PhraseApp

The Long Road from Capistrano to Kubernetes

Tobias Schwab, Co-Founder of PhraseApp

Slides: http://bit.ly/cap-to-kube

How to deploy Ruby on Rails?

Deploying Ruby on Rails

- required on all servers:
  - OS + system packages
  - ruby runtime in specific version
  - gems in specific versions + system packages
  - nginx reverse proxy
- deployment:
  - copy source code to all servers
  - parallel restarts on all servers (zero downtime)

2007: Capistrano

Capistrano

Framework and utility for executing commands in parallel on multiple remote machines, via SSH. The primary goal is to simplify and automate the deployment of web applications.

- one directory per release
- current symlink
- hooks
- rollbacks

2007: Capistrano

- 3 bare metal servers, SMB hoster
- deployment with capistrano
- manual config and library management
- **pros:** simple, easy to understand
- **cons:** hardly any automation

2010: Bundler

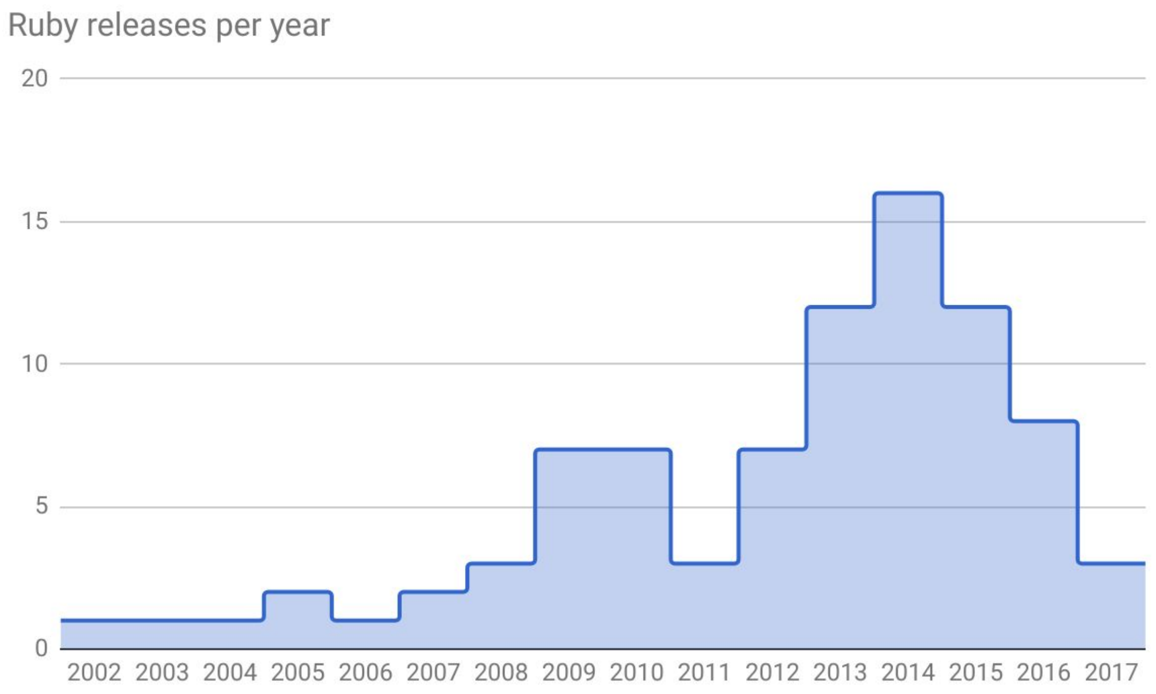
2010: Bundler

- ~ 10 bare metal servers at telefonica/o2
- library (gem) management with bundler
- bundle install via capistrano hook on every deployment
- config management with versioned shell scripts
- **pros:** semi-automated config management
- **cons:** system state hard to manage with shell scripts

The problem with convenience

- creating, adding and updating gems is easy
- more gems => less code to write
- more gems => more gems to maintain
- more gems => more system packages necessary
- more system packages => more demand for good config management

