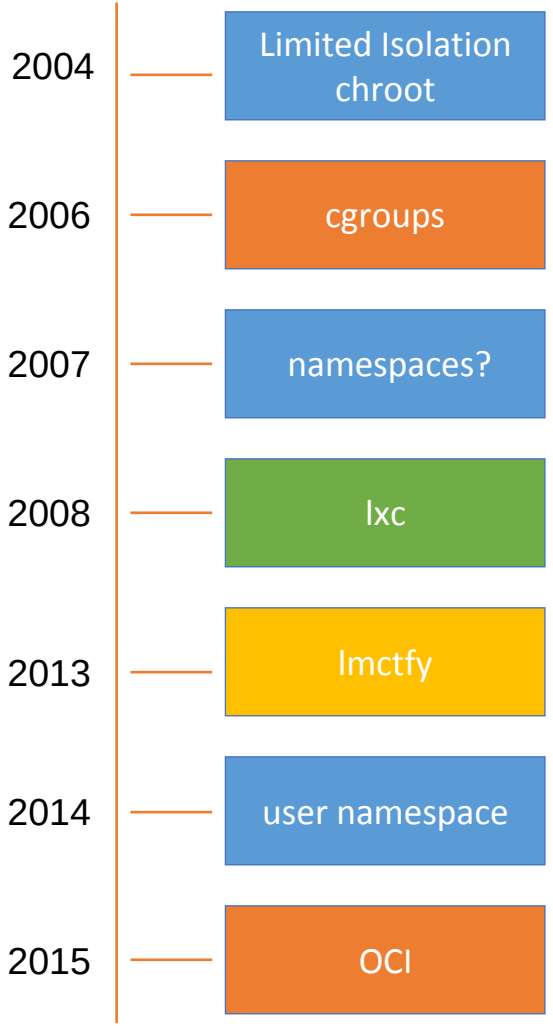


Image from Redhat

```
$ sudo mount -t tmpfs cgroup_root /sys/fs/cgroup
$ sudo mkdir /sys/fs/cgroup/cpu_mem_cg
$ sudo mount -t cgroup -o cpu,memory hier1 /sys/fs/cgroup/cpu_mem_cg/
$ sudo mkdir /sys/fs/cgroup/cpu_mem_cg/cg1
$ sudo mkdir /sys/fs/cgroup/cpu_mem_cg/cg2
$ ls -l /sys/fs/cgroup/cpu_mem_cg
drwxr-xr-x 2 root root 0 May 10 02:42 cg1
drwxr-xr-x 2 root root 0 May 10 02:45 cg2
-rw-r--r-- 1 root root 0 May 10 02:41 cpu.shares
-rw-r--r-- 1 root root 0 May 10 02:41 memory.limit_in_bytes
-rw-r--r-- 1 root root 0 May 10 02:41 tasks
...
$ ls -l /sys/fs/cgroup/cpu_mem_cg/cg1
-rw-r--r-- 1 root root 0 May 10 02:41 cpu.shares
-rw-r--r-- 1 root root 0 May 10 02:41 memory.limit_in_bytes
-rw-r--r-- 1 root root 0 May 10 02:41 tasks
...
$ ls -l /sys/fs/cgroup/cpu_mem_cg/cg2
-rw-r--r-- 1 root root 0 May 10 02:41 cpu.shares
-rw-r--r-- 1 root root 0 May 10 02:41 memory.limit_in_bytes
-rw-r--r-- 1 root root 0 May 10 02:41 tasks
...
```



- cgroup v1 -> v2
  - multiple hierarchy
  - subcontainer and asymmetric isolation
  - v2: unified hierarchy

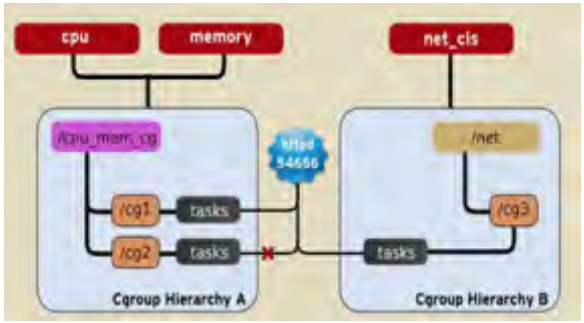
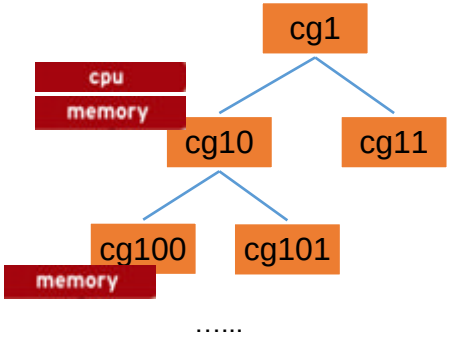
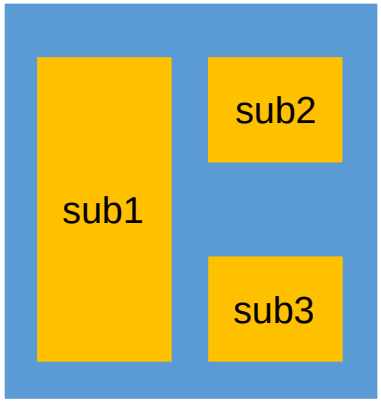
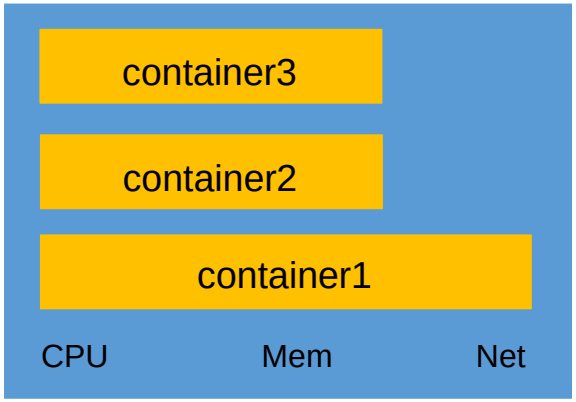


