Mesos at Yelp: Building a production ready PaaS

Rob Johnson
robj@yelp.com/@rob_johnson_
Who Am I:

- Rob Johnson
- Operations Team at Yelp
- Spend most of my time working on PaaSTA
Yelp’s Mission:
Connecting people with great local businesses.
Yelp Stats:
As of Q2 2015

83M
83M
68%
32
PaaSTA
Yelp’s homegrown Platform-as-a-Service
What’s the problem we’re trying to solve here?
- Yelp’s monolith is ~3 million LoC (that’s just the Python).

* 

- Increasing number of developers.

*as of 28/09/2015
- Code deployments become increasingly difficult to coordinate.
- Surface area for impact of a bug greatly increases.
What’s the solution?
SOA
Solves everything, right?
- Statically defined list of hosts to deploy a service on.

- Operations handle deciding which hosts to deploy to.
- Manually configure Nagios for each service.
- Manual deployment system. Lots of rsync wrappers to push code around.
This doesn’t scale well.
PaaSTA
- Built on the shoulders of established tools.
- ‘Glue Code’ that coordinates these tools.
Components
Mesos
Marathon
Chronos

(almost)
My work here is done, right?
Not Quite.
Services != Production
What makes a service production ready?
- easy deployment for developers
- easy deployment for developers
- discovery
- easy deployment for developers
- discovery
- monitoring
- easy deployment for developers
- discovery
- monitoring
- highly available
- easy deployment for developers
- discovery
- monitoring
- highly available
- operational support
- easy deployment for developers
- discovery
- monitoring
- highly available
- operational support
Services at Yelp tend to be:

- http api
- Python
- uWSGI
We want to be stack agnostic; developers shouldn’t be constrained by dependencies on a server.
- PaaSTA only runs Docker containers.

- Developers own the creation of the image.
PaaSTA currently has Java, Golang and Python apps in production.
PaaSTA provides tooling to automate the build and deployment of images via Jenkins.
PaaSTA uses Git as its control plane.
git push
make itest
push to registry
performance check
deploy to dev
(repeat for each dev env)
manual intervention
prod
Once a given image is marked for deployment in production, PaaSTA ‘bounces’ the app, gracefully upgrading the version.
- Reduces operational overhead of deploying service.
- Removes bottleneck of going through operations to deploy.
- easy deployment for developers
- discovery
- monitoring
- highly available
- operational support
- Originally written by Airbnb
- Yelp now has maintainers working on it.
There’s no place like
127.0.0.1
169.254.255.254
Why Smartstack?
- ZK/synapse/nerve dying doesn’t wipe us out.
- HAProxy has its own health checking system we can fall back to.
- HAProxy is a proven load balancer and http proxy.

- We can use Smartstack with non-PaaSTA services.
Zero-downtime HAProxy reloads:

http://bit.ly/1RsctGi
- easy deployment for developers
- discovery
- monitoring
- highly available
- operational support
- API allows us to send event data.
- Flexibility to assign alerts to service authors, rather than forcing it on operations team.
$ cat monitoring.yaml

--

# YAML configuration file for monitoring

```yaml
---
team: search_infra
notification_email: search@yelp.com
page: true
runbook: 'y/rb-myservice'

alert_after: 5m

realert_every: 10m

tip: 'The federator service is in the critical path for search, you should be fixing this'
```

"The federator service is in the critical path for search, you should be fixing this"
./check_marathon_services_replication
./check_hung_setup_marathon_jobs
- easy deployment for developers
- discovery
- monitoring
- highly available
- operational support
Yelp organises machines into latency zones.
$ cat smartstack.yaml
---
main:
  advertise: [superregion]
discover: superregion
proxy_port: 20603
By choosing a more specific latency zone, service owners optimize for RTT over availability.
- By being aware of these latency zones, PaaSTA can make smarter decisions on how to constrain applications.
Without this coupling, Marathon wouldn’t balance apps evenly amongst the latency zones.
- easy deployment for developers
- discovery
- monitoring
- highly available
- operational support
PaaSTA comes with a cli for managing PaaSTA services.
$ paasta metastatus -c norcal-prod
Cluster: norcal-prod
Warning: Dashboards in prod are not directly reachable. See http://y/paasta-troubleshooting for instructions. (search for 'prod dashboards')
User Dashboards (Read Only):
  Mesos:  http://mesos.paasta-norcal-prod.yelp/
  Marathon:  http://marathon.paasta-norcal-prod.yelp/
  Chronos:  http://chronos.paasta-norcal-prod.yelp/
  Synapse:  http://paasta-norcal-prod.yelp:3212/
Admin Dashboards (Read/write, requires secrets):
  Mesos:  http://paasta-norcal-prod.yelp:5050/
  Marathon:  http://paasta-norcal-prod.yelp:5052/
  Chronos:  http://paasta-norcal-prod.yelp:5053/
Mesos Status:
quorum: masters: 5 configured quorum: 3
  framework: marathon count: 1
CPUs: 359.25 / 1052 in use (34.15%)
Memory: 1076.62 / 3063.63GB in use (35.14%)
slaves: active: 61 inactive: 0
tasks: running: 227 staging: 0 starting: 0
Marathon Status:
  marathon apps: 66
  marathon tasks: 227
  marathon deployments: 0
$ paasta info -s example_service
Service Name: example_service
Description: Implements SCF3
External Link (CEP/SCF): https://docs.google.com/a/yelp.com/document/d/1_lBmSTY482-7YwtzbEKlue2WH-rkTYxkBpI-4W47cE4/view#
Monitor By: team paasta
Runbook: Please set a 'runbook' field in your monitoring.yaml. Like "y/rb-mesos". Docs: https://trac.yelpcorp.com/wiki/HowToService/Monitoring/monitoring.yaml
Git Repo: git@git.yelpcorp.com:services/example_service.git
Jenkins Pipeline: https://jenkins.yelpcorp.com/view/services-example_service

Deployed to the following clusters:
- nova-prod (http://example_service.paasta-nova-prod.yelp/)
- norcal-prod (http://example_service.paasta-norcal-prod.yelp/)
- mesosstage (http://example_service.paasta-mesosstage.yelp/)
- norcal-devb (http://example_service.paasta-norcal-devb.yelp/)
- norcal-devc (http://example_service.paasta-norcal-devc.yelp/)
- pmw-stagea (http://example_service.paasta-pmw-stagea.yelp/)
- norcal-stageb (http://example_service.paasta-norcal-stageb.yelp/)

Smartstack endpoint(s):

Dashboard(s):
- https://uchiwa.yelpcorp.com#/events?q=example_service (Sensu Alerts)
$ paasta status -s example_service -c norcal-prod
Pipeline: https://jenkins.yelpcorp.com/view/services-example_service

cluster: norcal-prod
  instance: canary
    Git sha: 9612bcc9
    State: Running - Desired state: Started
    Marathon: Healthy - up with (1/1) instances. Status: Running.
    Mesos: Healthy - (1/1) tasks in the TASK_RUNNING state.
    Smartstack: N/A - canary is announced in the main namespace.

instance: main
    Git sha: 80cfd3a0
    State: Running - Desired state: Started
    Marathon: Healthy - up with (3/3) instances. Status: Running.
    Mesos: Healthy - (3/3) tasks in the TASK_RUNNING state.
    Smartstack:
      norcal-prod - Healthy - in haproxy with (16/4) total backends UP in this namespace.

$
- easy deployment for developers
- discovery
- monitoring
- highly available
- operational support
Questions?