Monitoring Mesos, Docker, Containers with Zabbix

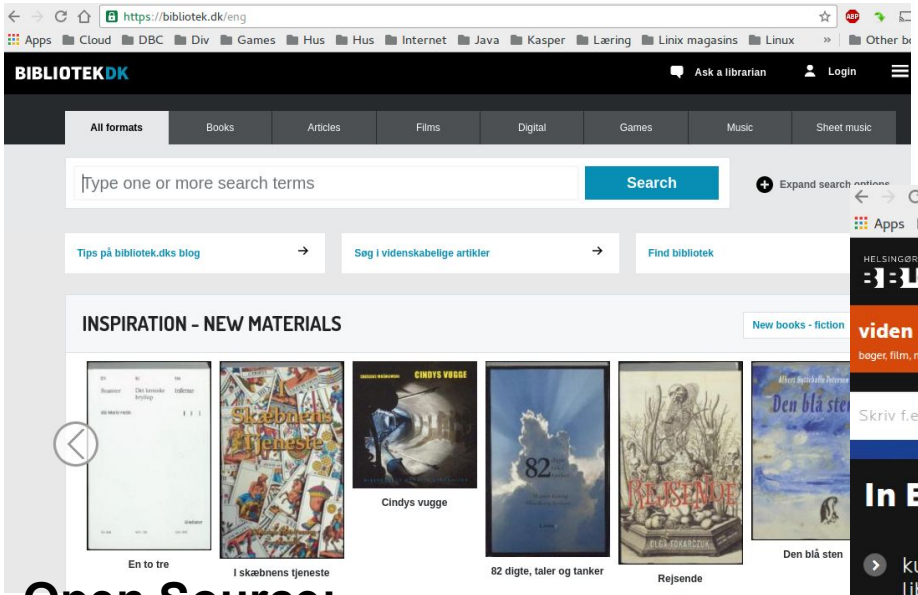
By Erik Skytthe, DBC A/S, Denmark

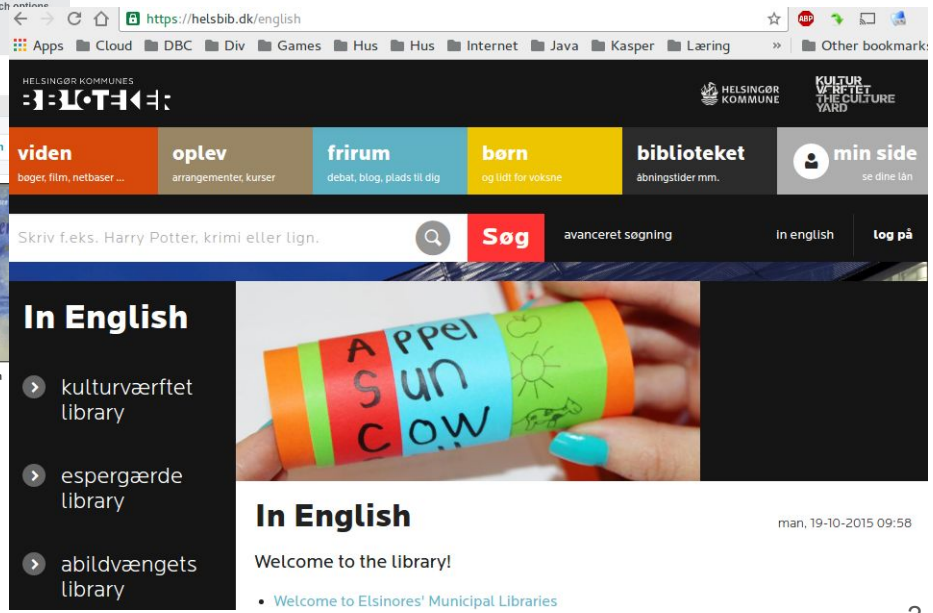
https://dk.linkedin.com/in/erik-skytthe-444a7720
(Zabbix IRC/Forum: eskytthe)
Email: ers ADD dbc.dk

Central material search:
(books, films, articles, research, facts …)
Central databases, web services etc.

DBC

Central libraries
Research libraries
Education libraries
Local libraries:
...





Open Source:

https://opensource.dbc.dk/
https://github.com/DBCDK/

- **Templates etc.here efter conference**

DBC

DBC

- > 600 devices (Linux, Windows, SAN, Network, Xen, VMware …)
- Software Development
- Many in-house developed applications (some very old)
- Have used Zabbix in ~ 6 years.
- Version 2.4 ~ will update to 3.X in short time

Erik Skytthe

- ~ DevOps … but most Ops
- Continuous Integration
- Continuous Deployment
- Tools and platforms to support this

Who are running?

Some sort of container cluster environment:

- **Mesos** (Apache)
- **Swarm** (Docker)
- **Kubernetes** (Google)
- **Fleet** (CoreOS)
- Or **Other** ...

- **Swarm** ?
- **Kubernetes**?
- **Mesos**?

Who are using Mesos?

It is all around you ... http://mesos.apache.org/documentation/latest/powered-by-mesos

- Twitter
- Apple - Siri (voice recognition)
- Netflix
- PayPal
- Cisco
- Microsoft Azure
- eBay, Airbnb, Uber, CERN ……

and

- DBC :-) … **small** scale **Cons**: not long prod experience

What is Mesos?

A distributed system kernel

A cluster of resources
Offer resources from

- Cpu, memory, storage, network …

Resources consumed by Frameworks

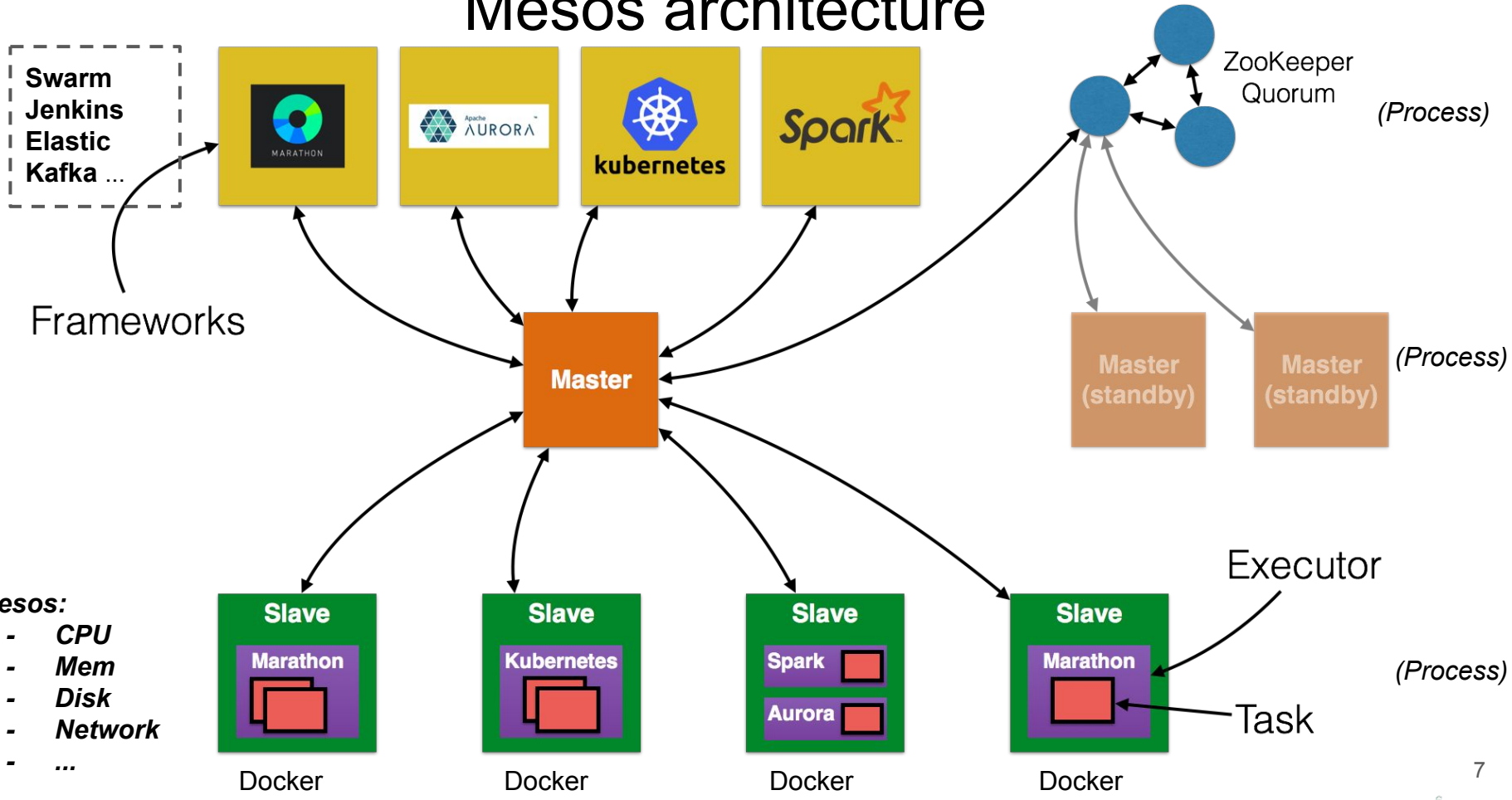
Runs on bare metal, virtual hosts and cloud

