Advanced Mesos and Marathon

Michał Łowicki

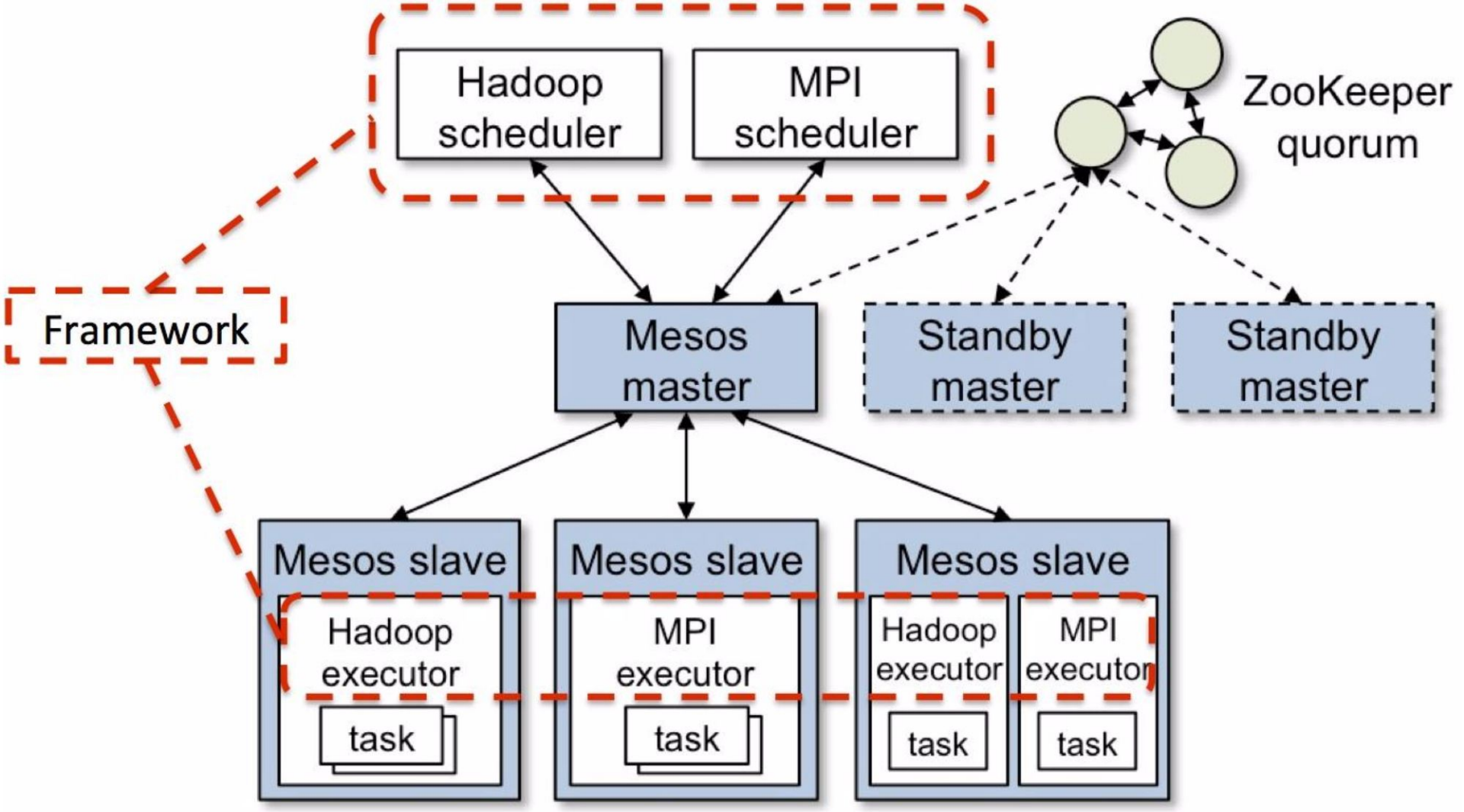
About me

- Web Services at Opera
- medium.com/@mlowicki
- mlowicki@opera.com

Agenda

- Monitoring & incidents management
- Load balancing
- mgr

Introduction to Mesos and Marathon

