Coding.net 轻量级运维实践

吴柯

What ?

Coding.net 是什么 ？

关于Coding.net

Coding.net 是一个面向开发者的云端开发协作平台


项目协作


代码托管


运行空间


质量控制

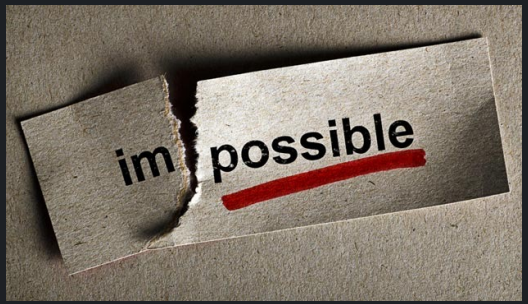

自动测试


持续集成

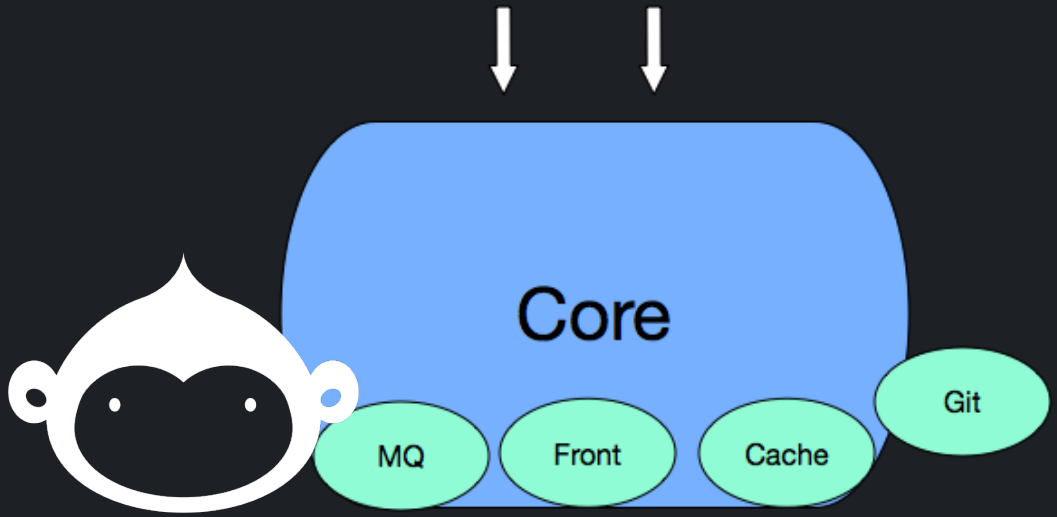

在线IDE

Coding v0.1时代

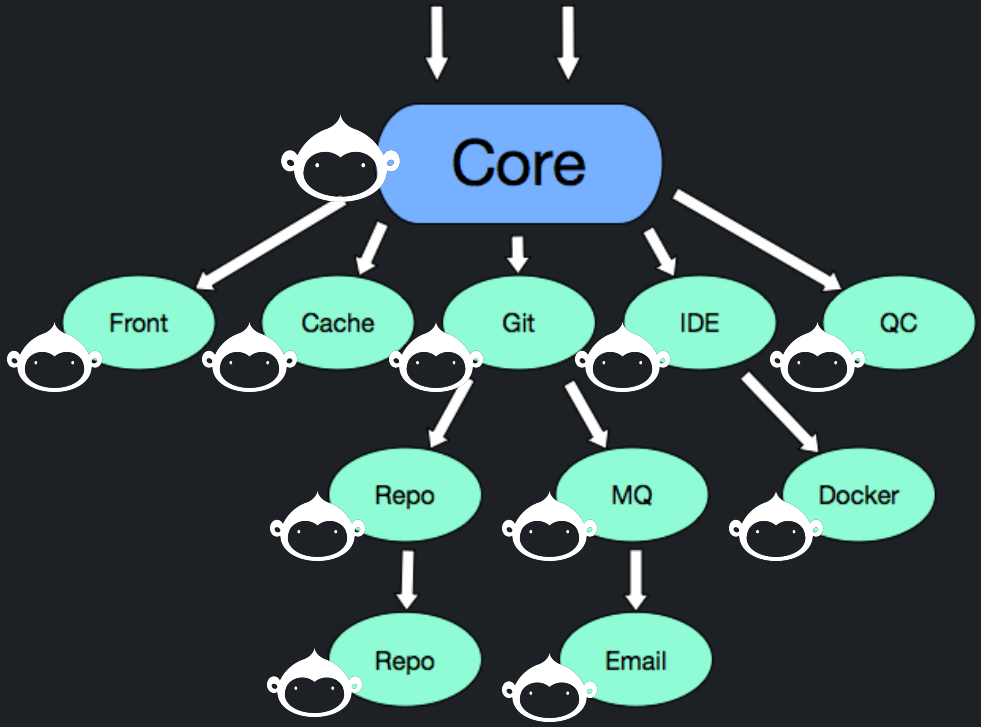
- 保证开发速度，服务模块过于耦合
- 部署上线很虐心
- 运维变更靠超人
- 监控靠有限的外部服务
- 救火常态化



