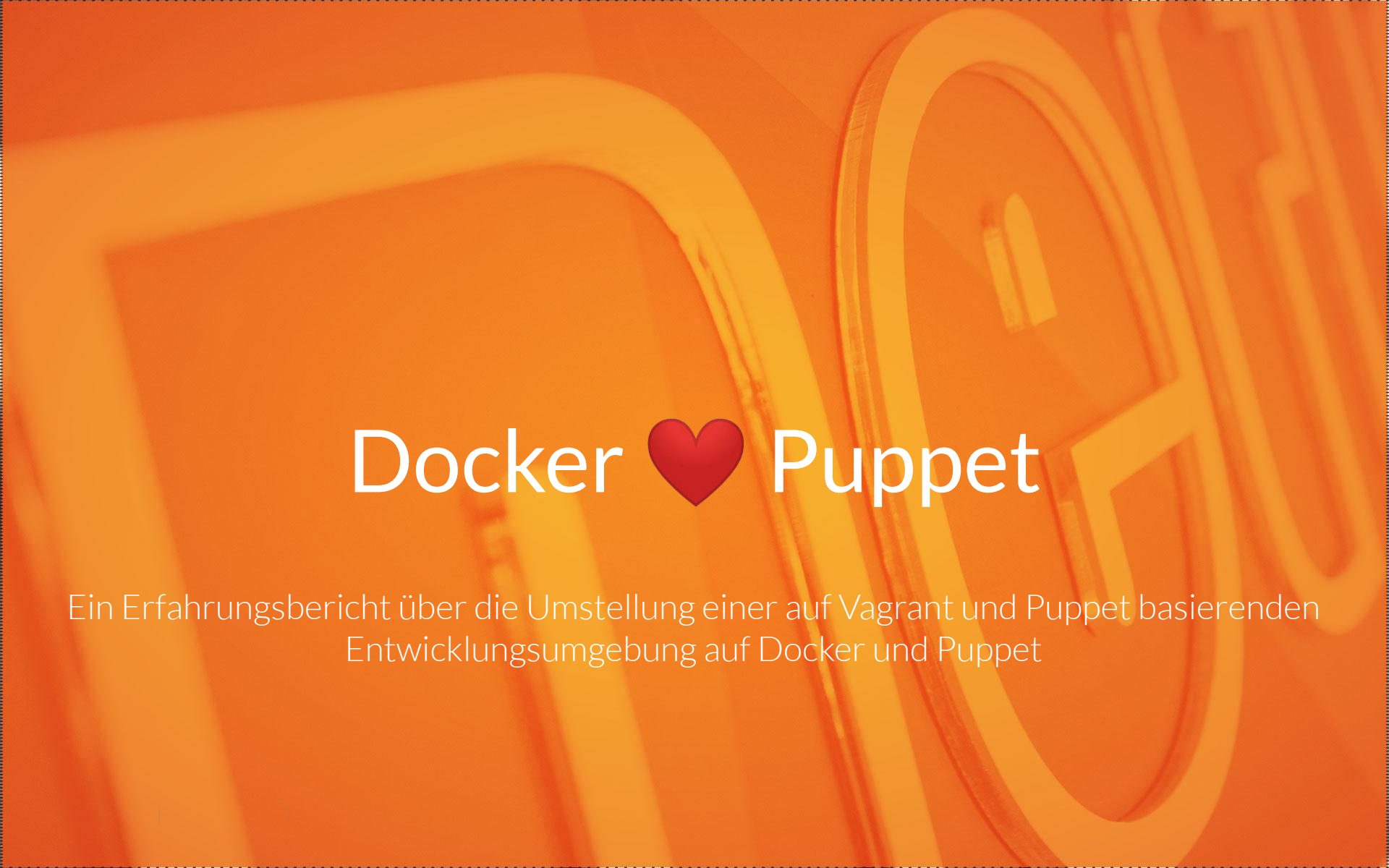

Docker ❤ Puppet
Ein Erfahrungsbericht über die Umstellung einer auf Vagrant und Puppet basierenden Entwicklungsumgebung auf Docker und Puppet

About me

- Lienhart Woitok
- Seit über 10 Jahren bei netlogix im Bereich DevOps tätig
- 10 Jahre Erfahrung mit Puppet, aber wenig Erfahrung mit Docker
- Keine sozialen Medien – <lienhart.woitok@netlogix.de>