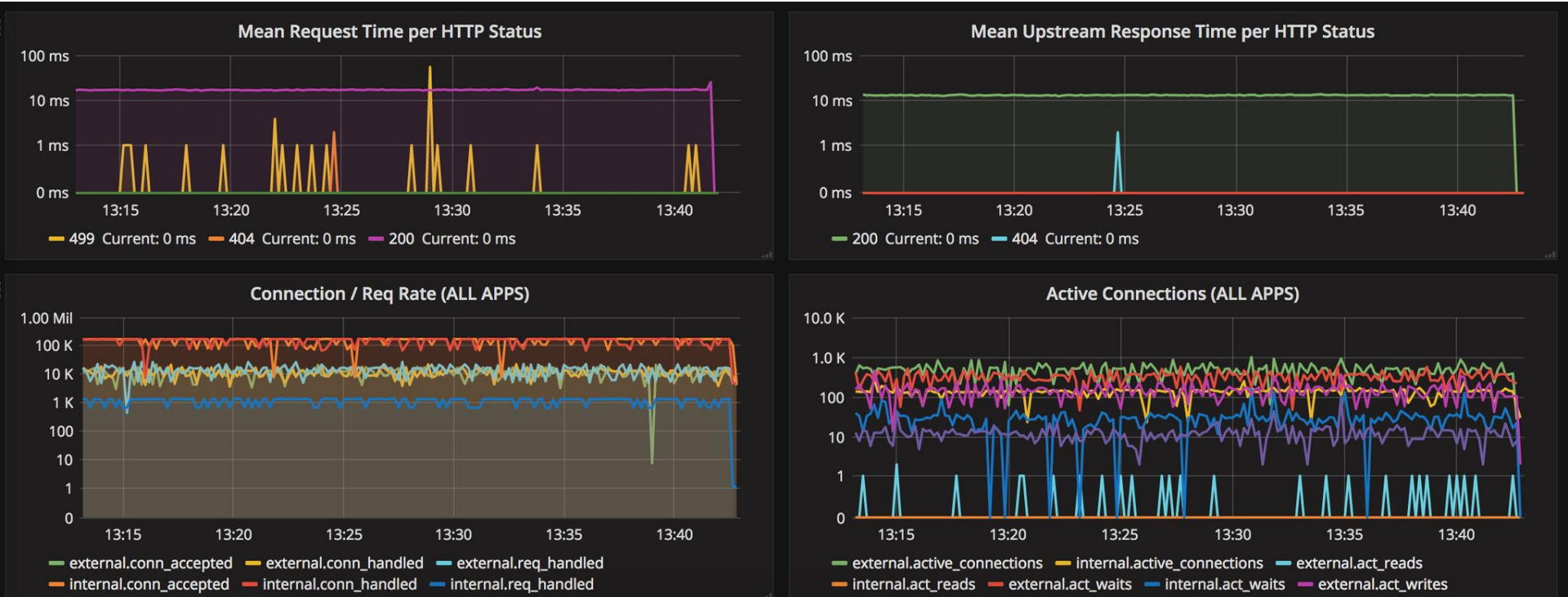

Hadoop scheduler
MPI scheduler
ZooKeeper quorum
Framework
Mesos master
Standby master
Standby master
Mesos slave
Mesos slave
Mesos slave
Hadoop executor
task
MPI executor
task
Hadoop executor
task
MPI executor
task