A low level distributed system

Apache Mesos abstracts CPU, memory, storage, and other compute resources away from machines … into a shared pool.

http://blog.couchbase.com/2016/may/docker-apache-mesos-marathon

Mesos architecture



Mesos:

- ***CPU***
- ***Mem***
- ***Disk***
- ***Network***
- ***...***

Frameworks - **“Infrastructure as code”**

Orchestration of containers ~ applications / tasks

- Start, Stop, Scaling, Destroy ...
- Health checks
- Grouping
- Dependencies (Mysql container -> Zabbix container)
- Load balance / Proxy

Examples:

Marathon: Long running tasks

Chronos: “Cron” jobs

Jenkins: Up and down scaling of builds

Frameworks - Marathon

```
{
  "id": "/stagning/scrum-team/webserver",
  "cpus": 0.1,
  "mem": 64.0,
  "instances": 1,
  "container": {
        "type": "DOCKER",
        "docker": {
        "image": "docker-repo/nginx",
        "forcePullImage": true,
        "network": "BRIDGE",
          "portMappings": [{
          "containerPort": 80, "hostPort": 0 }]
        }
  },
```

```
 "healthChecks": [{
        "path": "/",
        "gracePeriodSeconds": 5,
        "intervalSeconds": 5,
        "timeoutSeconds": 5,
        "maxConsecutiveFailures": 3
 }]
}
```

```
curl -X POST -H "Content-type: application/json"
http://mesosmasterX:8080/v2/apps -d @webserver.json
```

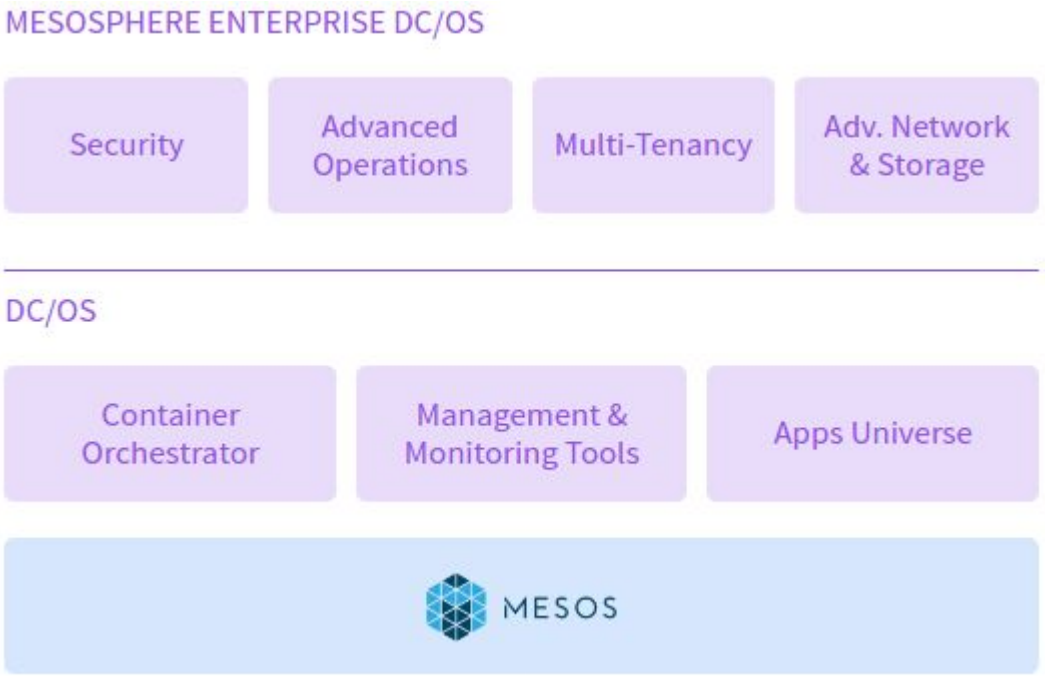
Full stack Mesos - Mesosphere DC/OS

https://dcos.io
(Data Center OS)

https://mesosphere.com/

Enterprise:

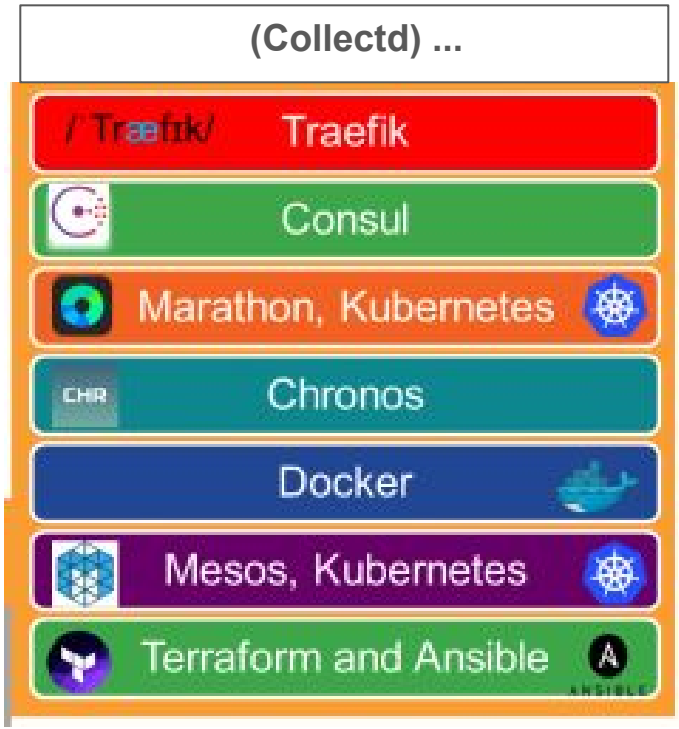
Open Source:

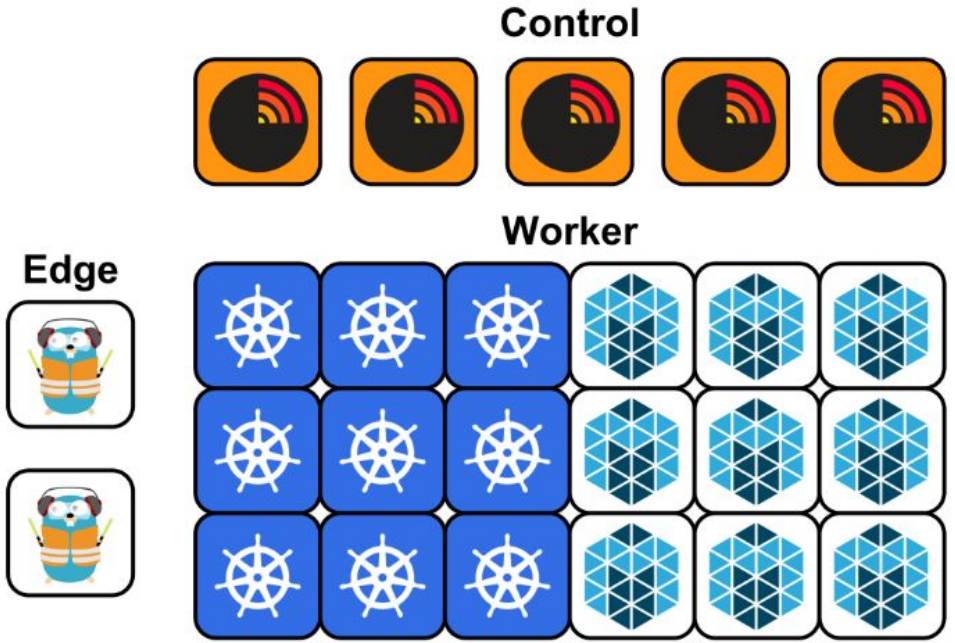


Full stack Mesos - Mantl

http://mantl.io - Cloud infrastructure Cisco

- We use it as a reference model





MesosCon Amsterdam 2016

http://events.linuxfoundation.org/events/mesoscon-europe

- Fault tolerence in mesos: http://sched.co/7n7x
- Mesos 1.0: http://sched.co/7n7s
- ...