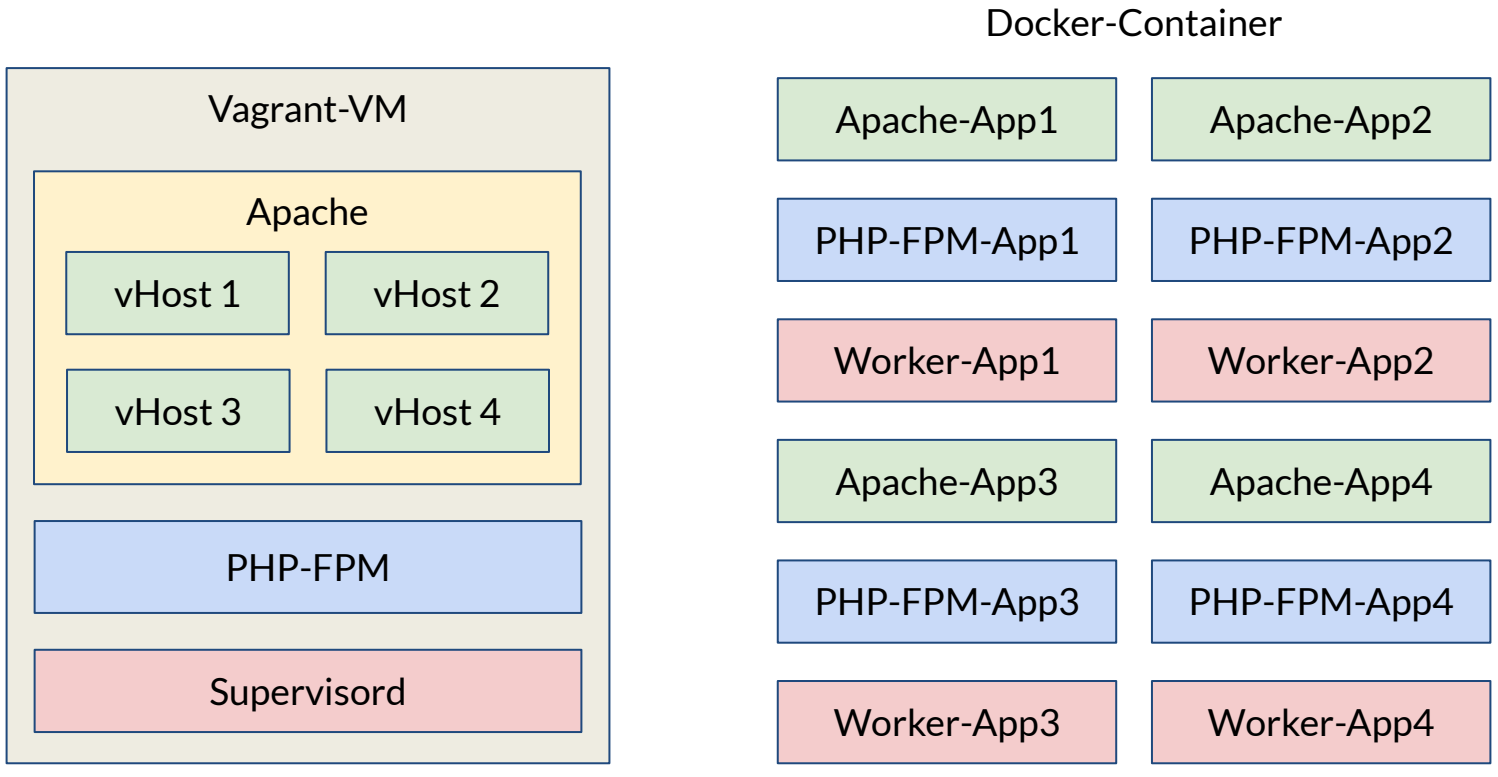
Das Projekt

- PHP-Applikation (Framework: Neos Flow)
- 16+ unabhängige Instanzen
- Pro Instanz:
  - Webserver mit PHP
  - min. 1 Worker
- Zentrale Dienste:
  - MySQL
  - Redis
  - RabbitMQ

Ausgangslage

- Vagrant
- Puppet
- Probleme:
  - langsam
  - “spezielles” Setup – nicht extern nutzbar

Der Plan



1. Versuch

- Klassische Dockerfiles
- Schnelle erste Erfolge
- Probleme:
  - viel Boilerplate-Code
  - viel duplicate Code
  - Logik in Shell-Scripten

2. Versuch

- Docker + Puppet, geht das?
- Puppet in laufenden Containern? Nein!
- puppetlabs/image_build ist tot
- Packer kann Docker und Puppet

Vorteile von Puppet

- Code-Reuse
- Abstraktion von Komplexität
  - durch Kapselung in Klassen und Defines
  - über Image-Grenzen hinweg
- Vorhandene Logik nutzen

Funktionsweise

- Packer ...
  - startet Docker-Container
  - installiert puppet
  - invalidiert evtl. vorhandene Zertifikate
  - führt Puppet aus
  - entfernt Zertifikate
  - speichert Container als Image
  - taggt das Image
  - pusht das Image

Showtime!

Code

https://github.com/netlogix/docker-puppet-demo

Fragen?

Diskussion: wie macht ihr das eigentlich bei komplexeren Projekten?


netlogix