Image from Redhat

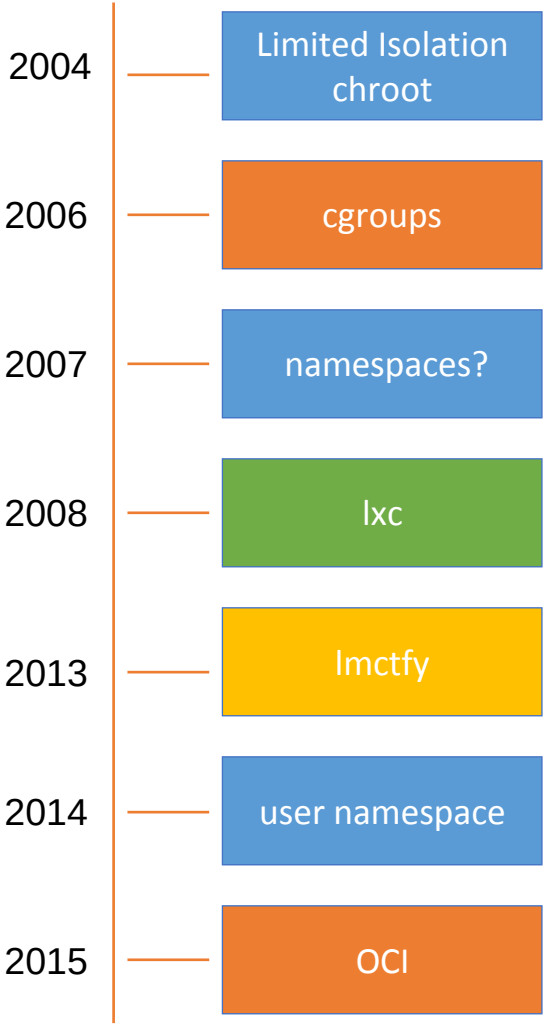




subcontainer



asymmetric isolation



- Namespaces
  - isolate process's view of the operating environment
  - Mount, UTS, IPC, PID, Network, User

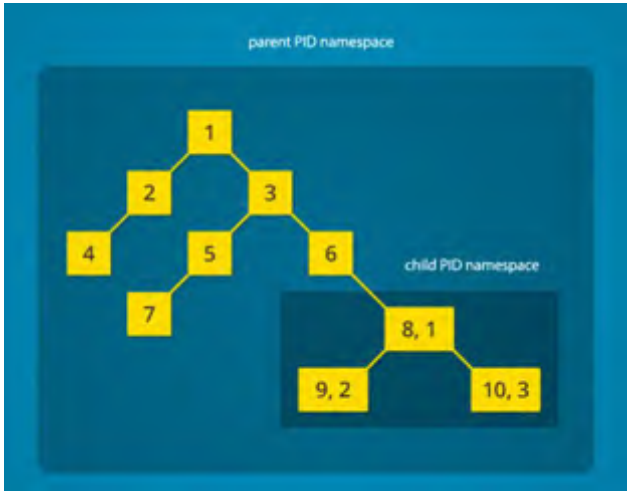
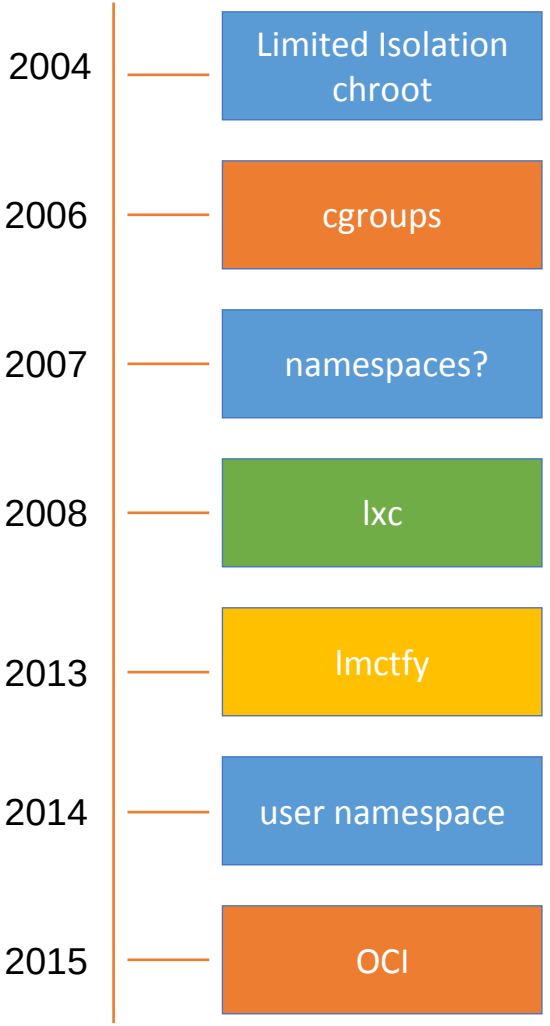


Image from Toptal

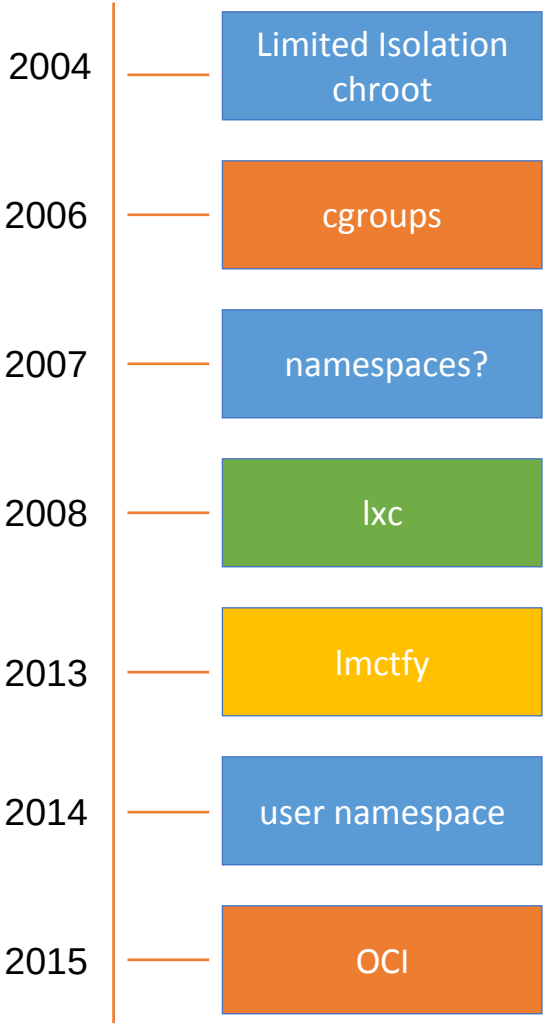
- Support added fair recently
  - Primarily Linux cgroups
  - 'namespace' is done at user-space policies



