Configuration Management vs. Container Automation

Arnold Bechtoldt,
Johannes M. Scheuermann

Gent, 02.02.2016
Arnold Bechtoldt
Systems Engineer @ inovex (Germany)

› Software-Defined Datacenters (XaaS)
› Infrastructure as Code
› Continuous Integration/Delivery
› High Availability & Scale-Out
› DevOps & Consulting
Johannes M. Scheuermann
Student @ inovex (Germany)

› Software-Defined Datacenters (XaaS)
› Container (Docker, rkt)
› Orchestration (Kubernetes, Mesos)
› High Availability & Scale-Out
› OpsDev
Agenda

1. Motivation
2. Pitfalls & Lessons Learned
3. Use Cases
4. Conclusions
5. Q&A
Motivation

- Several backgrounds
- Different opinions
- Experience in small to large scaled enterprise environments
- (Some) beers and discussions
- Goal: Share ideas/lessons learned
Pitfalls & Lessons Learned
„Config Management sucks“ (Pitfalls)

› Software development „at its best“
  - VCS/refactoring/pair programming/CI/CD
  - Code quality is important

› Over-Engineering (Generic Code)
  - KISS

› „Configuration Management is Legacy (OS) Management!“
  - Lightweight containers (process isolation) to reduce OS management „overhead“?
Your Shipment of Fail has Arrived
„Containers suck, too“ (Pitfalls)

› „Docker is still a hype!“
  - Gives you lots of opportunities
  - Choose your tool stack wisely
  - Avoid „Bing Bang“ changes

› Lack of well-known/established open standards
  - Open Container Initiative
  - CNI (CoreOS) vs. CNM (Docker)
Containers suck, too“ (Pitfalls)

› „Docker security is a mess!“
  ▪ Basic understanding of cgroups/caps/ns are helpful
  ▪ Physical separation!
  ▪ Talk & work with your devs

› „Images on Docker Hub are insecure!“
  ▪ Just community contributions
  ▪ Docker images are packages/artifacts, treat them like VMDK/VHD/VDI/DEB/RPM
  ▪ Build your own (lightweight) (base) images
  ▪ Use base images without lots of userland tools if possible (e.g. Alpine Linux)
"Containers suck, too" (Pitfalls)

› Over-Engineered Dockerfiles ("/bin/bash")
  • KISS
  • Software/Process isolation
  • Replace large shell scripts with CM running outside the container

› Scheduling/Orchestration is a whole new area
  • KISS if possible
  • Step-by-step/Smooth migration
It doesn’t matter how many resources you have.

If you don’t know how to use them, it will never be enough
Use Cases
Setup

Private Cloud

Public

IaaS

PaaS/CaaS

VMs

Containers

Apps

CM

CM

CM?
Use Cases

› Continuous integration & delivery
› Microservices (rapid deployment)
› Blue/Green Deployment (immutability)
Conclusions
Conclusions

› Containers aren’t the holy grail, but enhance architecture
› Containers have lower overhead compared to VMs
› Containers and CM share the same problems
› Containers won’t work without Dev(+)Ops cooperation
› Containers and CM Systems will coexist 😊
We are hiring

http://www.inovexperts.com
Q&A