一开始..



后来..

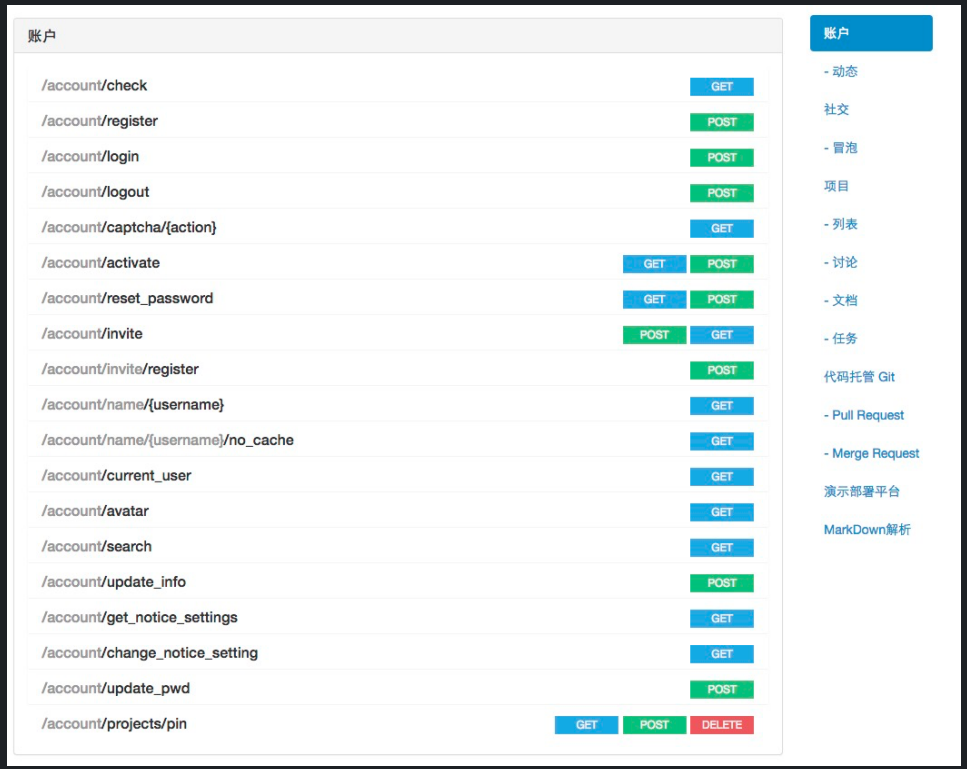


Coding v1.0

- 服务模块化
- 通信接口化
- 部署脚本化
- 主机管理批量化

Restful API

- GET
- POST
- PUT
- DELETE



- 基于SSH管理
- 批量管理&配置主机
- inventory 主机分组管理
- 支持Playbook

/hosts file：

```
[backup]
backup-1.coding.local   ansible_ssh_user=root

[core-app]
core-app-1.coding.local   ansible_ssh_user=root   ansible_ssh_port=2222
core-app-2.coding.local   ansible_ssh_user=root   ansible_ssh_port=2222

[core-db]
core-db-1.coding.local   ansible_ssh_user=root
core-db-2.coding.local   ansible_ssh_user=root

[core-cache]
core-cache-1.coding.local   ansible_ssh_user=root
core-cache-2.coding.local   ansible_ssh_user=root

[core-queue]
core-queue-1.coding.local   ansible_ssh_user=root
core-queue-2.coding.local   ansible_ssh_user=root

[core:children]
core-app
core-db
core-cache
core-queue
```