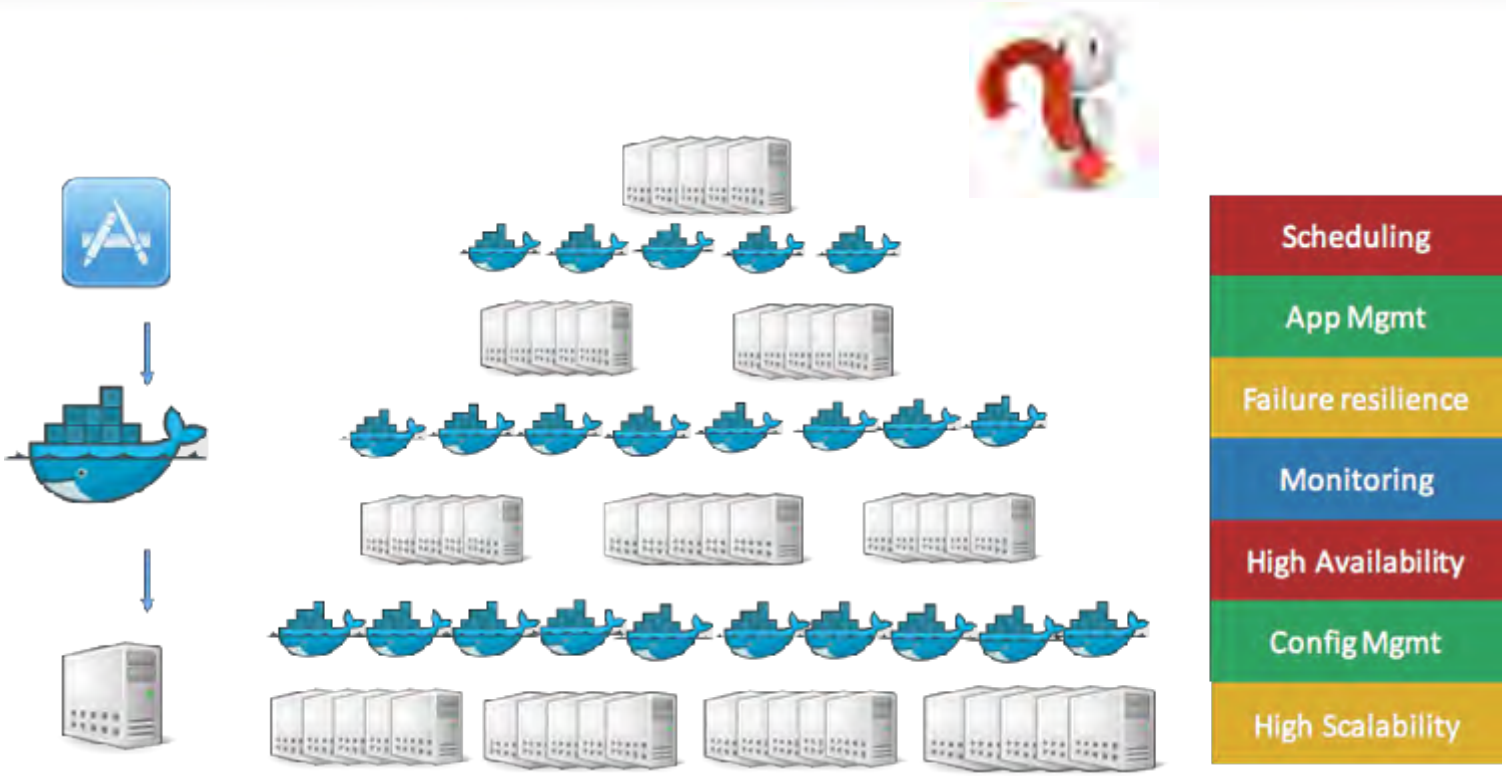
- lxc? lmctfy?
  - internal version of lmctfy exists long before lxc
  - lxc: no strong abstraction
    - namespace abstraction + ~raw cgroup
  - lxc: no programmable API
    - must be built to work with other tools
  - lmctfy: more abstraction and enhancement
    - e.g. subcontainer and asymmetric isolation
    - e.g. quality of service

```
lxc.utsname = cctc.csdn
lxc.network.type = veth
lxc.network.name = eth0
lxc.network.ipv4 = 10.2.3.5/24 10.2.3.255
lxc.cgroup.cpu.shares = 1234
lxc.cgroup.devices.allow = c 1:3 rw
lxc.cgroup.devices.allow = b 8:0 rw
lxc.mount = /etc/fstab.complex
lxc.rootfs = /mnt/rootfs.complex
lxc.cap.drop = sys_module mknod setuid net_raw
```

- OCI
  - Container: runtime + image
  - Based on Docker



Container?

