Ops hates containers. Why?

Martin Alfke - example42 GmbH
Customer meeting with Dev, Sec and Ops

Developers say:
• Ops are responsible for
  • base container image
  • running container
  • staging containers
  • container security
Security / Developer dialogue:

- **Security:**
  - “Dev must ensure security”
  - “Dev must name Kernel capabilities and CGroup settings”
- **Developers:**
  - “what is CGroup settings?”
- **Security:**
  - “We probably should meet when we all know the basics.”
Ops hates containers! Why? OSAD 2018 - Martin Alfke © example42 GmbH
Container Myths

- No need for configuration management - anywhere!
- Easier to build, deploy and run!
- Easier to test and verify!
- Easier to fix issues!
More container Myths

• No need to check status and health!
• No need to identify security!
• No need to login, no need for logs!
• No need for dedicated hardware, runs on cloud!
Container de-mystified

• It is just a change-root, delivered as a ‘package’

• Build steps are layers like VCS commits

• Containers need infrastructure

• Containers are managed like binaries
Container packaging and distribution

- People start with docker because it is easy
- `docker pull / docker run`
- like `curl -k | bash`
Simple can be harder than complex: You have to work hard to get your thinking clean to make it simple. But it's worth it in the end because once you get there, you can move mountains. - Steve Jobs
OPS, NET, SEC, DEV together can move mountains

- Mainframe
- PC
- VM
- Container

Image: example42 GmbH

Ops hates containers! Why? OSAD 2018 - Martin Alfke © example42 GmbH
• Uptime decreasing
• Maintenance increasing
OPS, NET, SEC, DEV
together can move mountains

- Staff does not scale with platform
OPS, NET, SEC, DEV together can move mountains

- 100% Automation!
OPS, NET, SEC, DEV
together can move mountains

• 80/20 - Pattern:
OPS, NET, SEC, DEV
together can move mountains

• 80/20 - Pattern:

• 80% time spending on 20% not automated
Operating containers
Understanding containers

- Short living instances
- 12factor (http://12factor.net)
- Persistent vs volatile data
Managing container runtime

- Automate Docker installation and configuration (puppetlabs-docker)
- Use Puppet Discovery to see what you have.
Containers for Ops

- CI/CD Pipelines (like CD4PE or Puppet Pipelines or GitLab CI)
- Build Processes
- Dashboards
- Puppet Infrastructure(!) (https://github.com/puppetlabs/pupperware/)
Container orchestration
Understanding container orchestration

- Multi node container runtime
- Orchestration
- Network (Egress / Ingress / Proxy)
- Maintenance
Managing container orchestration

Installation and configuration of an application stack.

Choose one of the options:

• puppetlabs-kubernetes
• puppetlabs-docker_uccp
• puppetlabs-docker_ddc
Commercial container orchestration

• Self hosted vs Managed

• everybody does K8s?
Other container orchestration tools

- Mesos/Aurora/Marathon
- Titus (Netflix)
- Docker Swarm
- Nomad/Terraform
- CoreOS / rkt
Everything is in containers. Is there something missing?
Containers and Configuration Management

- Where do you run your databases?
- Can you move everything to containers?
- What about legacy and enterprise applications?
Containers and Monitoring

- Dynamic Resources need dynamic monitoring solution
- Global platform and service health
- Some monitoring tools for containers: sysdig, cAdvisor, Puppet Discovery, Prometheus
Containers and Hardware

- Serverless does not mean no hardware
- Opsless does not mean no Ops
- Check with finance (CAPEX vs. OPEX)
Conclusion
Recommendations

• Containers adoption increases.

• Ops people: start learning, stop play and complaining.

• Security first - even on PoC

• Automate everything

• Choose wisely your container tools and environments.
Ops hate not having full control of their systems. Not containers.

Martin Alfke - example42 GmbH