批量远程操作

```
~$ ansible 'core' -i /home/ubuntu/ansible/hosts -a ‘uptime'

core-cache.coding.local | success | rc=0 >>
 17:50:56 up 152 days, 38 min,  1 user,  load average: 0.30, 0.37, 0.36

core-db.coding.local | success | rc=0 >>
 17:50:56 up 152 days, 33 min,  1 user,  load average: 0.23, 0.23, 0.18

core-queue.coding.local | success | rc=0 >>
 17:50:56 up 152 days, 37 min,  1 user,  load average: 0.03, 0.04, 0.05

core-app.coding.local | success | rc=0 >>
 17:50:56 up 152 days, 36 min,  1 user,  load average: 0.99, 0.89, 0.95
……
~$
```

Puppet & Ansible

Puppet
- 支持主流的操作系统
- 历史悠久相对成熟
- 基于Ruby 性能相对慢

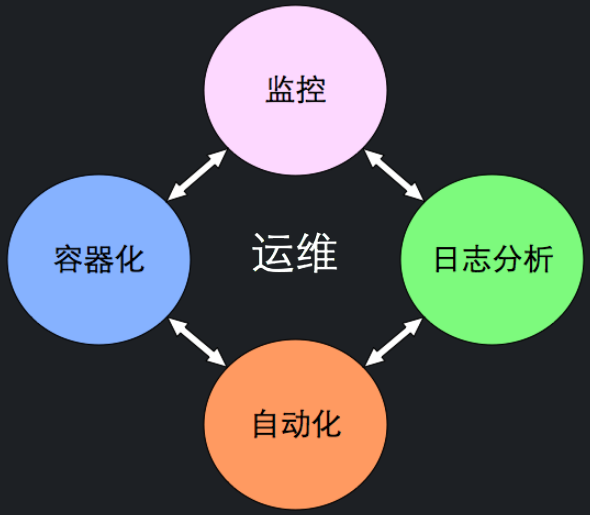
Ansible
- 基于SSH 无代理
- 安装简单
- 命令行操作简单方便
- 基于python 速度快，性能好

但是...

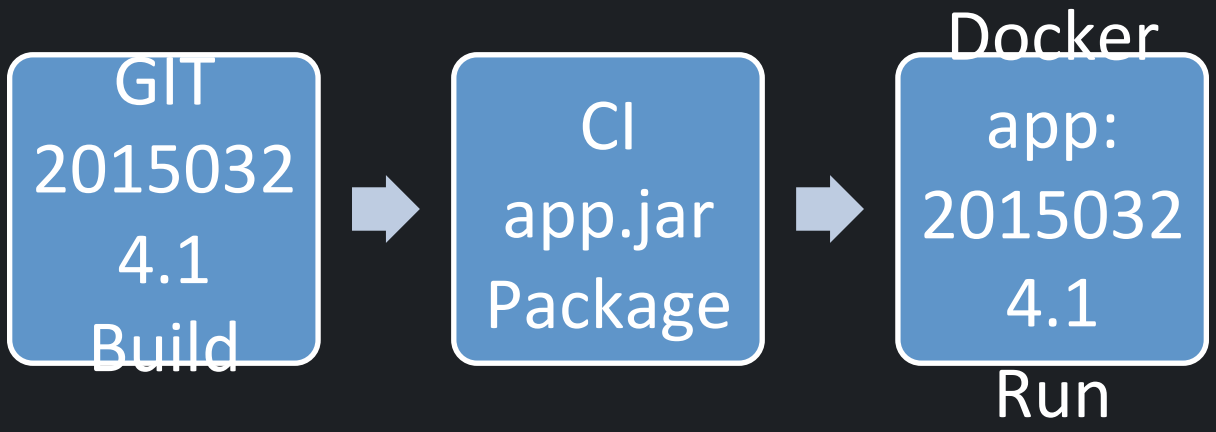
- 60%的批量操作是配置环境
- 众多微服务难管理
- 自动化脚本难维护
- 监控体系不健全
- 日志分散数据沉睡