MesosCon America 2016

Video sessions:
https://www.linux.com/news/mesoscon-north-america-2016-video-sessions

DevOps / container config and deploy:

- “Lessons Learned from Running Heterogeneous Workload on Apache Mesos”
- “All Marathons Need a Runner. Introducing Pheidippides”
- …

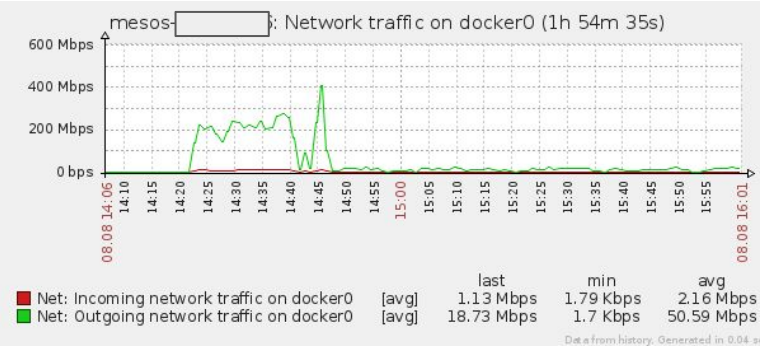
In deep container troubleshooting: (sysdig: cli / runtime tool)

- “Monitoring Microservices: Docker, Mesos and DCOS Visibility at Scale”
- https://sysdig.com/blog/monitoring-mesos/
-

Hardware - Server monitoring

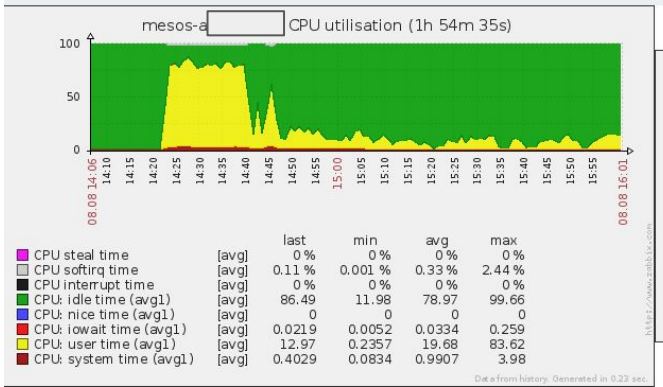
Standard zabbix agent - **plain vanilla**

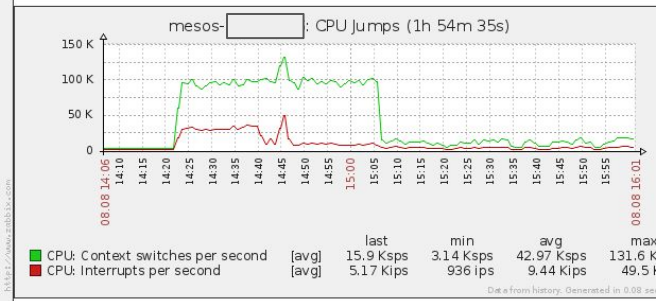
- “Standard” Linux templates
- Autodiscovery of Disk and Networks
- Syslog to ELK



Trigger / alert note:

- Filesystem space and inodes! - because of Docker (images etc.)





Getting Data Metrics

curl -s 'http://mesosmasterx:5050/metrics/snapshot' | jq '.'


```
{
 ...
  "master/tasks_starting": 0,
  "master/tasks_staging": 0,
  "master/tasks_running": 38,
  "master/tasks_lost": 0,
  "master/tasks_killing": 0,
  "master/tasks_killed": 770,
  "master/tasks_finished": 0,
  "master/tasks_failed": 129,
  "master/tasks_error": 0,
  "master/task_killed/source_slave/reason_executor_unregistered": 1,
  "master/slaves_inactive": 0,
  "master/slaves_disconnected": 0,
  "master/slaves_connected": 8,
  "master/slaves_active": 8,
....
}
```

Collectd

We did choose collectd ... so far, because:

- **Mantl** (ready made ansible roles etc.)
- Recommendations in **mesos mail list**
- We already running **graphite**
  - (Nice analyse possibilities)

Cons:

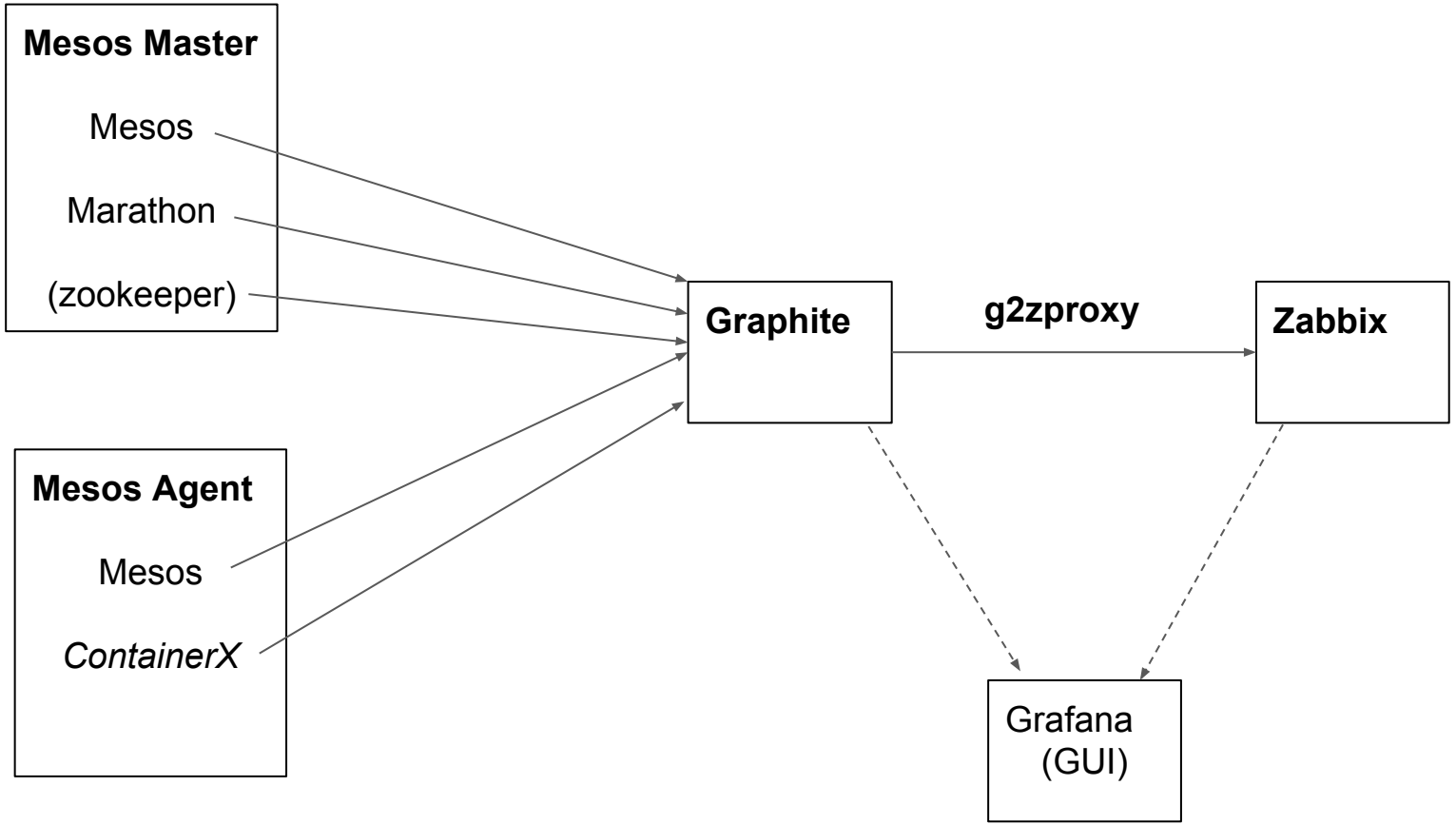
- No zabbix write plugin so far
- Easy to make e.g. python program to get data from urls.

