Google Borg

Nathan Pemberton
September 26, 2016

Boring Stuff

The What's and Why's

- **What is it for? ->** Sharing clusters among many users
  - Clusters are O(10k) nodes, some are "much bigger"
- **Why is it new? ->** Home-grown at Google, needed scale, backwards compatibility, and control
- **Why should you care? ->**
  - Led to Kubernetes (which you can download and use)
  - Google sees unique scale and complexity
  - Business/Practical perspective

Terminology

Cell	Individually managed cluster
Job	Run of a multi-node application
Task	Single-node component of a job
Alloc	Guaranteed resource slot on a node (per-user, can span jobs/tasks)
Alloc Set	Cluster-wide group of allocs (like a job)
BorgMaster	Central Manager, main point of contact for Borg
Borglet	Per-node executor (daemon)

Unique Opportunities

The Customer Isn't Always Right When You Sign Their Paychecks

- **No fair-sharing algorithms:** Quotas enforce company policy
- **Social Engineering:** Trainings and web-UI "hints" tell engineers what are good and bad shapes for jobs
- **Big Hammer:** When things go wrong (or just need to change), you can always apply brute-force to the problem

Livestock vs Pet

- Servers/tasks are Livestock, Not Pets
  - Server gets sick? -> Shoot it
  - More important task shows up? -> Shoot it
  - Configuration Changes? -> Shoot it
  - …
- Enabling Insights:
  - Everything must be fault-tolerant anyway!
  - Employees don't complain (as much)!

Priority Jobs + Backfill

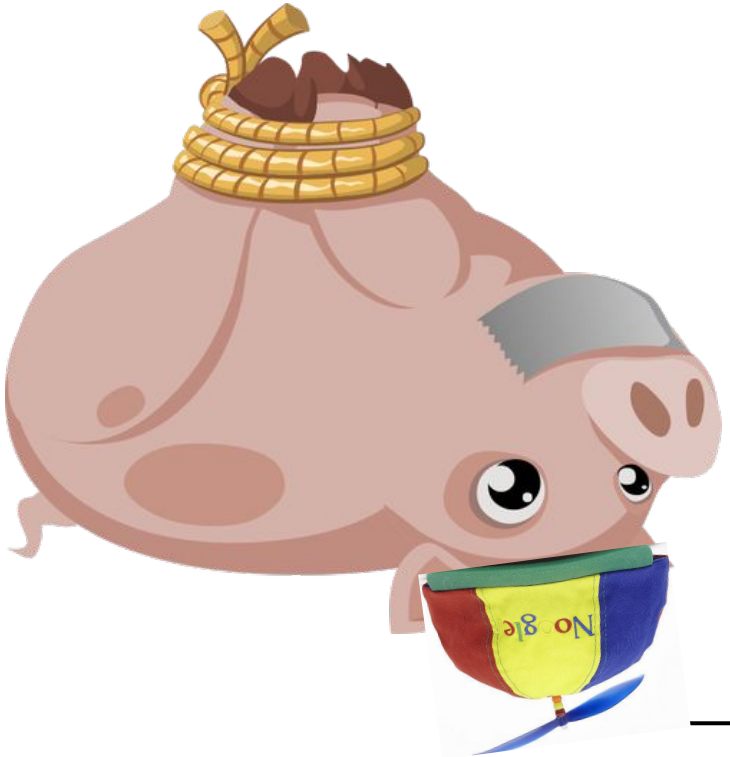
- Production vs Internal
  - “Prod Jobs” are customer facing and *must* work at all times
  - Enormous left-over compute for “batch” and low-priority jobs
- Latency-Sensitive vs Batch
  - Latency-sensitive tasks get better cgroup policies
  - Batch jobs oversubscribe resources more

Key Design Choices

Resiliency is King (99.99%)