The problem with popularity

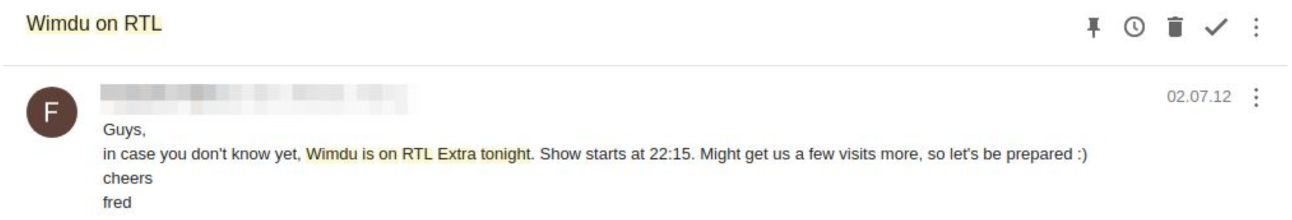


2012: Hetzner

2012-01: Hetzner

- first PhraseApp deployment
- single hetzner host
- manually installed ruby version and system packages
- **pros:** good enough, “let’s fix things if we need to”
- **cons:** “not important”

2012: Scaling Rails



Wimdu on RTL

F [redacted] 02.07.12

Guys,
in case you don't know yet, Wimdu is on RTL Extra tonight. Show starts at 22:15. Might get us a few visits more, so let's be prepared :)
cheers
fred

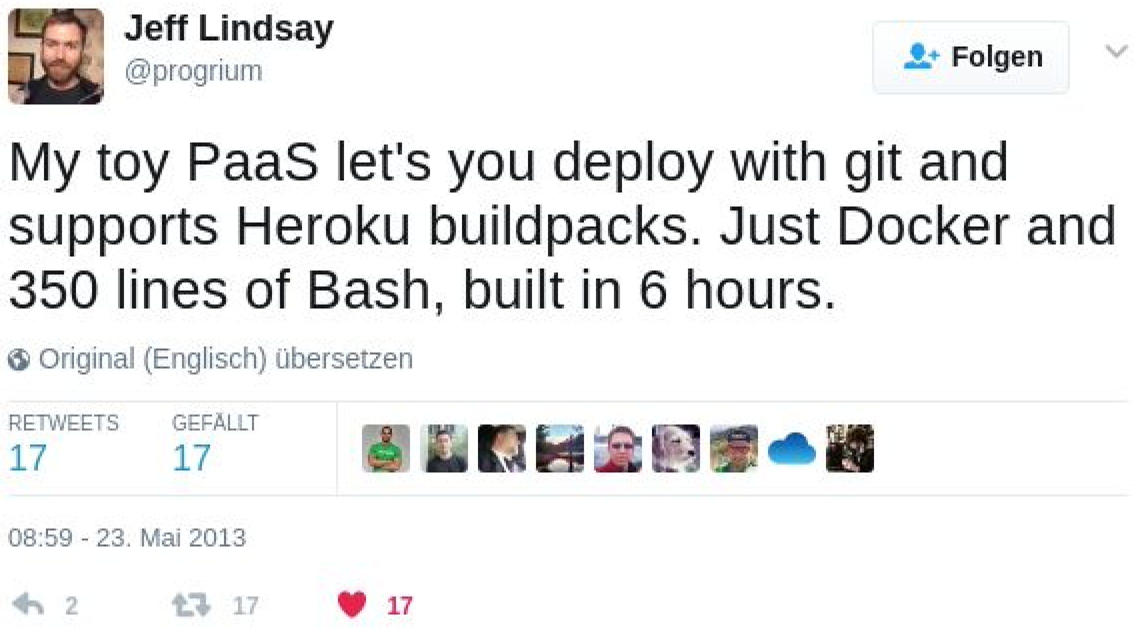
2012-07: Scaling Rails

- hybrid cloud on Rackspace: bare metal + VMs
- config management with chef
- custom HAProxy
- **pros:** automated config management, suitable for fire fighting
- **cons:** learning curve for chef, long bootstrapping times, capistrano difficult with changing hosts

2013: Immutable Infrastructure

2013-01: Immutable Infrastructure

- replacing chef-server based deployment
- golden master AMIs on AWS
- linear shell script to create AMIs, no config management
- deploying by replacing full instances
- **pros:** stateless, immutable, scalable
- **cons:** slow deployments, custom deploy code



Jeff Lindsay
@progrium

My toy PaaS let's you deploy with git and supports Heroku buildpacks. Just Docker and 350 lines of Bash, built in 6 hours.