Ansible role from Mantl project - to install:

~/work/mantl/roles/collectd (*git clone* …)

Remark: Debian require libpython2.7

Collectd data flow



Collectd - write to graphite plugin

```
/etc/collectd/
├── collectd.conf
├── collectd.conf.d
│   ├── carbon.conf
│   ├── filters.conf
│   ├── mesos-master.conf
│   └── thresholds.conf
└── collection.conf
```

collectd.conf:

```
# Sampling interval sec.
Interval 20
<Include "/etc/collectd/collectd.conf.d">
        Filter "*.conf"
</Include>
```

Carbon.conf

 # For version 5.1 and later using the Write Graphite plugin

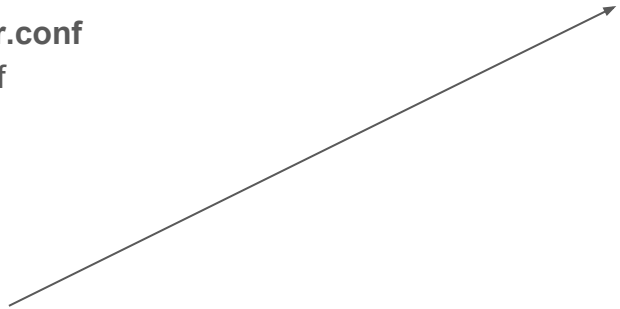
```
FQDNLookup false
Timeout 2
ReadThreads 5
LoadPlugin write_graphite
<Plugin "write_graphite">
   <Carbon>
      Host "graphitehost name"
      Port "{{ GRAPHITE_PORT | default("2003") }}"
      Protocol "tcp"
      Prefix "{{ GRAPHITE_PREFIX | default("collectd.") }}"
      EscapeCharacter "."
      StoreRates true
      AlwaysAppendDS false
      SeparateInstances true
   </Carbon>
</Plugin>
```

Collectd mesos-master.conf

```
/etc/collectd/
├── collectd.conf
├── collectd.conf.d
│   ├── carbon.conf
│   ├── filters.conf
│   ├── mesos-master.conf
│   └── thresholds.conf
└── collection.conf
```

mesos-master.conf:

```
LoadPlugin python
<Plugin "python">
   ModulePath "/usr/share/collectd/plugins/"
   Import "mesos-master"
   <Module "mesos-master">
      Host "localhost"
      Port 5050
      Verbose false
      Version "0.22.1"
   </Module>
</Plugin>
```



/usr/share/collectd/plugins/mesos-master.py

Colltecd mesos-agent.conf

```
/etc/collectd/
├── collectd.conf
├── collectd.conf.d
│   ├── carbon.conf
│   ├── filters.conf
│   ├── mesos-agent.conf
│   └── thresholds.conf
└── collection.conf
```

/usr/share/collectd/plugins/mesos-agent.py

*Note: May be named "mesos-**slave**" also*

mesos-agent.conf:

```
LoadPlugin python

<Plugin "python">
   ModulePath "/usr/share/collectd/plugins/"
   Import "mesos-agent"
   <Module "mesos-agent">
      Host "localhost"
      Port 5051
      Verbose false
      Version "0.22.1"
   </Module>
</Plugin>
```

Same config model for marathon, zookeeper etc.

Graphite - Zabbix integration

Graphite to Zabbix proxy: Blacked graphite-to-zabbix
https://github.com/blacked/graphite-to-zabbix

Crontab:

*/1 * * * * g2zproxy -z https://**zabbixhost** -zu {zabbixUser} -zp {zabbixPass} -g http://**graphitehost** ...

Graphite key: ***mesos-masterx.mesos-master.gauge.master_elected***

Zabbix Host: ***mesos-masterx***

Zabbix key: ***graphite[mesos-master.gauge.master_elected]***

Pros: Possible to use Graphite functions in zabbix requests (zabbix key)

Mesos Master Monitoring

Mesos metrics:

http://mesos.apache.org/documentation/latest/monitoring/

Monitoring and trigger recommendations:

https://docs.mesosphere.com/1.7/administration/monitoring/performance-monitoring/

Cons:

- Data only from elected master
- Bug? 2.8 lost tasks? (counter)

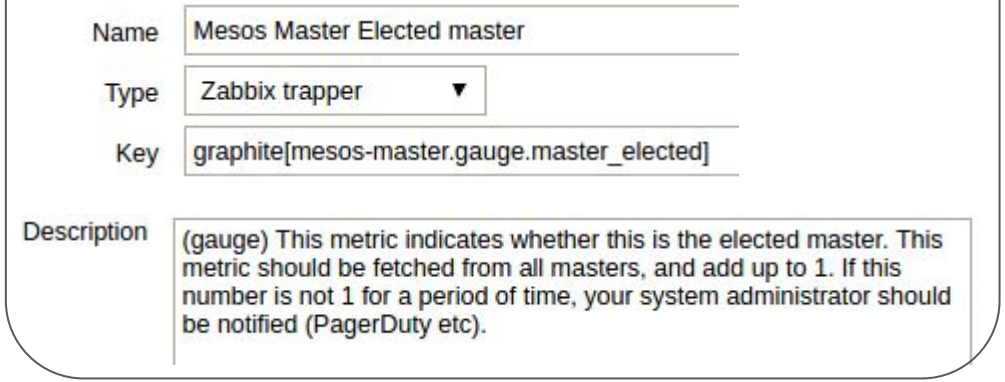
etcd:

https://github.com/shamil/zabbix-etcd

zookeeper:

https://github.com/zhangqin/zookeeper-zabbix-template

Recommendations included in item description:



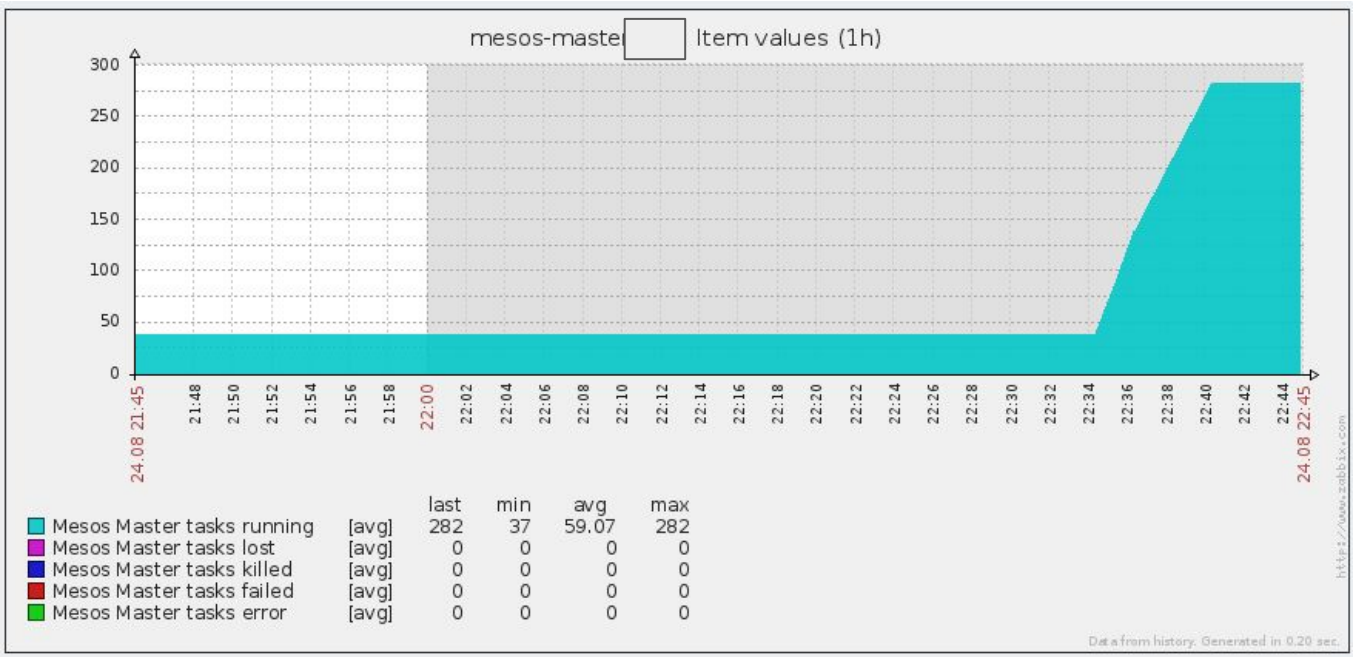
Mesos Master template

Name	Triggers	Key
Mesos Master messages decline offers		graphite[mesos-master.counter.master_messages_decline_offers]
Mesos Master agent removals		graphite[mesos-master.counter.master_slave_removals]
Mesos Master agent re-registrations	Triggers (1)	graphite[mesos-master.counter.master_slave_reregistrations]
Mesos Master tasks error		graphite[mesos-master.counter.master_tasks_error]
Mesos Master tasks failed		graphite[mesos-master.counter.master_tasks_failed]
Mesos Master tasks killed		graphite[mesos-master.counter.master_tasks_killed]
Mesos Master tasks lost		graphite[mesos-master.counter.master_tasks_lost]
Mesos Master cpus total		graphite[mesos-master.gauge.master_cpus_total]
Mesos Master cpus used		graphite[mesos-master.gauge.master_cpus_used]
Mesos Master Elected master		graphite[mesos-master.gauge.master_elected]
Mesos Master frameworks active	Triggers (1)	graphite[mesos-master.gauge.master_frameworks_active]
Mesos Master frameworks connected		graphite[mesos-master.gauge.master_frameworks_connected]
Mesos Master mem total		graphite[mesos-master.gauge.master_mem_total]
Mesos Master mem used		graphite[mesos-master.gauge.master_mem_used]
Mesos Master agents active		graphite[mesos-master.gauge.master_slaves_active]
Mesos Master agents connected	Triggers (1)	graphite[mesos-master.gauge.master_slaves_connected]
Mesos Master agents disconnected		graphite[mesos-master.gauge.master_slaves_disconnected]
Mesos Master tasks running		graphite[mesos-master.gauge.master_tasks_running]
Mesos Master uptime	Triggers (1)	graphite[mesos-master.gauge.master_uptime_secs]