Coding v2.0

- 服务容器化
- 操作代码化
- 部署自动化
- 建立监控体系
- 日志收集&分析

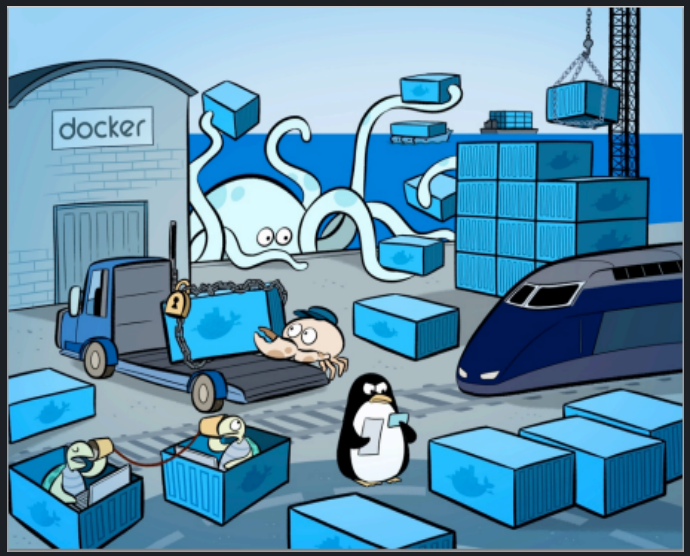


容器化过程


GIT
2015032
4.1
Build
CI
app.jar
Package
Docker
app:
2015032
4.1
Run

运行代码化

```
jobs : <
 name: “coding-front"
 image: “coding-front:X“
>
jobs : <
  name: “coding-backend"
  image: “coding-backend:Y"
  env: <
      key: “xxxxxx”
      value: “xxxxxx”
>
jobs: <
 name: “mysql"
 image: “mysql:5.5"
>
```

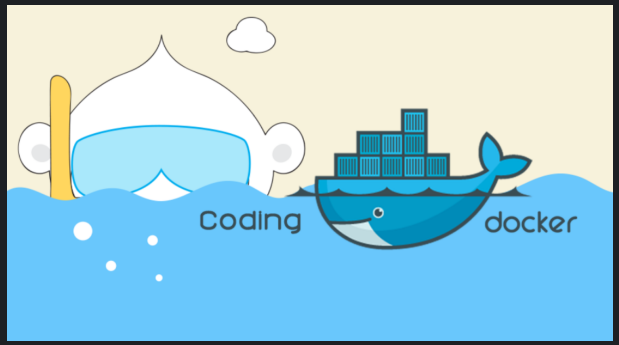


运行代码化

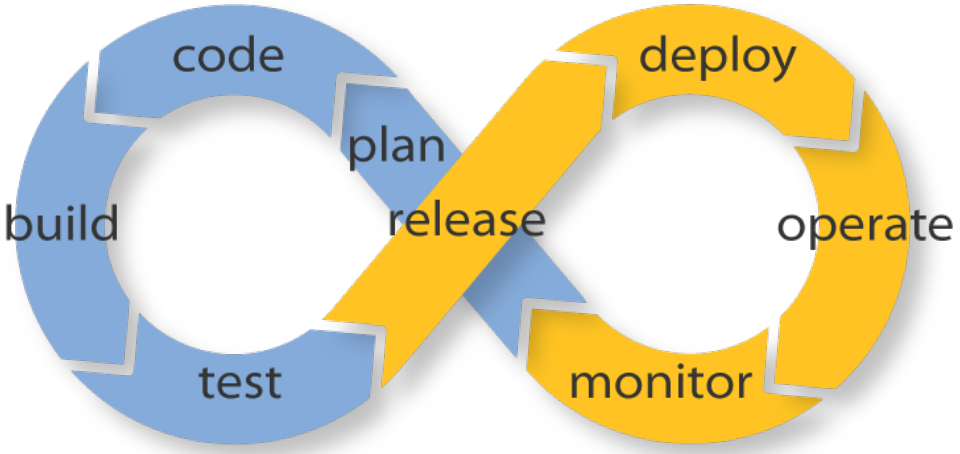
```
$ go run stack.go up
Job: coding-front
Image: coding-front:X
State: [/coding-front_X]: Up 1s

Job: coding-backend
Image: coding-backend:Y
State:  [/coding-backend_Y]: Up 1s

Job: mysql
Image: mysql:5.5
State: [/mysql_5.5]: Up 1s
```