Original (Englisch) übersetzen

RETWEETS 17 GEFÄLLT 17

08:59 - 23. Mai 2013

Docker

- OS + packages + libraries + code in one image
- API
- immutable infrastructure
- reduce deployment times
- moves dev closer to prod

2013: Docker

2013-09: Docker

- data protection requirements
- static VMs on “not so good” but german hoster
- deployment
  - docker image per deployment
  - create new containers via API
  - update HAProxy after
  - terminate old containers
- **pros:** immutable infrastructure on static VMs, fast
- **cons:** 100% custom deployment code

2014: AMIs, Cloudformation, ASGs

2014-01: AMIs, Cloudformation, ASGs

- deployment
  - create Golden Master AMI
  - update AutoScaling Group
  - scale up and scale down to replace instances
- **pros:** rock solid, not much custom code, transparent
- **cons:** slow deployments, chaotic rollbacks

2015: Wunderproxy

2015-03: Wunderproxy

- reverse proxy with API for container management
- config on S3 picked up in ASG Userdata
- deployment
  - create docker image
  - create new containers via API
  - update reverse proxy
  - config update and terminate old containers
- **pros:** faster deployments, simpler rollbacks
- **cons:** custom code, stuck with legacy docker version

2016: ECS vs. Kubernetes

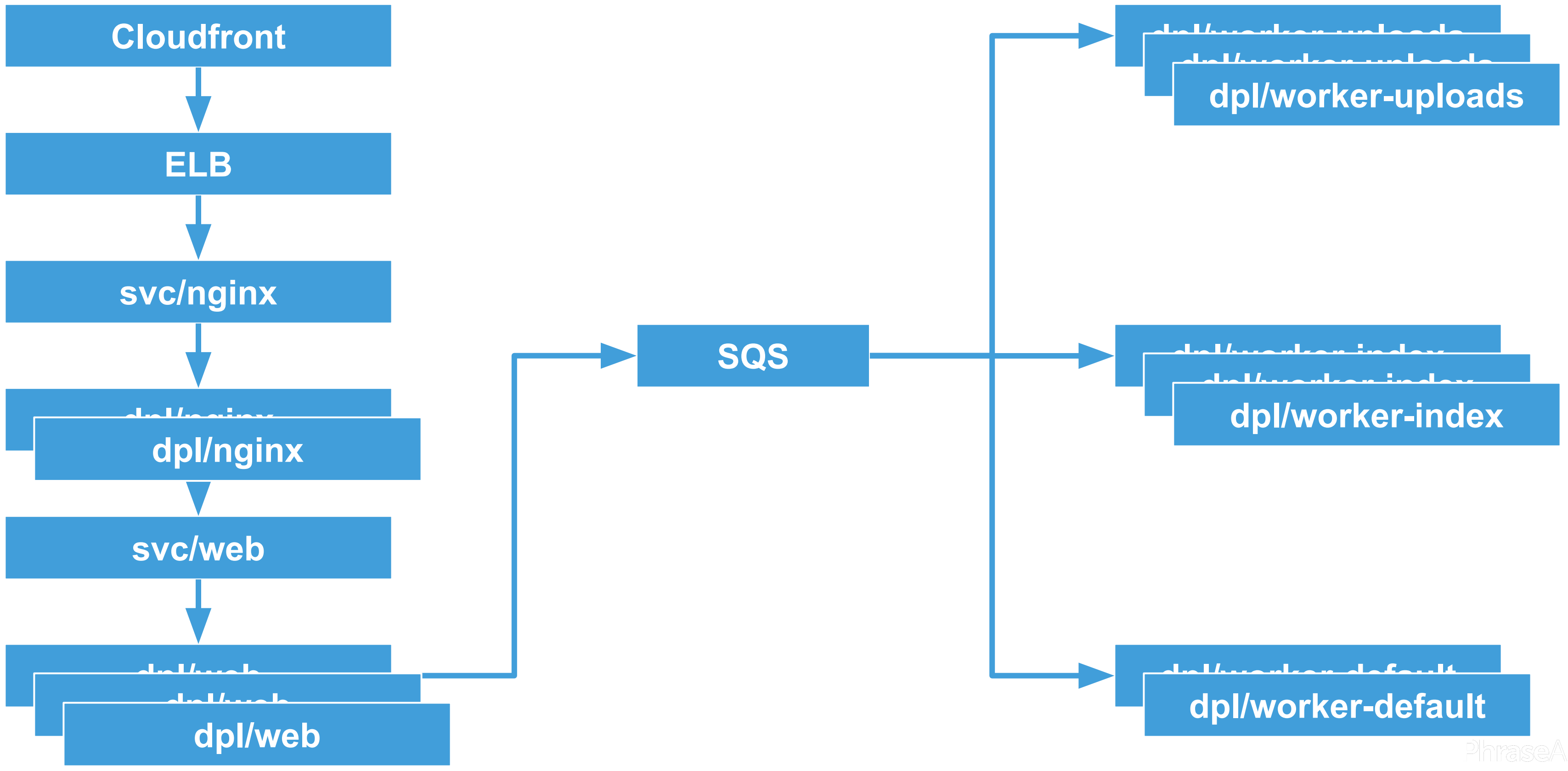
PhraseApp goes Kubernetes

- 2016-03: begin of evaluation of ECS, then Kubernetes
- 2016-05: initial commit to kc
- 2016-06: internal growth app deployed with kc on kubernetes
- 2017-02: PhraseApp deployed with kubernetes

PhraseApp: Architecture

- monolithic Ruby on Rails application
- MySQL, ElasticSearch, Redis
- asynchronous Jobs
- cronjobs

Architecture PhraseApp



kc

(our) generic way for building and deploying applications on kubernetes

kc build

- Dockerfile or build script in repository
- build image in separate build pods
- automatic builds triggered by master changes