Mesos Master (19 Items)		
Mesos Master agent re-registrations	2016-08-24 ...	0
Mesos Master agent removals	2016-08-24 ...	0
Mesos Master agents active	2016-08-24 ...	8
Mesos Master agents connected	2016-08-24 ...	8
Mesos Master agents disconnected	2016-08-24 ...	0
Mesos Master cpus total	2016-08-24 ...	112
Mesos Master cpus used	2016-08-24 ...	16.2
Mesos Master Elected master	2016-08-24 ...	1
Mesos Master frameworks active	2016-08-24 ...	1
Mesos Master frameworks connected	2016-08-24 ...	1
Mesos Master mem total	2016-08-24 ...	878152
Mesos Master mem used	2016-08-24 ...	256128
Mesos Master messages decline offers	2016-08-24 ...	0
Mesos Master tasks error	2016-08-24 ...	0
Mesos Master tasks failed	2016-08-24 ...	0
Mesos Master tasks killed	2016-08-24 ...	0
Mesos Master tasks lost	2016-08-24 ...	0
Mesos Master tasks running	2016-08-24 ...	282
Mesos Master uptime	2016-08-24 ...	23789.27
ZooKeeper (18 Items)		

Mesos Master Monitoring

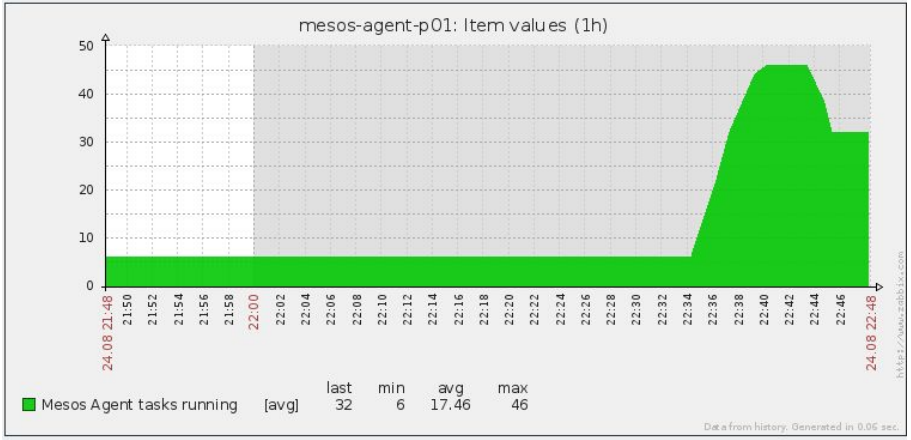
-



Mesos Agent (Slave) Monitoring

Name	Triggers	Key
Mesos Agent registered	Triggers (1)	graphite[mesos-slave.gauge.slave_registered]
Mesos Agent tasks running		graphite[mesos-slave.gauge.slave_tasks_running]
Mesos Agent uptime	Triggers (1)	graphite[mesos-slave.gauge.slave_uptime_secs]

Mesos Agent (3 Items)		
Mesos Agent registered	2016-08-24...	1
Mesos Agent tasks running	2016-08-24...	32
Mesos Agent uptime	2016-08-24...	25088.68



- Plus process monitoring e.g. mesos, docker, etcd

Marathon Framework Monitoring

http://mesosmasterx:5050/metrics

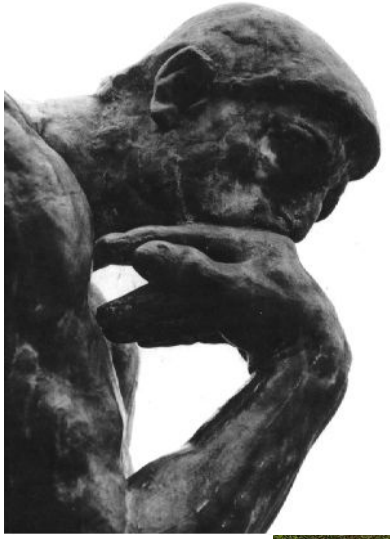
Colletctd marathon plugin: Not working with authentication and SSL!
https://github.com/klynch/collectd-marathon

Marathon metrics flags
--reporter_graphite : t*cp://**graphitehost**:2003?prefix=**marathon-test**&interval=10*
Report metrics to Graphite as defined by the URL.

Cons:

- Bug in tasks metric
- Some metric names not updated in documentation

https://docs.mesosphere.com/1.7/administration/monitoring/performance-monitoring



Container Monitoring

Construction work ahead !

Note: Huge topic - Need separate presentation

Container Monitoring

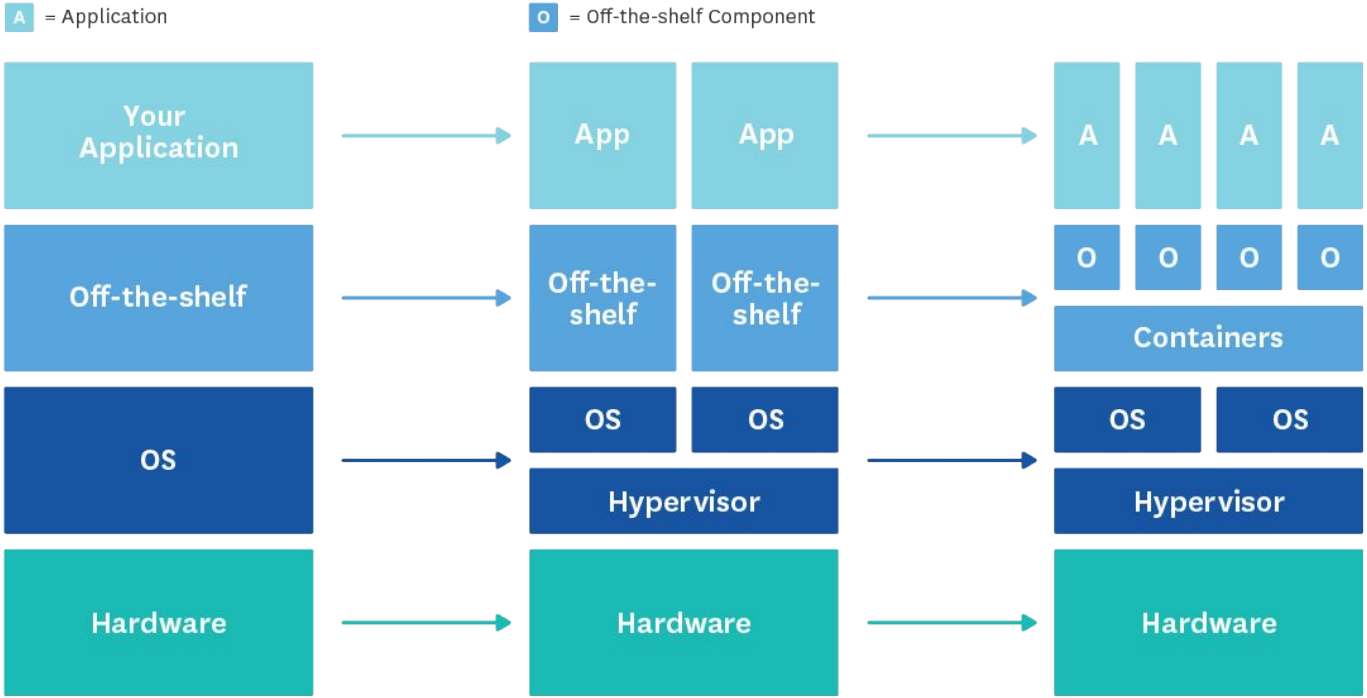
How to represent containers ~ applications / tasks in Zabbix ?

How to show them …? In a structural way ? …

How to find them?

How to aggregate and analyse them?

Container Monitoring



Evolution of the standard application stack. *(Off-the-shelf = eg. Java EE)*

Source: Datadog - https://www.datadoghq.com/blog/the-docker-monitoring-problem/

Container Monitoring

“we need a new approach, one that does **not treat everything as a host.**”

- *Containers can be on any agent / node in your cluster*
- *Containers can be in 1 ~ many instances*
- *Containers can start and get destroyed in ms. / sec.*

“**Treat containers as hosts** that come and go every few minutes.