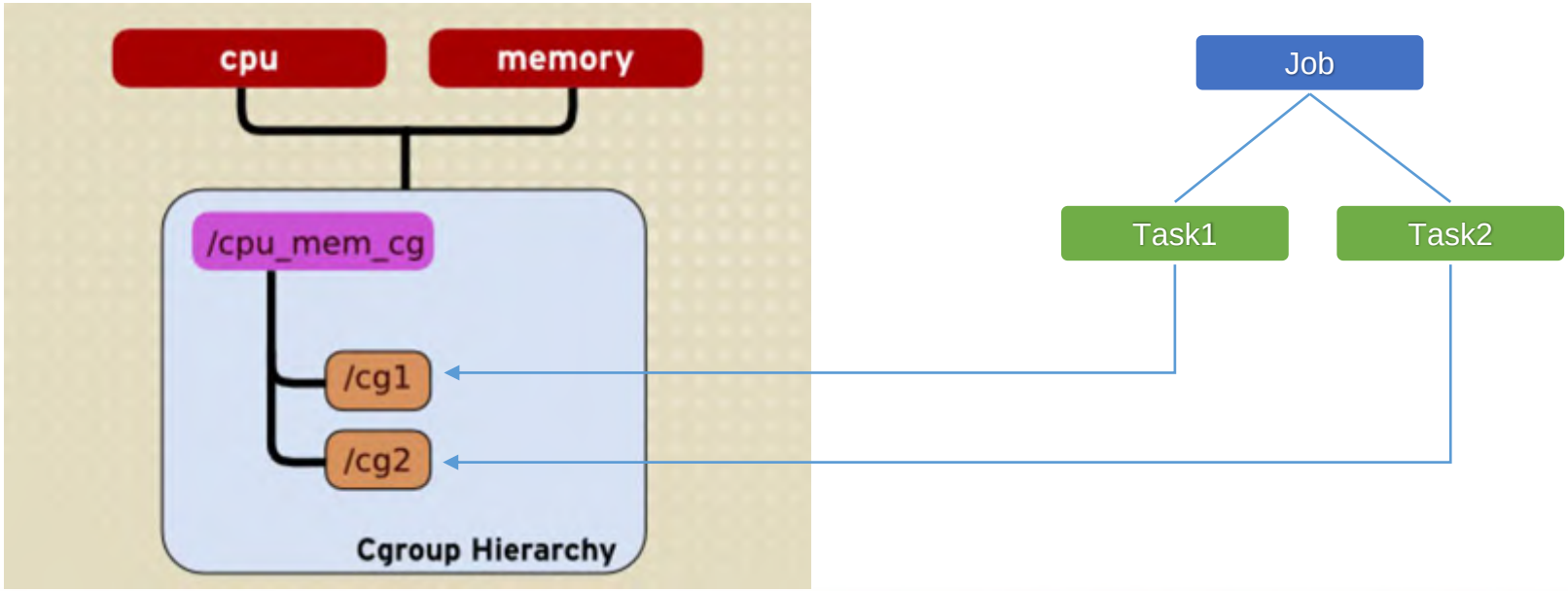


- Container management
  - clustering is the hard part
  - hundreds of engineerings building cluster management system

Borg: concepts

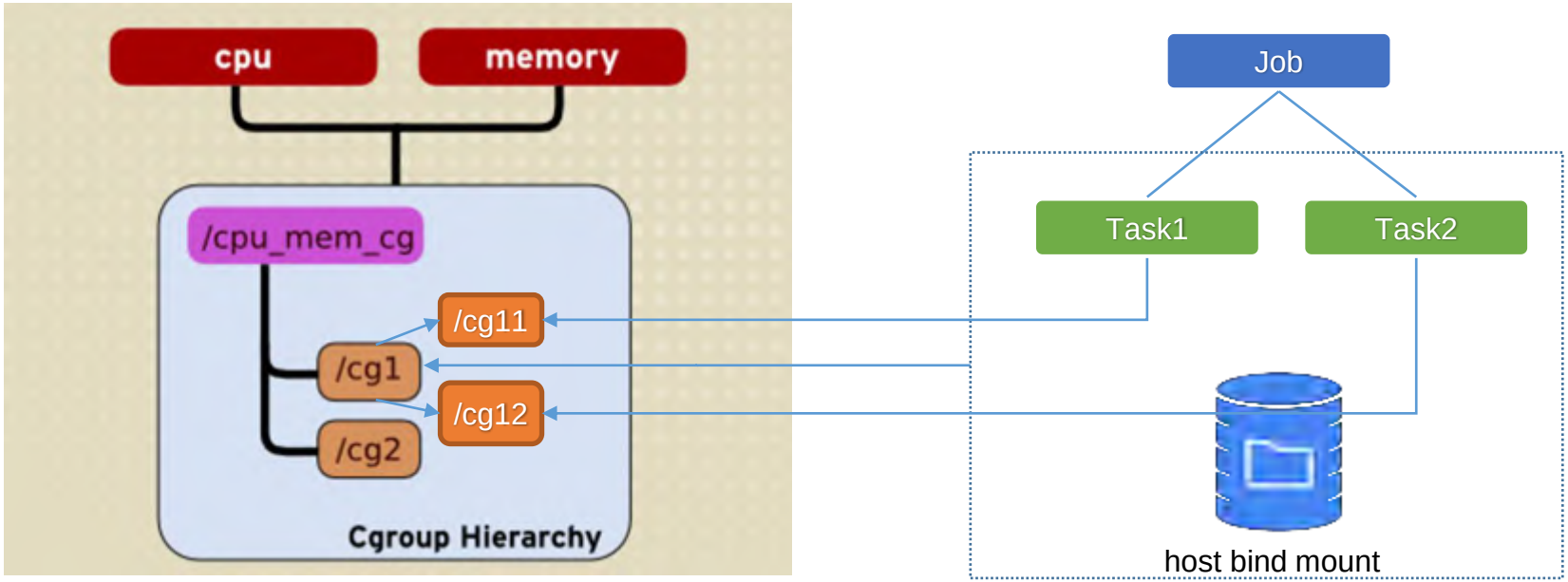
- Configuration
  - Borg configuration language
  - Most ‘hated’ language

- Job and Task
  - Job: unit of deployment, including resources requirement, number of tasks, etc
  - Task: unit of running entity
  - Resource container: the container to run task



