Efficiently Develop Highly-Scalable Applications using Docker Containers

Live Demo

Thiemo Morth

https://github.com/mortht/t3c-demo

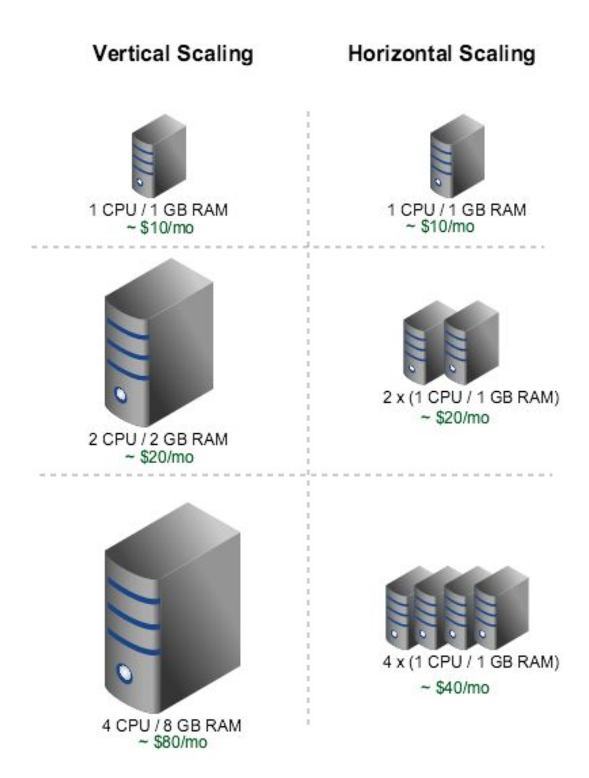
thiemo.morth@arhs-developments.com

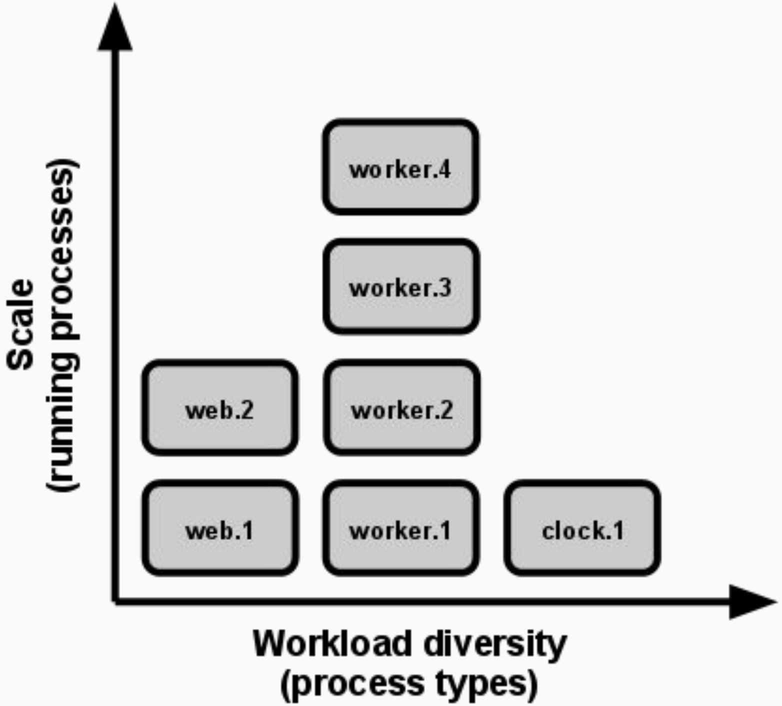
The Twelve-Factor App - Methodology

1. [...]
2. **Dependencies** - Explicitly declare and isolate dependencies
3. **Config** - Store config in the environment
4. **Backing services** - Treat backing services as attached resources
5. [...]
6. **Processes** - Execute the app as one or more stateless processes
7. **Port binding** - Export services via port binding
8. **Concurrency** - Scale out via the process model
9. [...]
10. **Dev/prod parity** - Keep dev, staging, and production as similar as possible
11. [...]
12. [...]