In this case your life is miserable because the monitoring system always thinks half of your infrastructure is on fire”

“**Monitor all layers of your stack together**, so that you can see what is happening everywhere, at the same time, with no gaps”

“**Tag your containers** so that you can monitor them as queryable sets rather than as individuals”

Container Monitoring

“**Treat containers as hosts** that come and go every few minutes”.

“In this case your life is miserable because the monitoring system always thinks half of your infrastructure is on fire”

Not necessarily true

- Long running containers
- Depend on triggers (if any)
- Depend on your use

Our solution: Treat containers as hosts in zabbix

Container Monitoring

Test of two mesos collectd solutions:

https://github.com/bobrik/collectd-mesos-tasks

https://github.com/rayrod2030/collectd-mesos

Cons:

- Open Mesos issue: incorrect CPU metrics
- Can not get them to work (out of the box) in newer mesos versions

Container Monitoring

Solution 1: Monitoringartist (Jan): zabbix-docker-monitoring
https://github.com/monitoringartist/zabbix-docker-monitoring

Book: "Monitoring Docker"
By Russ McKendrick
Chapter 4 - Zabbix install etc.

Container name
CPU system time
CPU user time
Used cache memory
Used RSS memory
Used swap

Cons:

- Limited version: “provides only docker metrics, TLS features and Zabbix agent server IP check are disabled”

Note: See also monitoring analytics and other:
https://hub.docker.com/r/monitoringartist/monitoring-analytics

Container Monitoring

Solution 2: Bobrik: Collectd-docker (run as docker container)

https://github.com/bobrik/collectd-docker

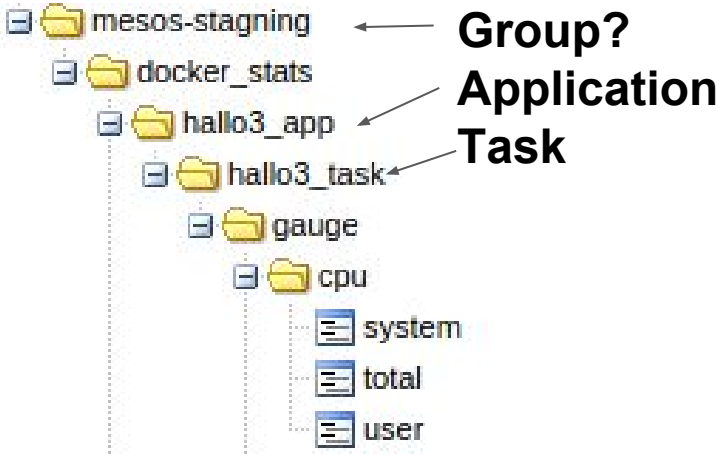
Cons:

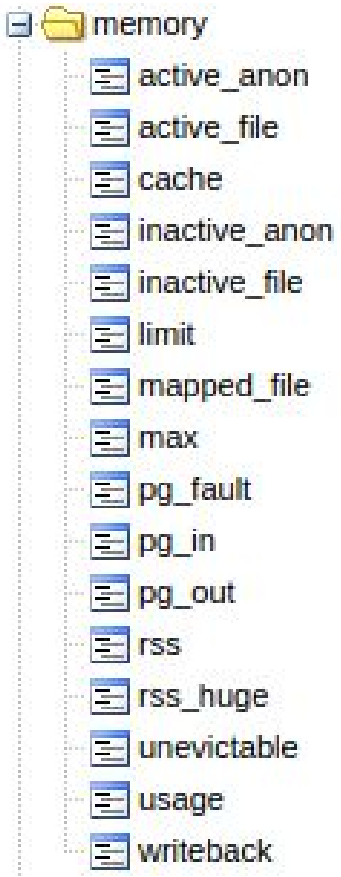
- Only docker
- Need to modify graphite to zabbix proxy tool (g2zproxy)

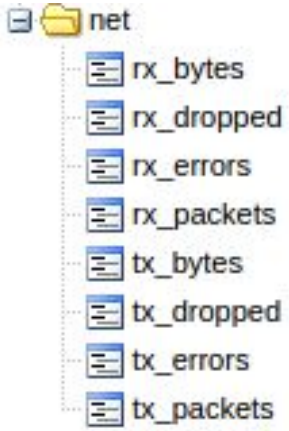
Pros:

- Possible to make a “**zabbix container discover**” solution
- Reliable cpu metrics
- Use of application and task tagging
- Only containers with specific labels will be monitored
- Nice grahana Dashboard

Container Monitoring

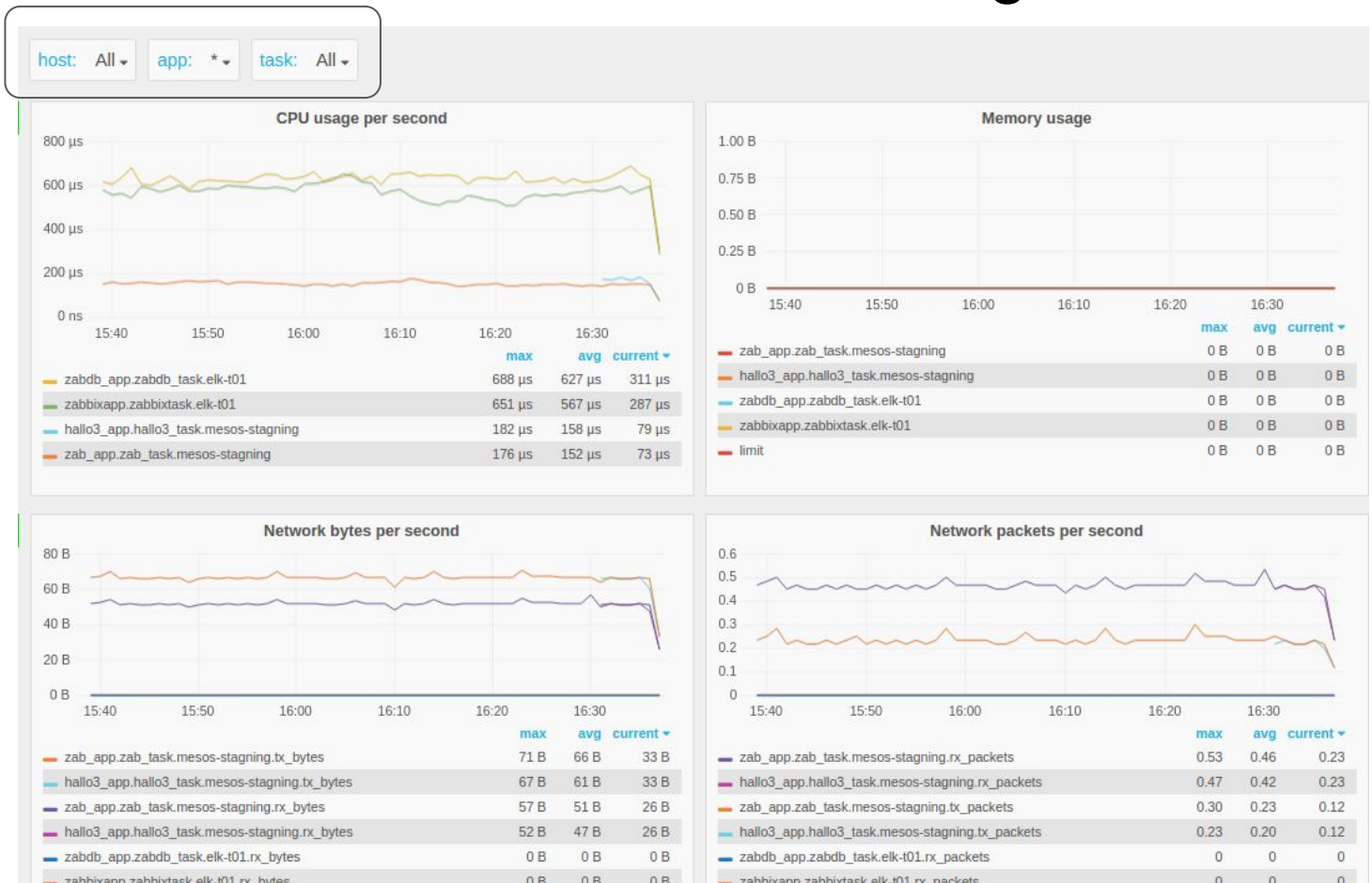






Host, App, Task

Container Monitoring



Container Monitoring - Grouping

Marathon: Name group hiraki e.g:

/ Environment / team / application / [sub application] / ...