Borg: concepts

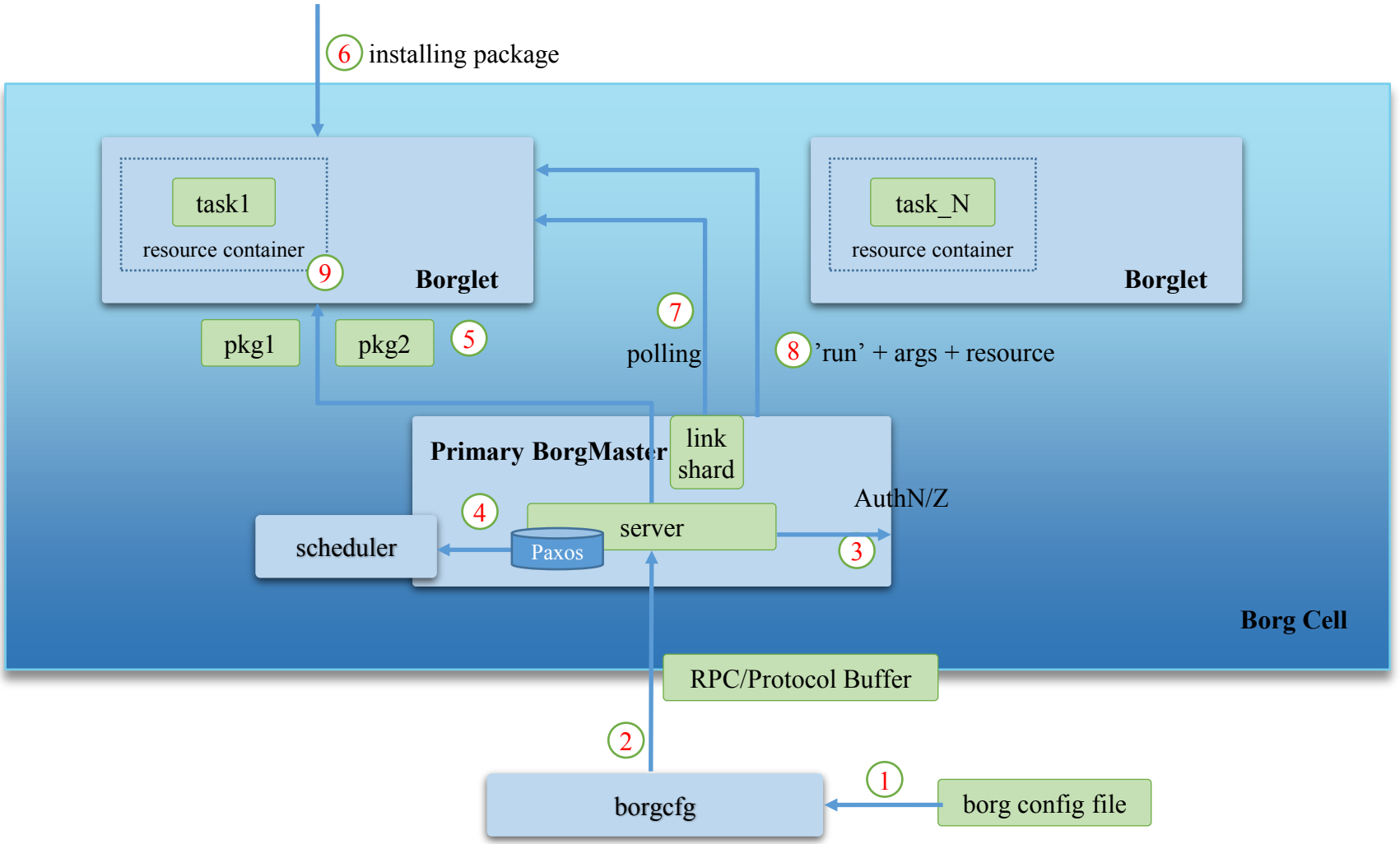
- AllocSet and Alloc
  - co-scheduling: e.g. logsaver and application
  - share resources for all tasks
  - persist data when tasks exit



Borg: concepts

- Scheduling
  - priority
  - resources: resource has priority
  - quota: quota has priority
  - packages: vs docker image
  - machine constraints
  - %ports% (BNS, chubby)
  - options:
    - repeated failures
    - quorum requirement
    - etc
- Runtime
  - application class
  - resource estimation:
    - decrease resource reservation (gradual decay function)

Borg: Request Lifecycle



Borg: The Good, The Bad, The Ugly

- 高稳定性、高自动化、高智能
- 极其复杂的配置文件

  最流行语言排行榜？
  Go, Python, Java, C++, Javascript, …

  最令人畏惧语言排行榜？
  Python, Borgcfg, Borgmon, Shell, …

- Borgmaster becomes borg monster
- People step on each other’s foot

Omega: 早年寄予厚望

- From Monolithic to Shared State
- Persistent Storage as the ground truth
- Look to the Future!
- Collaboration across the globe
- A bottom up driven project

Omega: 中年四面楚歌

- Engineering mayhem
  - Testing cell
  - API rewrites, all Borg callers have to change (along with behavioral change)!
  - Borgmaster is hard to rewrite