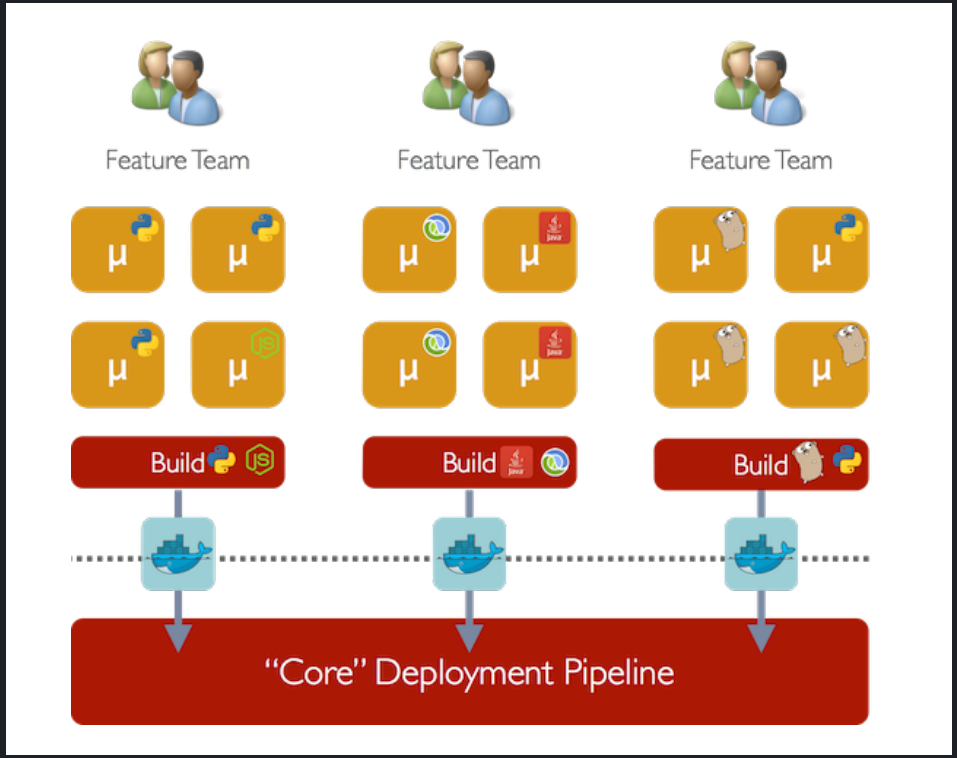


DevOps

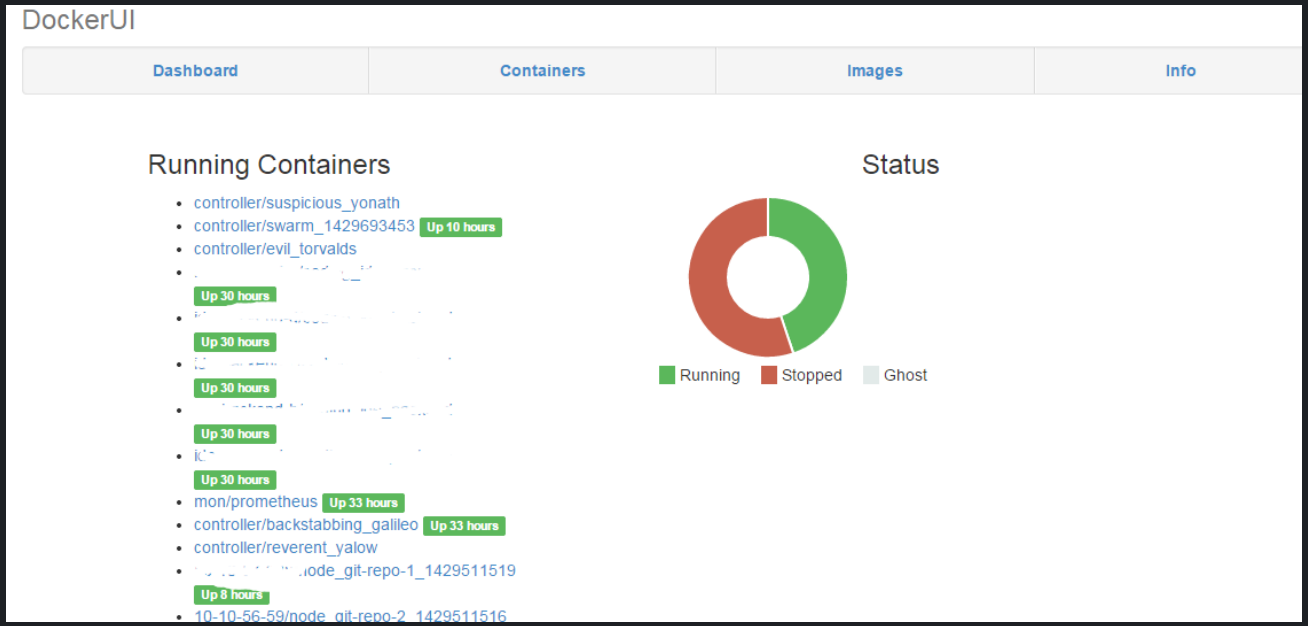


Endless Possibilities: DevOps can create an infinite loop of release and feedback for all your code and deployment targets.