/prod/dataio/dbc-glassfish-harvester

/stagning/dataio/dbc-glassfish-harvester

Task = Running container / instance of application:

dbc-glassfish-harvester.760fedd6-684b-11e6-bfc6-0242c91e8407

Application / task relationship:

- One or more different tasks per application
- Many instances of one task
- **Depend how you will group them ...**

Container Monitoring - tagging / labels

Bobrik: Collectd-docker: Only containers with specific **labels** will be monitored

Application / Task: Logical grouping of containers
- You own choice of grouping!

```
docker run \
    -d \
    --name zabbix-db \
    --env="MARIADB_USER=zabbix" \
    --env="MARIADB_PASS=my_password" \
    --label collectd_docker_app="zabbix" \
    --label collectd_docker_task="zabbixdb" \
    monitoringartist/zabbix-db-mariadb
```

Application (= zabbixhost)

Task

Container Monitoring

How to represent containers ~ applications / tasks in Zabbix ?

As hosts

How to show them …? In a structural way ? …

As hosts. But could be better - use of tags etc.

How to find them?

As host search. Needed: Use of tags / labels

How to aggregate and analyse them?

Needed: ~ top10 etc. tools in Zabbix

(Could be nice: image usage, container usage etc. …)

Old service - host discussion?
Container autodiscovery?
Ref: sysdig cloud

Log Monitoring

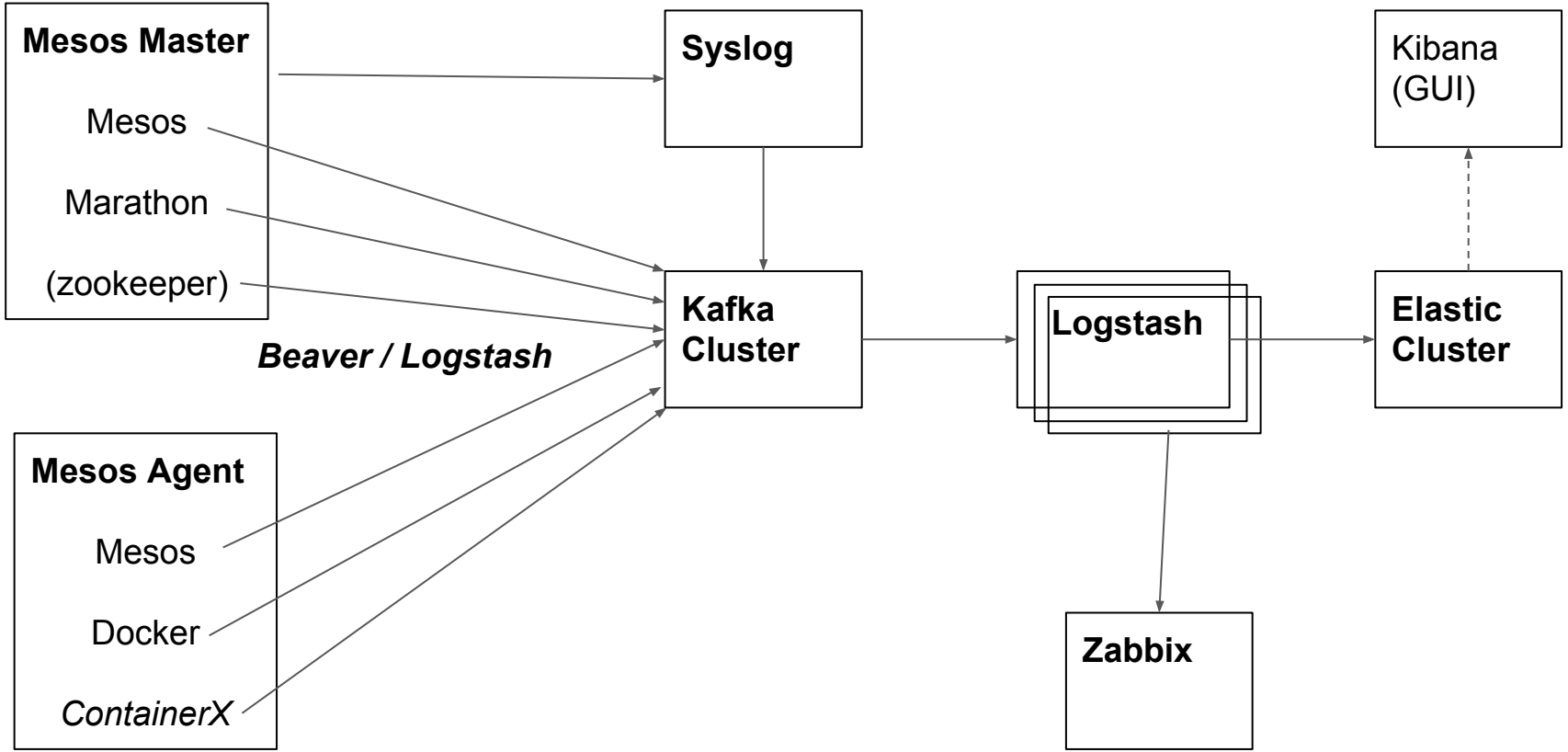
The Twelve-Factor App: https://12factor.net/

https://12factor.net/logs:

“A twelve-factor app never concerns itself with routing or storage of its output stream.

It should not attempt to write to or manage logfiles. Instead, each running process writes its event stream, unbuffered, to **stdout”**

Log Monitoring - Log flow



Log Monitoring - Logstash

Logstash - as log sender tool
https://www.elastic.co/guide/en/logstash/2.3/introduction.html

Mantl project
logstash config for **mesos agent** (ansible):
*~/mantl/roles/logstash/templates/**logstash.conf.j2**:*

```
file {
    path => [ "/logs/slaves/*/frameworks/*/executors/*/runs/*/stdout",
"/logs/slaves/*/frameworks/*/executors/*/runs/*/stderr" ]
    type => "mesos-framework-logs"
  }
```

Log Monitoring - Beaver

Beaver - python log sender tool
https://github.com/python-beaver/python-beaver

Pros:

- Do not require java
- Easy to config

Installation with ansible:
https://github.com/azavea/ansible-beaver

Note:
Docker log router: Logspout
https://github.com/gliderlabs/logspout

Log Monitoring - Beaver config

```
/etc/beaver/
├── beaver.ini
└── conf.d
    └── mesos.conf
```

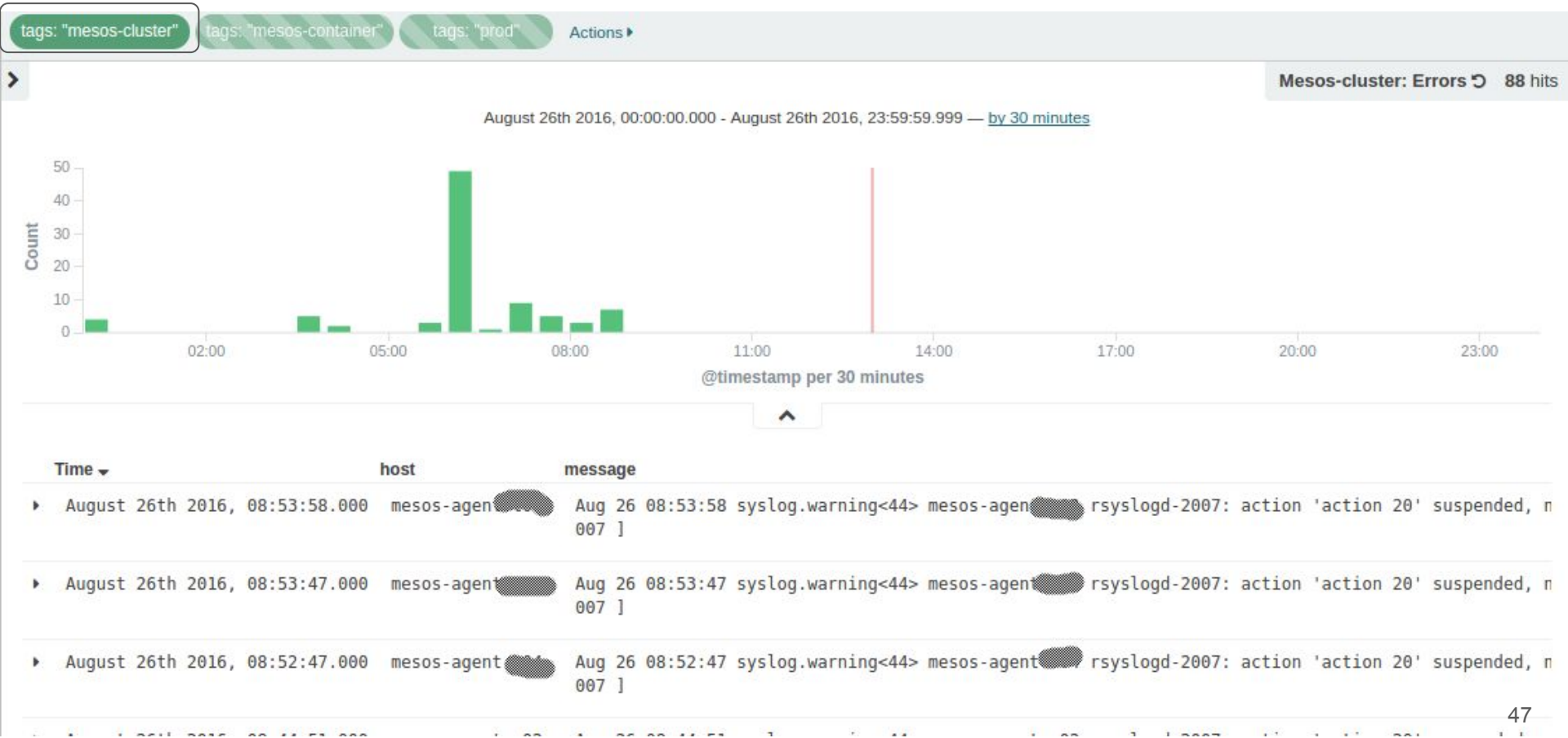
```
[beaver]
kafka_topic: elkprod
logstash_version: 1
kafka_hosts: kafkaX:9092,kafkaY:9092,kafkaZ:9092,kafkaXX:9092
; Only
queue_timeout: 43200
transport: kafka
```