- 向现实妥协：
  - 项目进度
  - 与Borg的关系

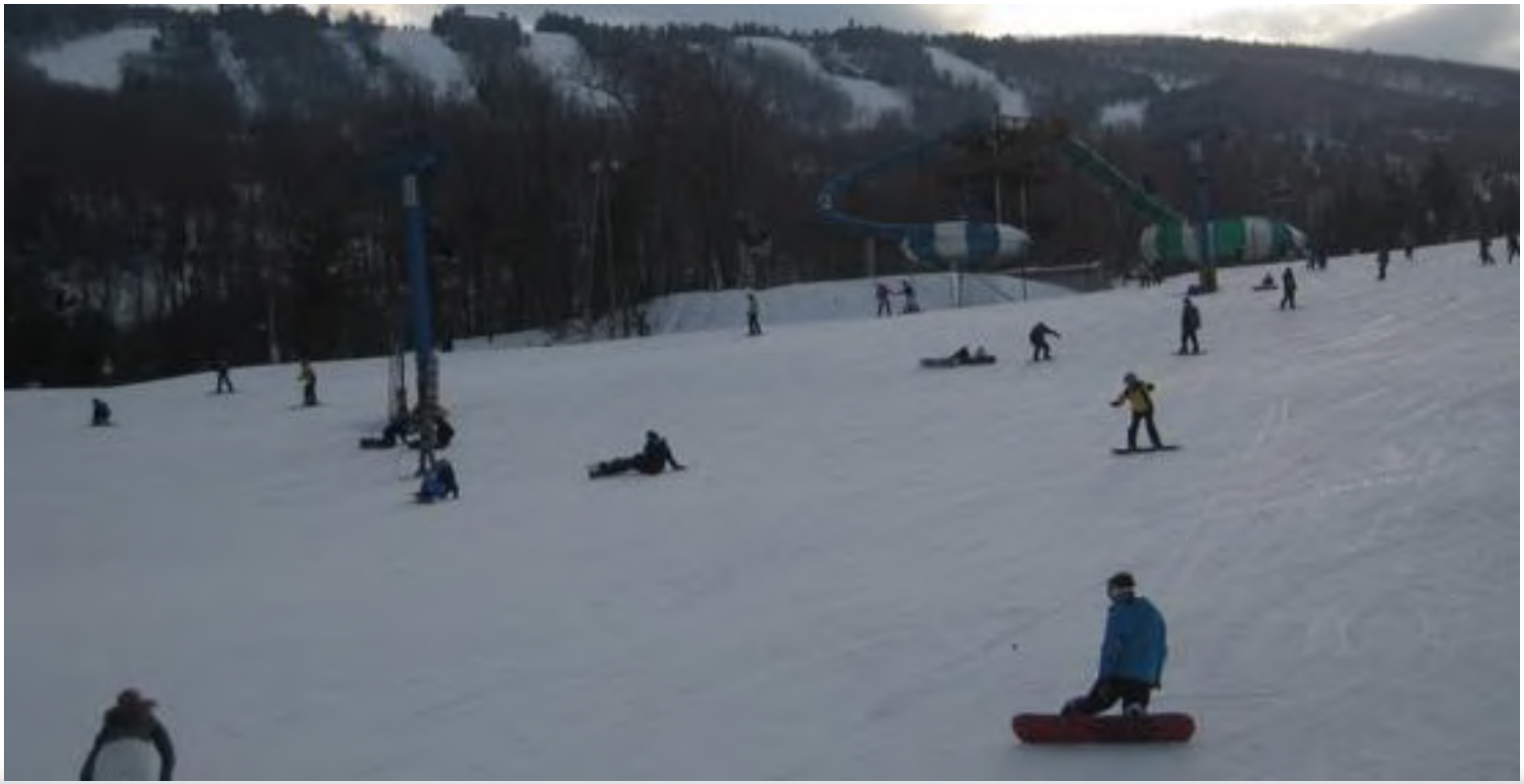
Omega: 晚年戛然而止

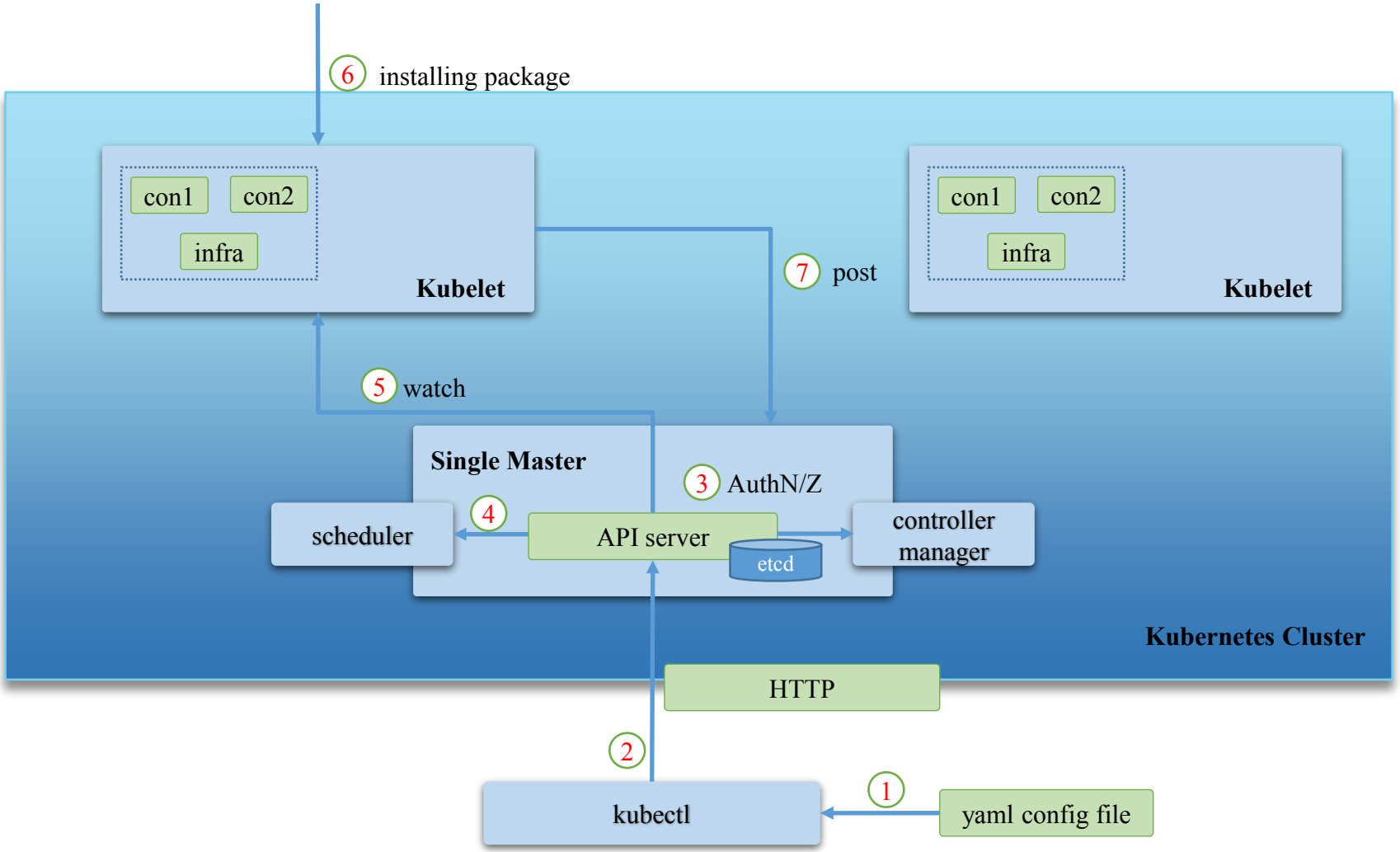
- 工程与理论的差距
  - 精确度 vs 吞吐量
  - 预测的困难
- Borg的进化
  - 性能
  - 软件工程
- 终究下架
  - Big shuffle

Cloud: The 'Urs'quake

- 经济上的巨大回报
- 技术上的巨大领先
- 产品化上的差异
- PaaS与IaaS的矛盾
  - GAE vs GCE
  - Managed VM

- 此处为何地?
  - Google Mountain View
  - Google Pittsburgh
  - Kubernetes birth place
- kubernetes乳名?