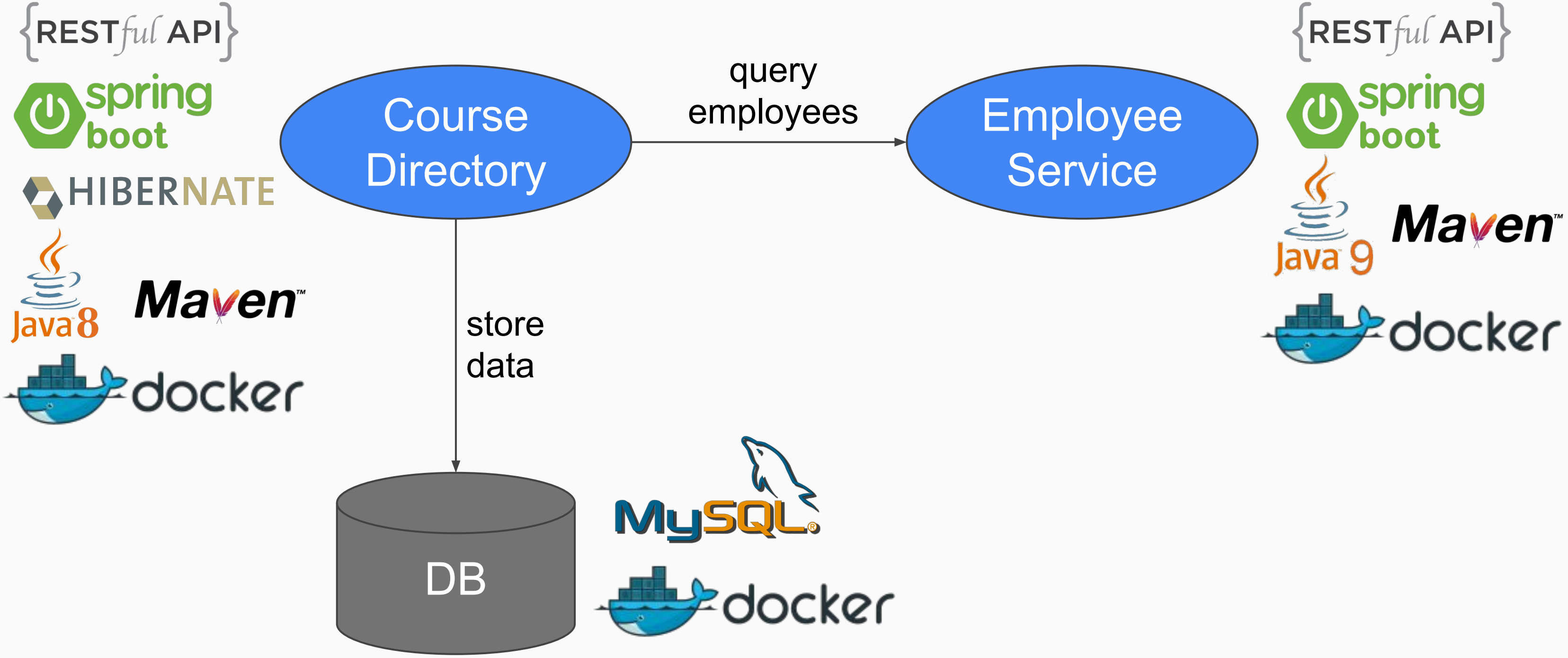
https://12factor.net

The Twelve-Factor App - Scaling





Demo Application - Course Directory



Course Directory

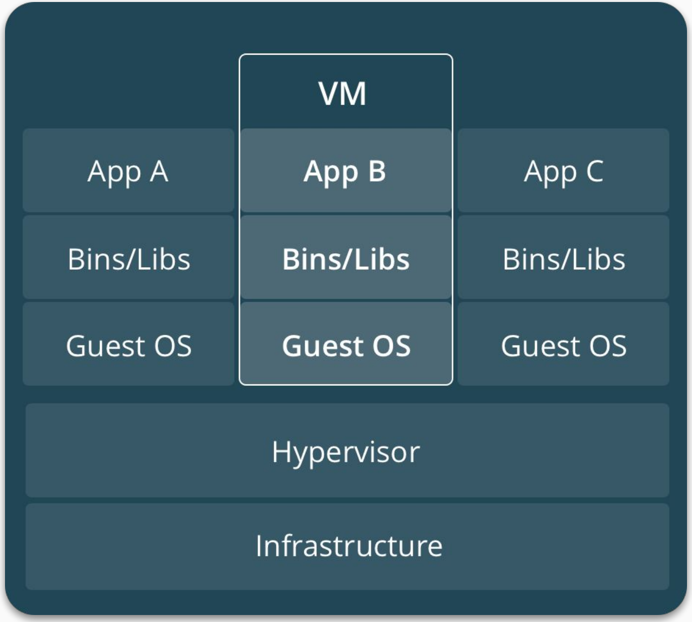
query employees

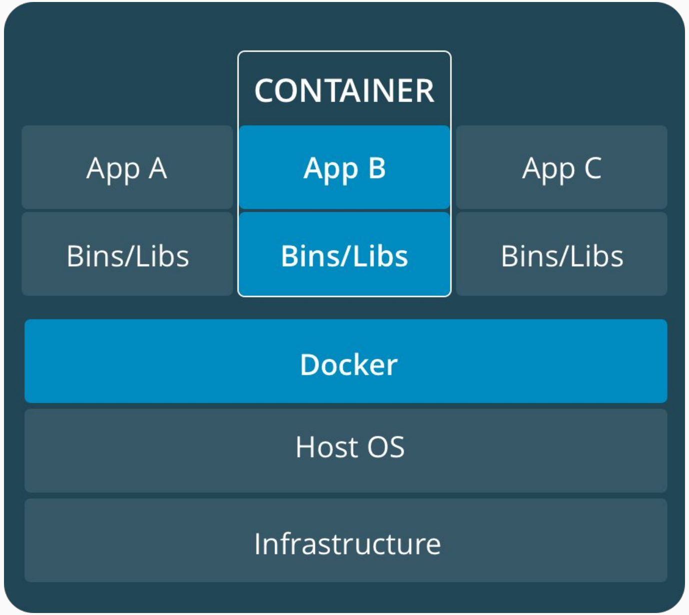