Marathon

Container orchestration platform for Mesos

https://mesosphere.github.io/marathon/

Monitoring & incidents management



pagerduty

	Service	Incidents	Last Incident	Integrations	Notification Urgency	Teams	Escalation Policy	
✔	**Autofill** Forms auto-filler. Production deploy	**0** triggered **0** acknowledged	Jan 25, 2017 10:19 AM	Pingdom Email Nagios MongoDB Cloud Manager	High	Web Services Infrastructure	Web Services Infrastructure - Normal	⚙
✔	**Autofill test** Forms auto-filler. Test deploy	**0** triggered **0** acknowledged	Jan 18, 2017 10:11 PM	MongoDB Cloud Manager	Low	Web Services Infrastructure	Web Services Infrastructure - Normal	⚙
✔	**Favicostore** Favicons service. Production deploy	**0** triggered **0** acknowledged	Jan 28, 2017 8:31 AM	Pingdom Email MMS Nagios	Low	Web Services Infrastructure	Web Services Infrastructure - Normal	⚙
✔	**Mesos cluster** Production deploy	**0** triggered **0** acknowledged	Jan 26, 2017 2:59 PM	Nagios REST	High	Web Services Infrastructure	Web Services Infrastructure - Normal	⚙
✔	**Mesos cluster (low priority)** Production cluster - low priority checks.	**0** triggered **0** acknowledged	Jan 27, 2017 3:52 PM	Nagios REST Grafana	Low	Web Services Infrastructure	Web Services Infrastructure - Normal	⚙

Web Services Infrastructure - Normal

Immediately after an incident is triggered

1 Notify:

Web Services Infrastructure
ON CALL NOW
Piotr Śliwka (redacted)

escalates after **30 minutes**

2 Notify:

Michał Łowicki (redacted)

escalates after **1 hour**

Repeats **3** times if no one acknowledges incidents

Used by **10** services

- Autofill
- Autofill test
- Favicostore
- Mesos cluster
- Mesos cluster (low priority)
- Redir
- Services commons
- Sitecheck
- sourcecode.opera.com
- SpeedDials

Used by **1** team

- Web Services Infrastructure

Flow

Service alert → Escalation policy → Schedule or specific operators

Feeding PagerDuty

- Pingdom
- Grafana
- Nagios
- Custom integrations (through official APIs)

pingdom

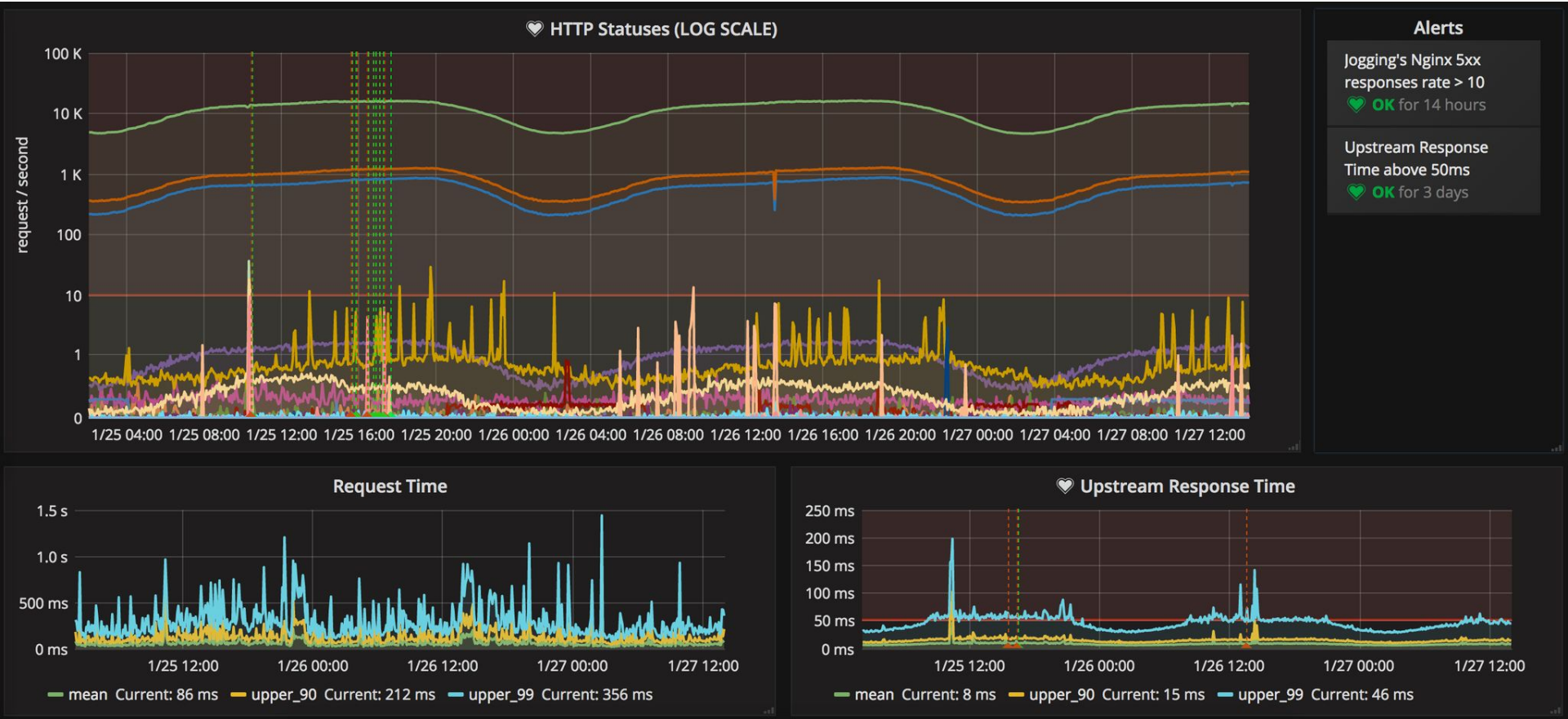
Pause Delete By status Last 24 hours Filter Add new Monitor my website

