Multi-host containerised HPC cluster
The new Docker networking put into action to spin up a SLURM cluster
The Bits and Pieces...
Docker

- Containers do not spin up their own kernel
  - All containers share the same host kernel
  - They are separated via Kernel Namespaces
  - Constrained using CGroups

Traditional Virtualisation

Containerisation
Docker

• Docker-Engine as a Container Runtime
  ▶ creates/starts/stops/manipulates/removes containers - all through RESTful API
  ▶ handles Namespaces, CGroups, IP connectivity, bind-mounts, etc.
Docker Networking

• Docker Networking spans networks across engines
  ▶ KV-store to synchronise (Zookeeper, etcd, Consul)
  ▶ VXLAN to pass messages along (macvlan might take over)
Docker Swarm

• Docker Swarm proxies docker-engines
  ▶ serves an API endpoint in front of multiple docker-engines
  ▶ does (simple) placement decisions.
Docker Swarm [cont]

[root@venus001 ~]# docker info -H docker1:2375 legrep --color=never "(192.168.12\.)" Containers: 50
Images: 374
docker1: 192.168.12.11:2376
  Containers: 5
venus001: 192.168.12.181:2376
  Containers: 12
venus002: 192.168.12.182:2376
  Containers: 6
venus003: 192.168.12.183:2376
  Containers: 4
venus004: 192.168.12.184:2376
  Containers: 4
venus005: 192.168.12.185:2376
  Containers: 4
venus006: 192.168.12.186:2376
  Containers: 5
  Containers: 3
venus008: 192.168.12.188:2376
  Containers: 6
CPUs: 76
Total Memory: 266.5 GiB
Name: a7d177accc81
[root@venus001 ~]# query docker-swarm
Introduce new Technologies
Introducing new Tech

credit: Russian Oil Claim
Introducing new Tech

Self-perception when introducing new tech…

credit: TF2 - Meet the Pyro
Introducing new Tech

… not always the same as the perception of others.

credit: TF2 - Meet the Pyro
Docker Buzzword Chaos!

- Distributions
- Auto-Scaling
- Orchestrations
- Solutions
  - On-Premise & OverSpill
  - production-ready
  - enterprise-grade
  - self-healing
• You say Docker - perception is ‘basically VMware’
  ▶ VMs were easy to shoehorn in legacy workflow, containers might break it
  ▶ spans environments: laptop, dev-cluster, staging, prod

• Not everyone is a unicorn, I do
  ▶ NOT want special distributions
    • useful for elasticity (AWS) and green-field deployment, not so much for a on-premise datacenter w/ legacy in it.
  ▶ want to leverage existing <stuff>
    • security (ssh infrastructure), user authentication
    • installation workflow, monitoring, logging
  ▶ want to keep up with upstream docker ecosystem
    • networking, volumes
    • features of docker-engine, -compose, -swarm
Reduce to the max!
• Leverage existing install/configuration workflow
  ▶ Kickstart + ansible-run

• Don’t focus on corner cases
  ▶ postpone snowflake-container (e.g. needs multi-tenant IB usage)
  ▶ User namespace: we get there (eventually)

• HPC environments assumptions
  ▶ single-tenant
  ▶ focus on performance
The Setup
Testbed

• **Hardware (courtesy of HPC Advisory Council)**
  - 8x Sun Fire x2250, 2x 4core XEON, 32GB, Mellanox ConnectX-2

• **Software**
  - Base installation
    - CentOS 7.2 base installation (updated from 7-alpha)
  - Ansible
    - consul, sensu
    - docker v1.9.1, docker-compose
    - docker SWARM
Docker Networking

• Synchronised by Consul

venus008

venus002

venus001
Docker SWARM

- Docker SWARM
  - Synchronised by Consul KV-store
• **SLURM within SWARM**
  - slurmd within app-container
  - pre-stage multiple container
  - spawn at job-start (pre-hook)
SLURM Cluster [cont]

```
venus001 rc=0 docker1:2375 provisioning (master) # docker exec -ti venus001/hpcgl sinfo
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
all* up infinite 8 idle hpcg[1-8]
odd up infinite 4 idle hpcg[1,3,5,7]
even up infinite 4 idle hpcg[2,4,6,8]
venus001 rc=0 docker1:2375 provisioning (master) # docker exec -ti venus001/hpcg1 srun -N8 hostname
hpcg7
hpcg3
hpcg2
hpcg6
hpcg4
hpcg5
hpcg8
hpcg1
venus001 rc=0 docker1:2375 provisioning (master) #
```
• Distributed Samza
  - Zookeeper and Kafka cluster
  - Samza instances to run jobs

Samza Cluster
Outta time!
• Using vanilla docker tech on-top of any distribution
  ▶ keep up with the ecosystem and prevent vendor/ecosystem lock-in

• 80/20 rule
  ▶ have caveats on the radar but don’t bother too much
  ▶ everything is so fast moving - it’s hard to predict

• Don’t scare away stakeholders
  ▶ KISS
  ▶ reuse workflow and infrastructure
  ▶ solution and not problem driven
eGalea Conference (Pisa)
20/21 April 2016


Q&A

http://qnib.org