Employee Service

store data

DB

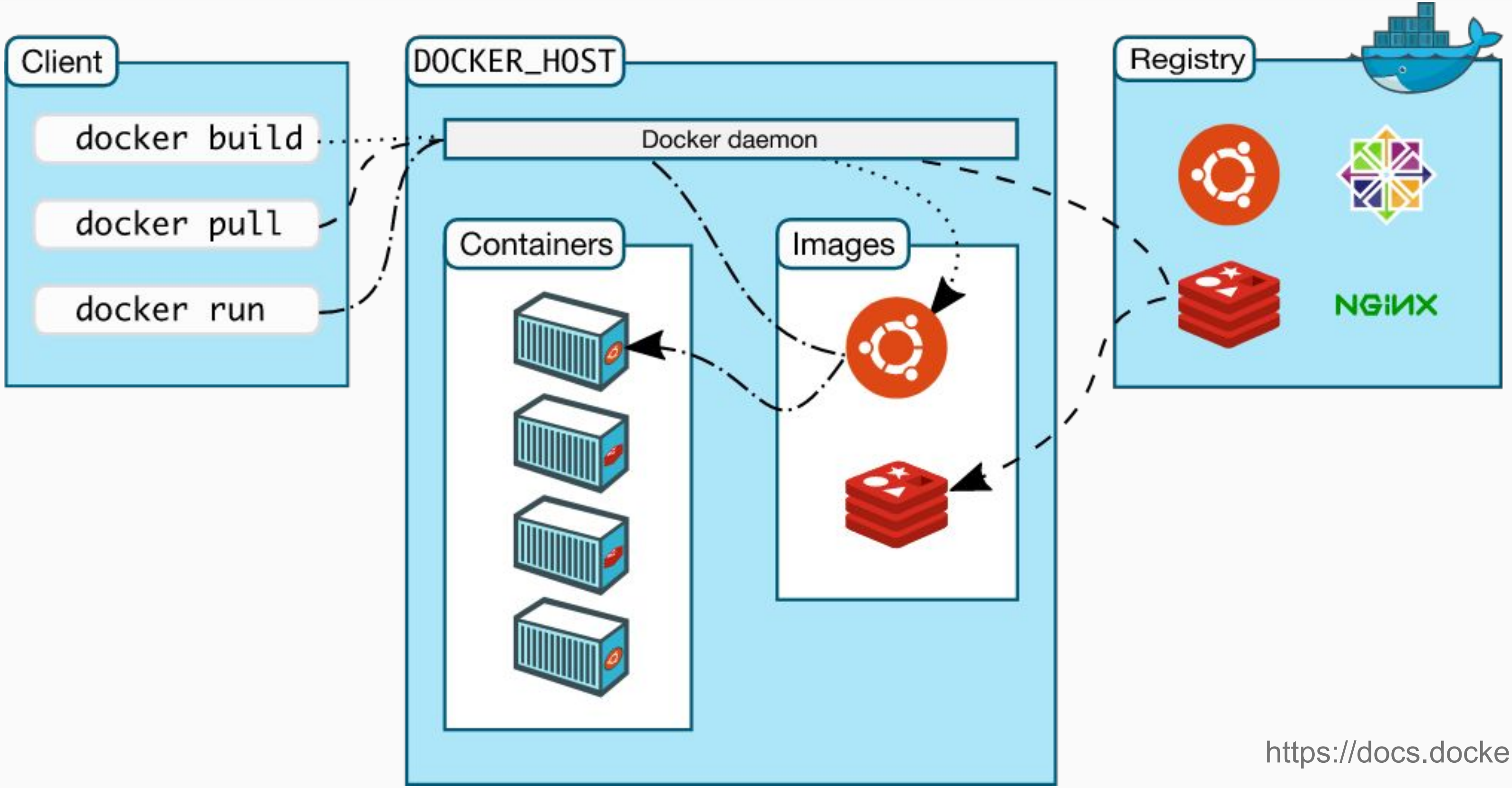
Docker - Container vs. VMs





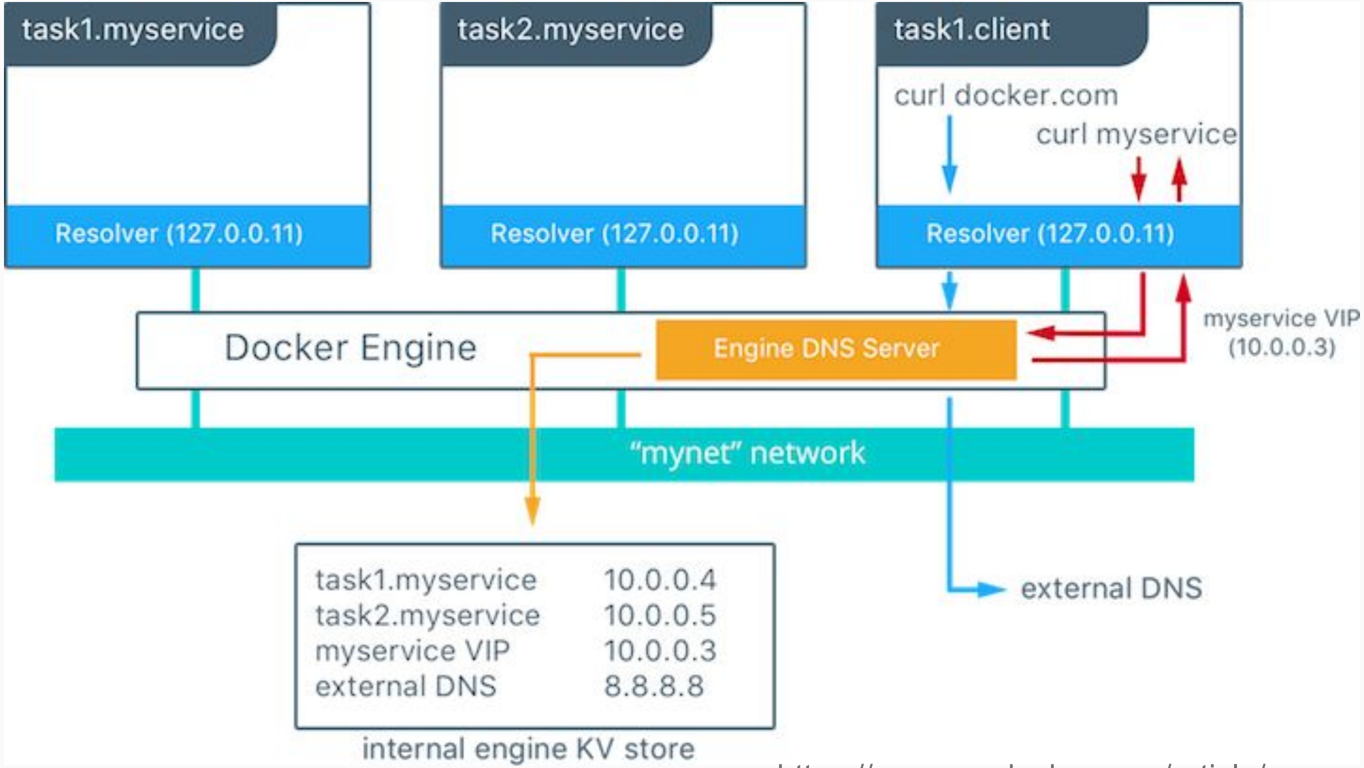
https://docs.docker.com

Docker - Architecture



https://docs.docker.com