Tags... Showing 3 of 20 checks Clear filters sync

up

	SITE NAME	TAGS	TYPE	UPTIME	UP SINCE	RESPO...	
	nl.sync.opera.com nl.sync.opera.com	1	HTTP	100.00%	4 d	116 ms 0 min	
	sync.opera.com sync.opera.com	1	HTTPS	100.00%	4 d	273 ms 0 min	
	us.sync.opera.com us.sync.opera.com	1	HTTP	100.00%	2 mo	165 ms 0 min	

Alert notifications in Grafana



Nagios + Marathon

- Designated Docker containers with all prerequisites to run service tests
- Nagios ssh there and run command(s) using *cli tools

```
#!/usr/bin/env bash

afquery $1 --server $2 --protocol protobuf 1> /dev/null

if [[ $? -ne 0 ]]; then
    exit 2
fi

echo "OK"
```

- Nagios uses *marathoncli* to find where container with checks resides

MARATHON

Applications | Deployments

Search all applications

STATUS
- Running 8
- Deploying
- Suspended
- Delayed
- Waiting

HEALTH
- Healthy 6
- Unhealthy
- Unknown 2

LABEL
Select

RESOURCES
- Volumes 1

Applications > services > autofill > production

Create Group | Create Application

Name	CPU	Memory	Status	Running Instances	Health
app	96.0	12 GiB		12 of 12	
checks NIXY_REALM:internal	1.0	128 MiB	Running	1 of 1	
fluentd HAPROXY_GROUP:internal NIXY_REALM:internal	2.0	256 MiB	Running	2 of 2	
memcached HAPROXY_GROUP:internal NIXY_REALM:internal	2.0	256 MiB	Running	2 of 2	
mms HAPROXY_GROUP:internal NIXY_REALM:internal	1.0	192 MiB	Running	2 of 2	
mongo HAPROXY_GROUP:internal NIXY_REALM:internal	12.0	48 GiB	Running	3 of 3	
statsd HAPROXY_GROUP:internal NIXY_REALM:internal	4.0	512 MiB	Running	4 of 4	
telegraf HAPROXY_GROUP:internal NIXY_REALM:internal	1.0	128 MiB	Running	1 of 1	

Looking for Nagios replacement

- checks triggered autonomously on the servers
- results sent to centralized infra
- something actively maintained
- extensible (custom checks)
- ideally written in Python (see previous point)