- build caching via dedicated build nodes
- support for all container registries (Google, ECR, etc.)

kc deploy

- allow updating multiple deployments at once
- wait for deployments to finish
- hooks
- multistage:
  - `kc prod deploy`
  - `kc staging deploy`

Hooks

- configure in .kc file
- executed with new image with current ENV in separate container
- pre-check with prompt
- abort on failure
- allow e.g. database migrations

Cronjobs

- triggered by jenkins (running in k8s cluster)
- *`kc run`*
- each job creates new container with current image and ENV
- jenkins task waits for jobs to finish (only one job at a time)

Staging

- running in the same cluster
- separate kc config
- separate ELBs (public and vpn), TCP only
- 2 x nginx ingress (public and vpn)
- TLS via kube-lego
- basic-auth proxy (because SEO)

Logging

- all important information in single nginx line
- rails App passes information via HTTP header
- fluented as DaemonSet on all nodes
- kinesis-Firehose → S3 → SQS → k8s → ES
- kibana

deployment/nginx

- versioned ConfigMap
- custom nginx image
- ngx_headers_more

Benefits

- no more config management (chef, puppet, etc.)
- fully transparent infrastructure
- better resource utilization, bigger instances
- new services deployed in minutes
- easy to scale cluster up and down
- hardly any lock-in to specific hosting infrastructure
- **spend more time on development, less time on ops**

What’s next?

- kops: advanced kubernetes cluster management on AWS
- extract first components from monolith via GRPC

Resources

- https://github.com/jetstack/kube-lego
- https://github.com/kubernetes/ingress/tree/master/controllers/nginx
- https://github.com/kubernetes/kops
- https://prometheus.io/
- http://www.fluentd.org/
- http://www.grpc.io/

PhraseApp

We are always looking for talent!

@tobstarr
tobias@phraseapp.com
phraseapp.com