Docker Swarm - Service Discovery and Load-Balancing

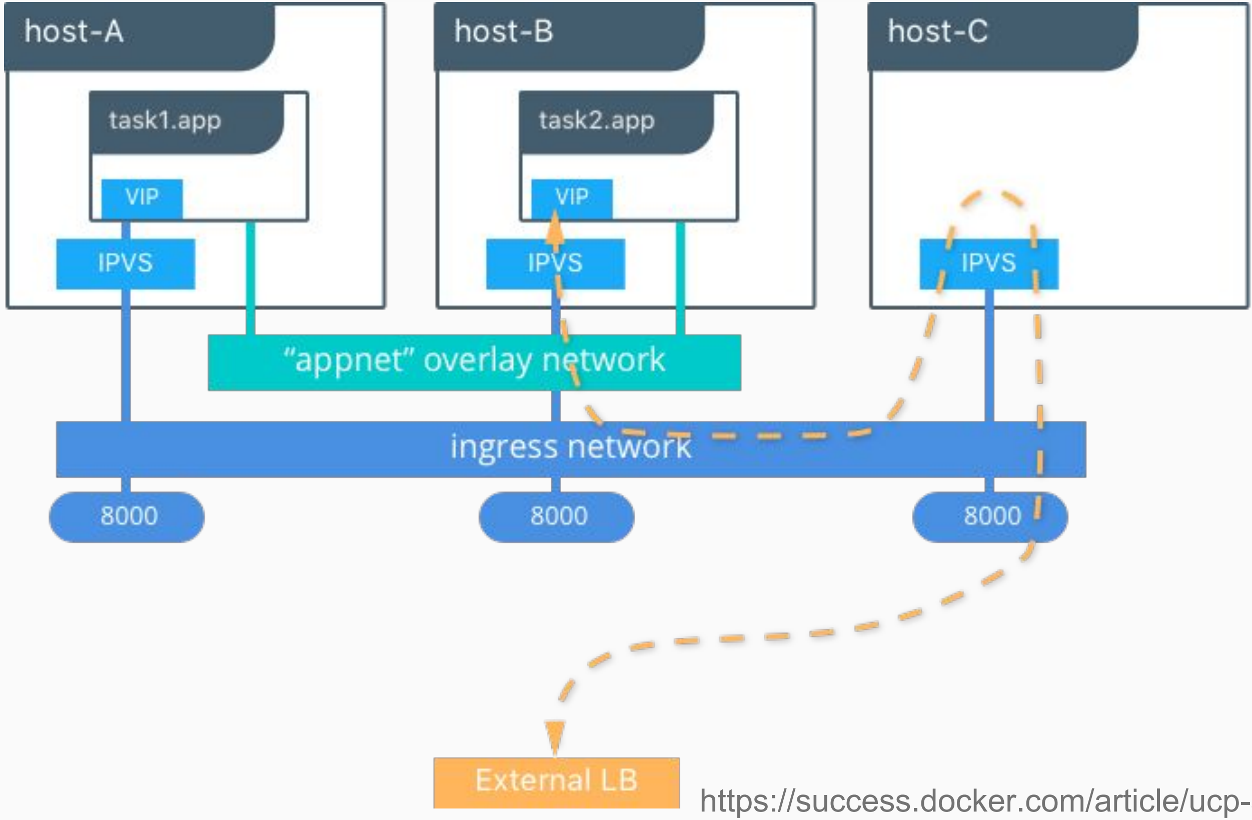


task1.myservice	10.0.0.4
task2.myservice	10.0.0.5
myservice VIP	10.0.0.3
external DNS	8.8.8.8

internal engine KV store

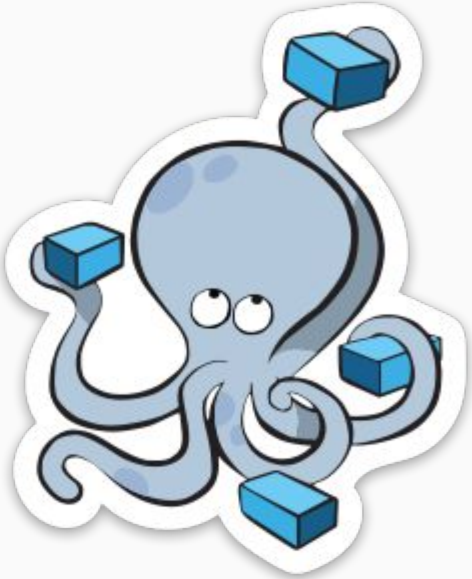
https://success.docker.com/article/ucp-service-discovery

Docker Swarm - Ingress Load-Balancing



https://success.docker.com/article/ucp-service-discovery

Questions ?



https://github.com/mortht/t3c-demo

thiemo.morth@arhs-developments.com