SENTRY

Stop hoping your users will report errors

https://sentry.io/welcome/

Error	Unknown parent /api/sync/command/ a few seconds ago — 9 months old | sync.api.handlers.commit		14m	12k
Error	Specifics changed /api/sync/command/ 4 minutes ago — a year old | sync.api.handlers.commit		295k	41k
Error	RequestDataTooBig /api/sync/command/ Request body exceeded settings.DATA_UPLOAD_MAX_MEMORY_SIZE. 6 minutes ago — 2 months old | sync.api.middleware		6.4k	110
Error	ValueError /api/sync/command/ 'hS\x19\xd0Q\x0eZ\xdbX\xcc\xd5\xd0\\\x0b\xdd\x13\x92\x19\x9a\x94\xf3\x83\x14\x944\xd3' h... 7 minutes ago — a month old | sync.api.middleware		920	29
Error	preference name or value too long /api/sync/command/ 13 minutes ago — 2 years old | sync.api.validators		11k	465
Error	old entity missing /api/sync/command/ 40 minutes ago — 2 years old | sync.api.handlers.commit		1.1m	194k

Specifics changed

/api/sync/command/ | sync.api.handlers.commit

ASSIGNED | EVENTS 295k | USERS 41k

Resolve | Ignore | Create JIRA Issue

Details | Comments 0 | User Feedback 0 | Tags | Related Events

Share this event

Event ea5242493d8a486e93403f67f21490da

January 29 2017 14:11:25 CET | JSON (43.5 KB)

Older | Newer

This event was reported with an old version of the python SDK. Learn More

TAGS

browser	Other
device	Other
level	error
logger	sync.api.handlers.commit
os	Other
release	3fb4278
server_name	front4.sync.ams.osa
transaction	/api/sync/command/
url	https://sync.opera.com/api/sync/command/
user	id:375904982

MESSAGE

(Default Environment)

LAST 24 HOURS

LAST 30 DAYS

FIRST SEEN

When: 4 months ago

September 22 2016 11:36:06 CEST

Release: 189da0b

LAST SEEN

Jogging

The loadbalancer for Marathon

Under the hood

- https://github.com/martensson/nixy
- Nginx (+watcher.sh)
- Diamond (for metrics)

Nixy uses persistent connection to Marathon to be notified right away about new events. It produces nginx.conf and if changed Nginx will be reloaded.

Run only one process per container

In almost all cases, you should only run a single process in a single container. Decoupling applications into multiple containers makes it much easier to scale horizontally and reuse containers. If that service depends on another service, make use of container linking.

Best practices for writing Dockerfiles

watcher.sh

```
#!/bin/sh

export NIXY_DRAFT=/etc/nginx/nginx.conf.nixy_draft

while true
do
    touch ${NIXY_DRAFT}
    inotifyd - ${NIXY_DRAFT}:cDM > /dev/null
    /var/reloader.sh &
done
```

Labels in Marathon

- NIXY_PROTOCOL=tcp
- NIXY_REALM=internal

Why reinventing the wheel?

Projects like marathon-lb, træfɪk didn’t worked because of their instability or lack of required features like load balancing raw TCP or UDP.

dumb-init

- A minimal init system for Linux containers
- https://github.com/Yelp/dumb-init
- It solves two issues: handling signals properly & reaping zombie processes

https://blog.phusion.nl/2015/01/20/docker-and-the-pid-1-zombie-reaping-problem/

mgr

- Simplifies dockerizing applications
- Integrations with Mesos / Marathon / GitLab
- Development environment backed on Docker Compose
- Supports legacy deployments (not based on Docker or Marathon)

Integration

```
> tree -L 1
.
├── docker
├── marathon
├── mgrfile.py
├── requirements
└── src
```

```
> mgr start
```

```
> mgr tests
```

Logs

```
> mgr start -s statsd

> mgr -e test marathon logs statsd

> mgr -e test marathon logs statsd --limit 0
```

Uses MultiTail under the hood

Shell

```
> mgr bash -s uwsgi

> mgr -e test marathon shell
```

Wrappers around API calls

```
> mgr -e test marathon scale +1 statsd
...
[INFO] Scaling /services/netinstaller/test/statsd in ams dc
[INFO] /services/netinstaller/test/statsd successfully scaled in ams dc
```

Questions?

Thank you.