Log Monitoring - Beaver config - mesos.conf

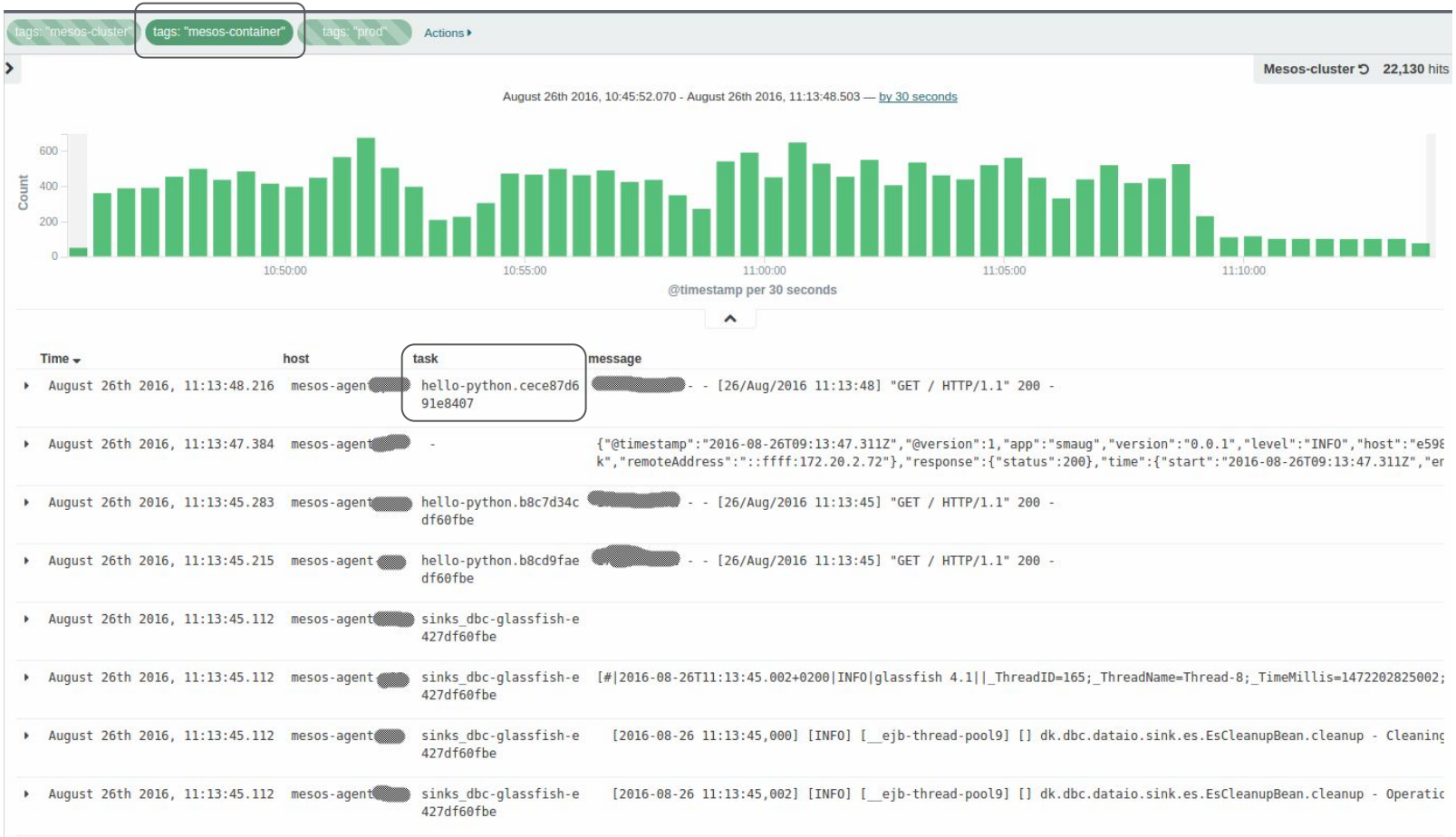
[/var/log/mesos/mesos-*.WARNING]
type: mesos
tags: mesos-cluster
[/var/log/mesos/mesos-*.INFO]
type: mesos
tags: mesos-cluster
[/var/log/mesos/mesos-*.ERROR]
type: mesos
tags: mesos-cluster
[/var/log/mesos/mesos-*.FATAL]
type: mesos
tags: mesos-cluster
[/data/mesos/slaves/*/frameworks/*/executors/*/runs/*/std*]
exclude: (latest)
type: mesos
tags: mesos-container

Log Monitoring - mesos-cluster

Log Monitoring - Error logs



Log Monitoring - mesos-container



Log Monitoring - Logstash

Logstash as log management tool

#“file” frem beaver:
#/data/mesos/slaves/a7d7fc82-c8de-4aff-84b1-f1d5c578efc7-S5/frameworks/a141ab38-8082-4c50-b04f-ff762b850aa2-0000/executors/**prod_dataio_dbc-glassfish-harvester.760fedd6-684b-11e6-bfc6-0242c91e8407...**

```
grok {
  match => { "file" =>
'/data/mesos/slaves/%{DATA}/frameworks/%{DATA}/executors/%{DATA:tags}_%{DATA:group}_%{DATA:task}
/runs/%{GREEDYDATA}'}
}
```

(Cons: grok filter need revision)

 # task: dbc-glassfish-harvester.760fedd6-684b-11e6-bfc6-0242c91e8407

```
grok {
  match => { "task" => '%{DATA:application}\.%{DATA}'}
}
```

Log Monitoring - Logstash - zabbix

t application		dbc-glassfish-harvester
t file		dbc-glassfish-harvester
t file2		/data/mesos/slaves/a7d7fc82-c8de-4aff-84b1-f1d5c578efc7-S5/frameworks/a141ab38-8082-40/executors/prod_dataio_dbc-glassfish-harvester.b8afcc98-69fd-11e6-87a8-0242c91e8407/44-9eef738c4a5e/stderr
t group		dataio
t host		mesos-agen[redacted]
t message		[2016-08-27 10:50:00,004] [INFO] [__ejb-thread-pool11] [] dk.dbc.dataio.harvester.ran.reload - Applying configuration: []|#]
t tags		mesos-container, prod
t task		dbc-glassfish-harvester.b8afcc98-69fd-11e6-87a8-0242c91e8407

Log Monitoring - Logstash - zabbix

Zabbix output plugin for Logstash: Zabbix conference 2015 - by **untergreek**: http://www.slideshare.net/Zabbix/aaron-mildenstein-using-logstash-with-zabbix

```
# Look for java Exceptions
filter {
  if [message] =~ "Exception" {
    mutate {
      add_field => {
        "[itemkey]" => "applog"
        "[alertmsg]" => "%{task}:    %{message}" }
      }
    }
}
```

```
output {
  if [alertmsg] {
    zabbix {
    zabbix_server_host => "zabbixhost"
    zabbix_host => "[application]"
    # Single value also possible
    multi_value => [
      "[itemkey]", "[alertmsg]"
    ]
  }
}
```

Note: Similar config used for **syslog** to zabbix

Other solutions?

Data to kafka?

- https://github.com/hengyunabc/kafka-zabbix

Data to Elastic?

Data via StatsD?

Collectd zabbix write plugin?

Zabbix container module?