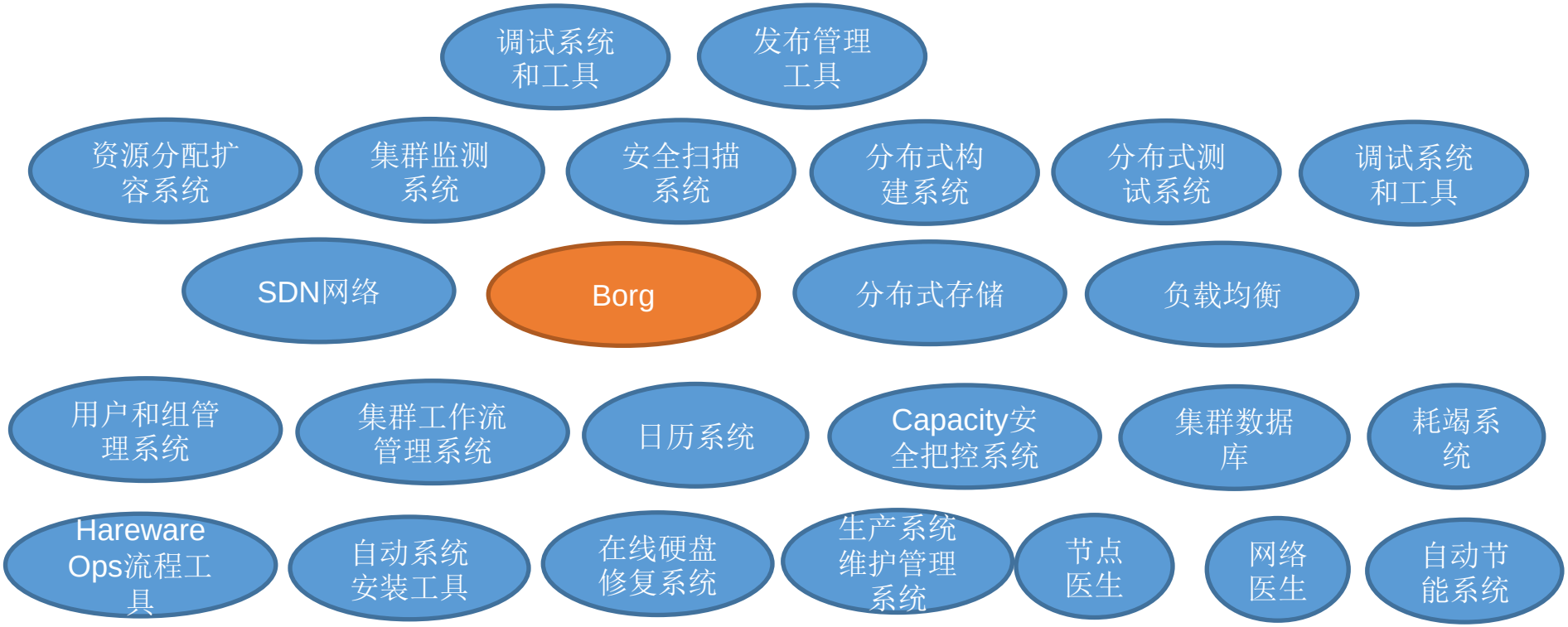



6 installing package
con1
con2
infra
Kubelet
7 post
con1
con2
infra
Kubelet
5 watch
Single Master
3 AuthN/Z
4
scheduler
API server
etcd
controller manager
Kubernetes Cluster
HTTP
2
kubectl
1
yaml config file

- 一个Cluster里有一个或多个Borg cell

- Borg容器之外的其他任务
  - Borg自身谁来管理?
  - 机器层面谁来管理?
  - 网络层面谁来管理?
  - 安全谁来控制?
  - 镜像如何管理?