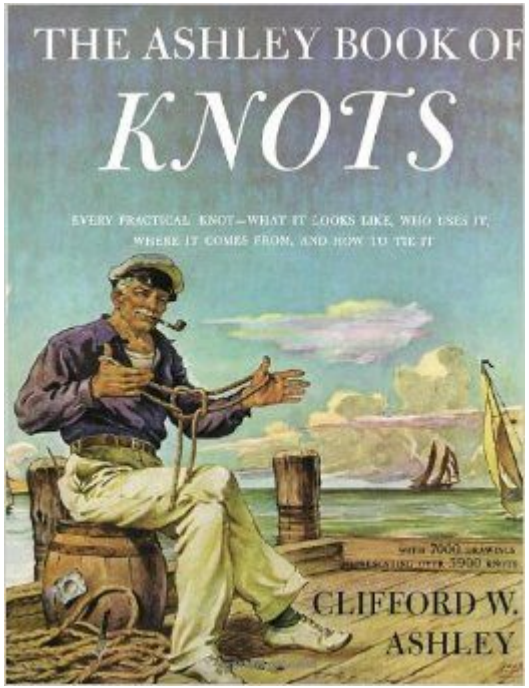
- **Declarative and Idempotent Commands**
  - Say how you want things to be
  - Keep saying it until it's true
- **Minimize Correlated Failures**
  - Many placement restrictions
  - Coarse-grained priorities to avoid "priority avalanches" (tasks keep preempting slightly lower priority tasks)
- **Highly Consistent Replication**
  - Everything important in multiple places
  - Heavy use of Paxos and Chubby for state

Enough Rope to Hang Yourself



- **Manual Override:**
  - Can hand-modify Borg checkpoint to work around issues/bugs
  - Per-task requirements
  - 1000+ line config files
- **Asterix City**
  - Almost every statement in paper should have a footnote (many do)
  - An exception for every rule

But a Very Good Knot Book

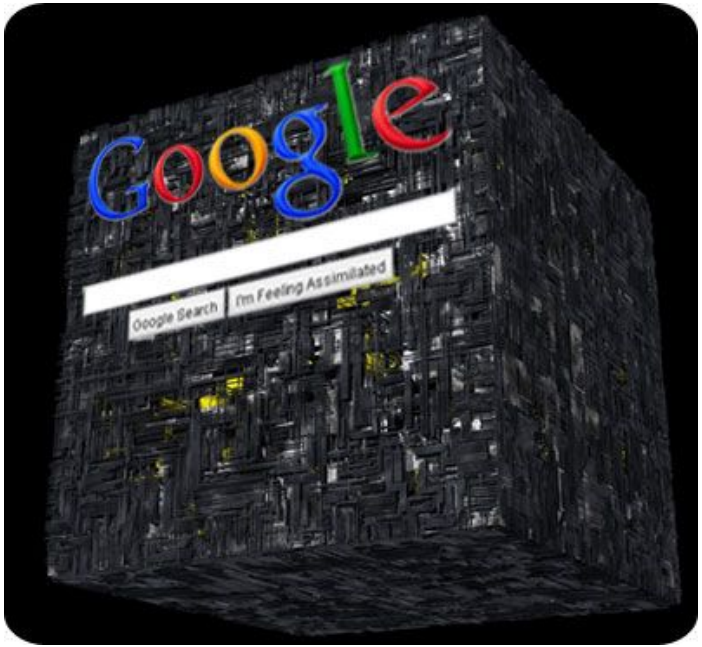


“If you aren’t measuring it, it’s out of control”

Dick Sites

- **Introspection is Key**
  - Yo dawg, I heard you like logs…
  - Most components export an HTTP server
  - Borgmaster aggregates task statistics constantly
- **Simulators and Tests Galore!**
  - Fauxmaster simulates new ideas and replicates bugs
  - Many small (<5k) clusters for trying out ideas
- **Fast rollback through checkpoints**

Conclusion



- Google Cheats (so take with a grain of salt)
  - Control their “customers”
  - Deep pockets
  - Unique workload
- But, they have some great insights
  - Fault-tolerance should be assumed
  - Good tooling and introspection solves many problems
  - Never assume you’ve covered all cases (backdoors can be useful)