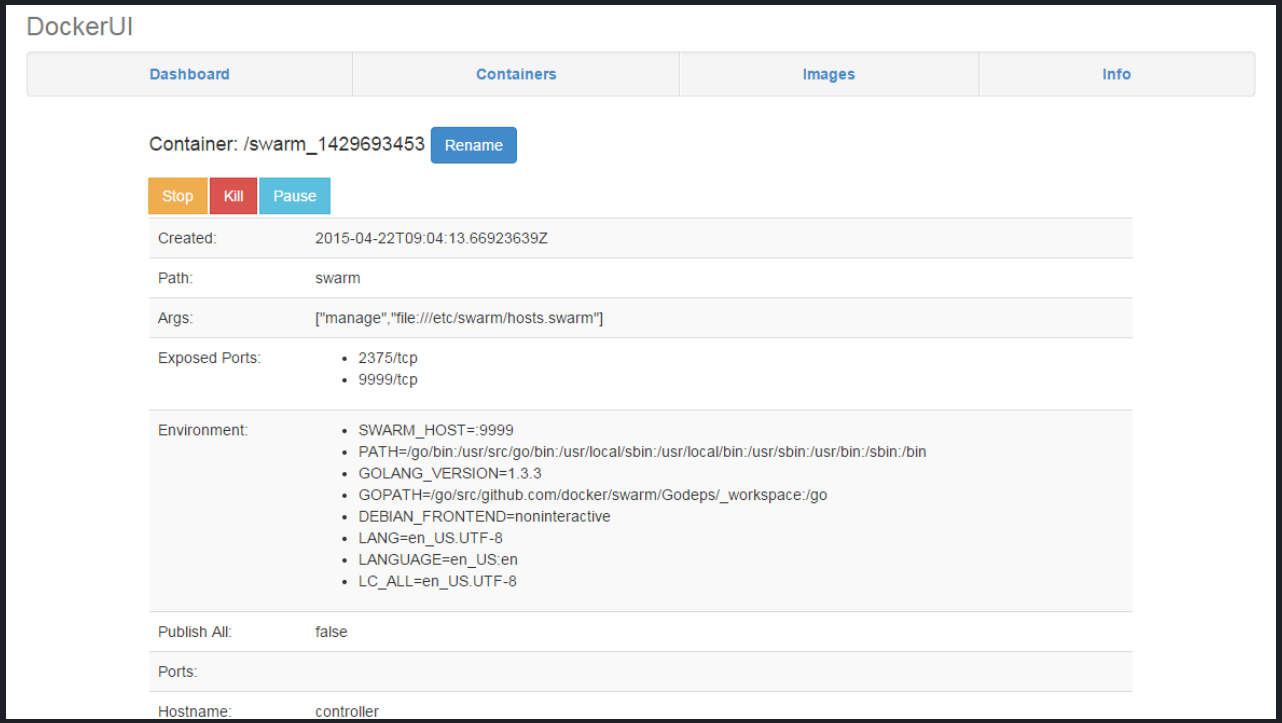
部署：Pipeline 模式



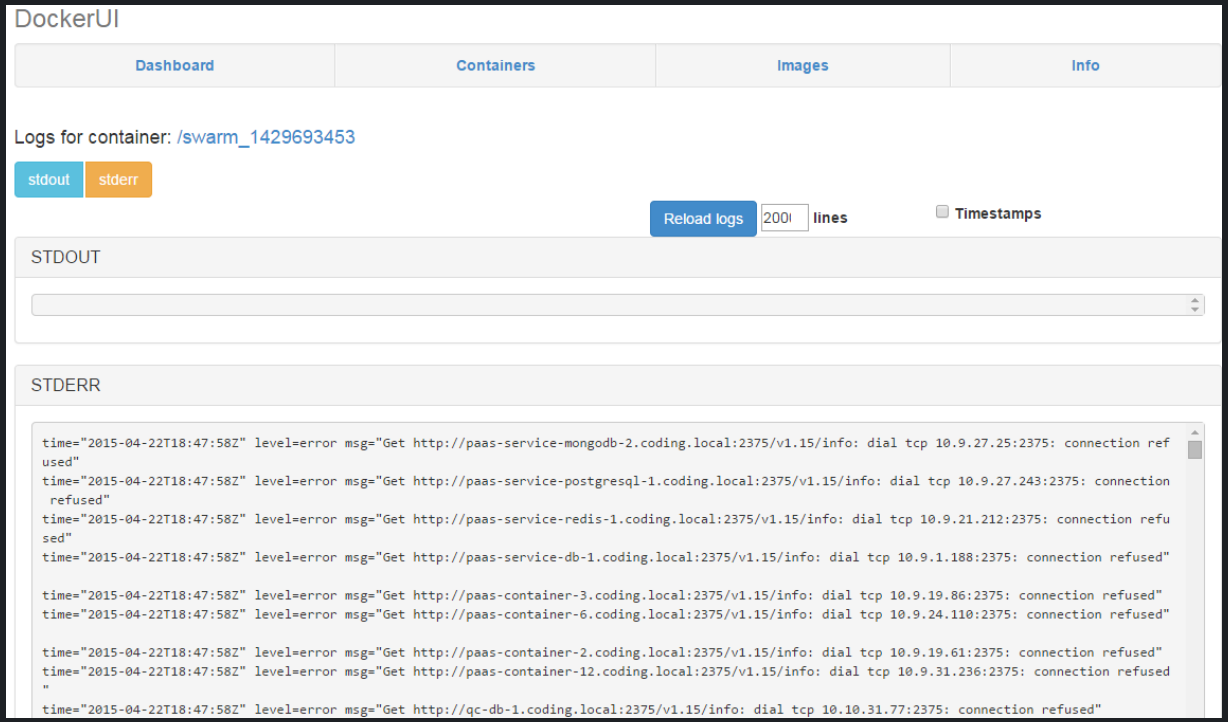
部署 Docker UI


DockerUI
Dashboard
Containers
Images
Info
Running Containers
controller/suspicious_yonath
controller/swarm_1429693453
Up 10 hours
controller/evil_torvalds
Up 30 hours
Up 30 hours
Up 30 hours
Up 30 hours
Up 30 hours
mon/prometheus
Up 33 hours
controller/backstabbing_galileo
Up 33 hours
controller/reverent_yalow
Up 8 hours
Status
Running
Stopped
Ghost