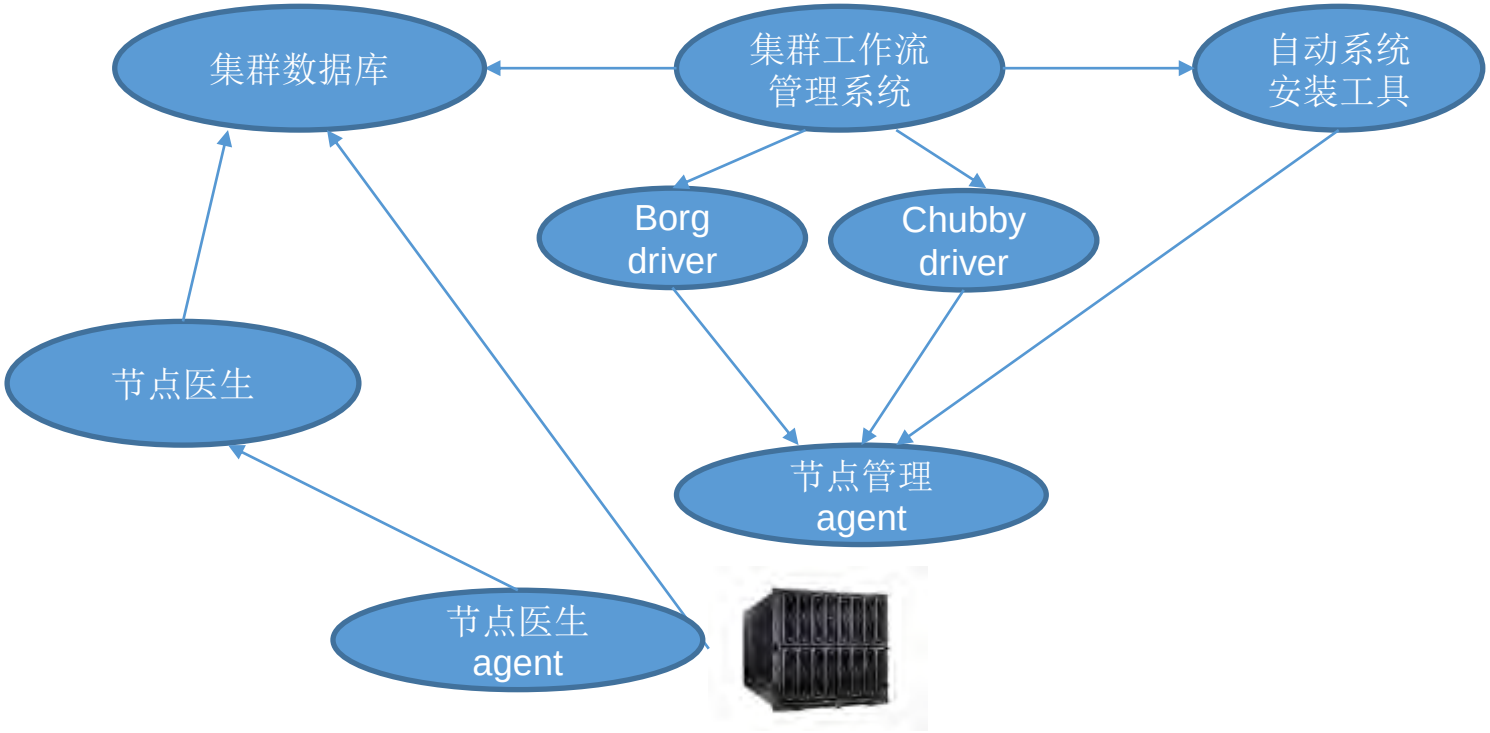
The entire family



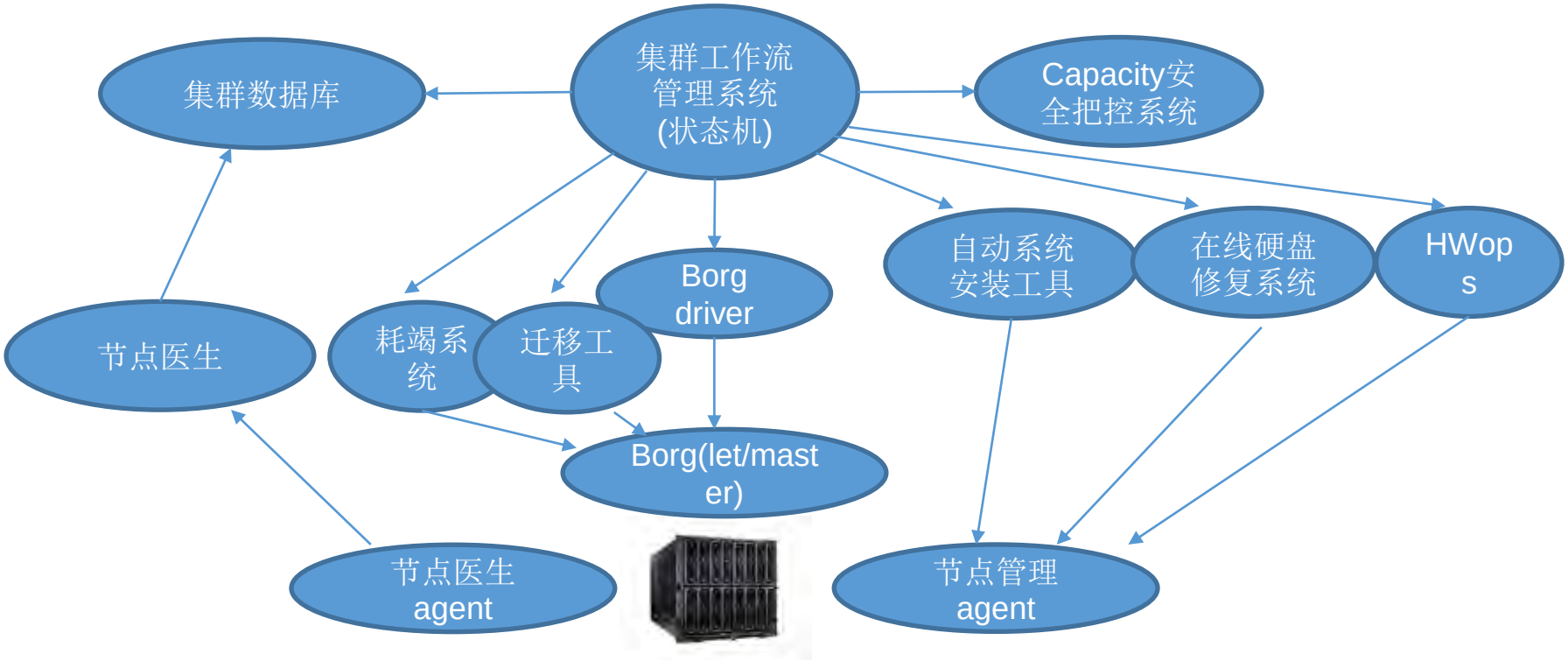
And ... The Robot

- Disk Erase
- Decommission
- Inventory Management

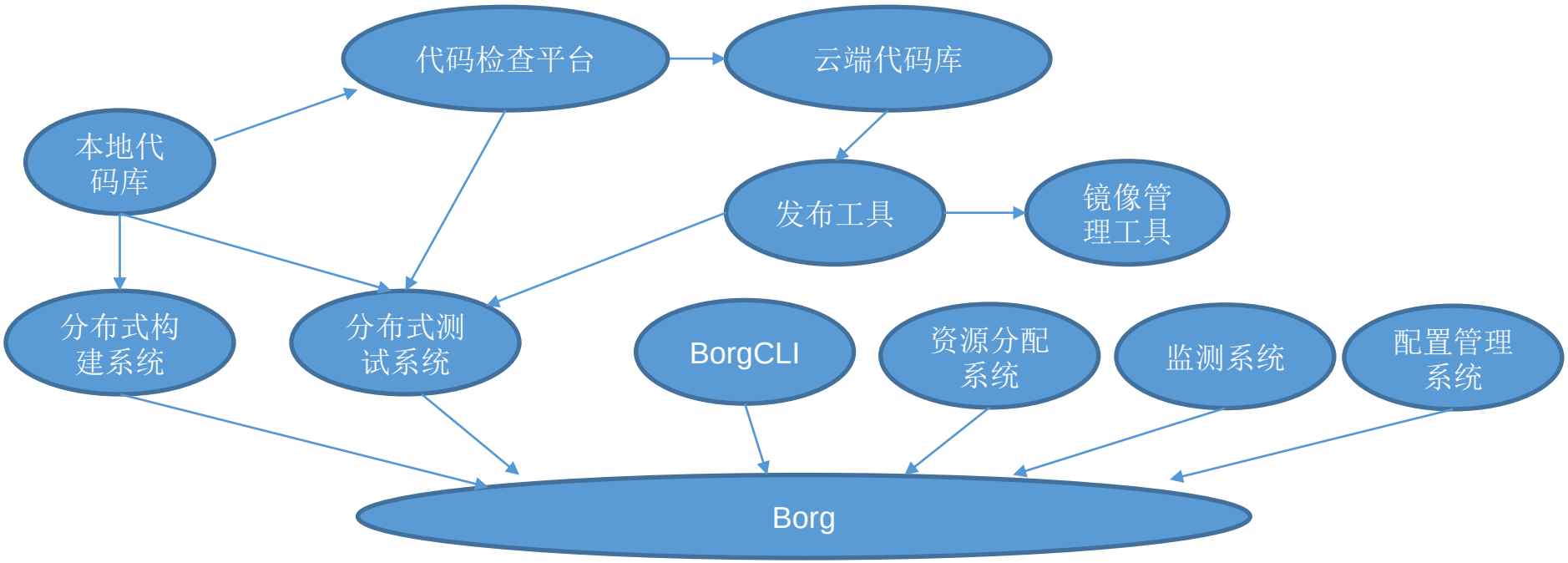
An example workflow: Pre-borg



An example workflow: with borg



An example workflow: more on the dev side



Thanks!