部署 Docker UI


DockerUI
Dashboard
Containers
Images
Info
Container: /swarm_1429693453
Rename
Stop
Kill
Pause
Created:
2015-04-22T09:04:13.66923639Z
Path:
swarm
Args:
["manage","file:///etc/swarm/hosts.swarm"]
Exposed Ports:
2375/tcp
9999/tcp
Environment:
SWARM_HOST=:9999
PATH=/go/bin:/usr/src/go/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
GOLANG_VERSION=1.3.3
GOPATH=/go/src/github.com/docker/swarm/Godeps/_workspace:/go
DEBIAN_FRONTEND=noninteractive
LANG=en_US.UTF-8
LANGUAGE=en_US:en
LC_ALL=en_US.UTF-8
Publish All:
false
Ports:
Hostname:
controller

部署 Docker UI

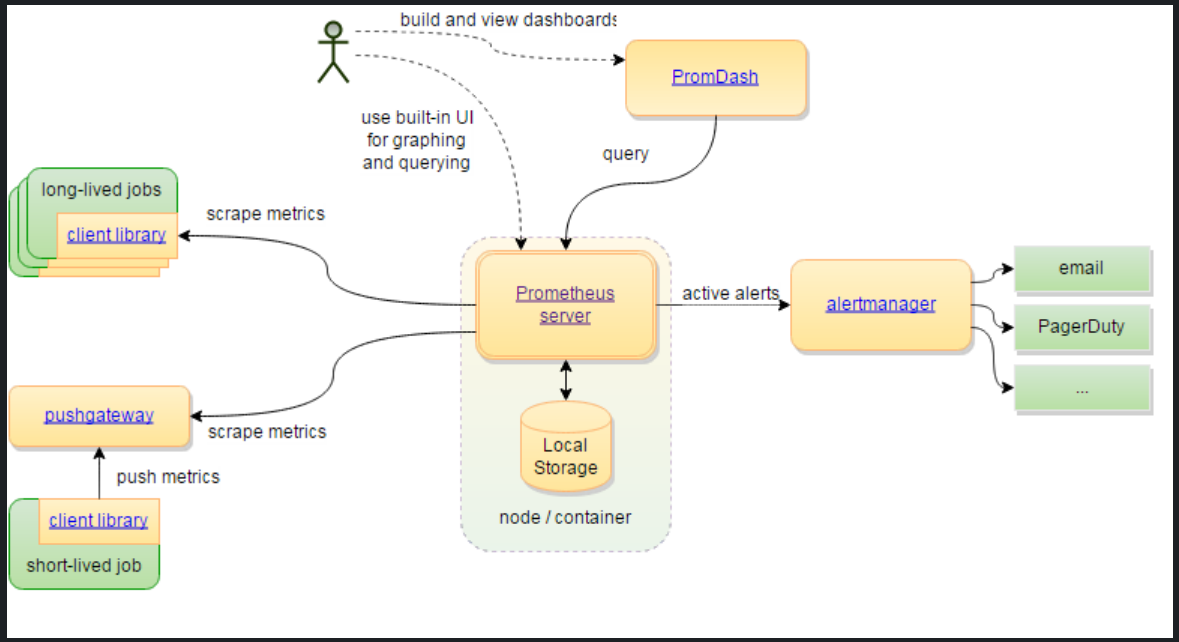

DockerUI
Dashboard
Containers
Images
Info
Logs for container: /swarm_1429693453
stdout
stderr
Reload logs
200
lines
Timestamps
STDOUT
STDERR
time="2015-04-22T18:47:58Z" level=error msg="Get http://paas-service-mongodb-2.coding.local:2375/v1.15/info: dial tcp 10.9.27.25:2375: connection refused"
time="2015-04-22T18:47:58Z" level=error msg="Get http://paas-service-postgresql-1.coding.local:2375/v1.15/info: dial tcp 10.9.27.243:2375: connection refused"
time="2015-04-22T18:47:58Z" level=error msg="Get http://paas-service-redis-1.coding.local:2375/v1.15/info: dial tcp 10.9.21.212:2375: connection refused"
time="2015-04-22T18:47:58Z" level=error msg="Get http://paas-service-db-1.coding.local:2375/v1.15/info: dial tcp 10.9.1.188:2375: connection refused"
time="2015-04-22T18:47:58Z" level=error msg="Get http://paas-container-3.coding.local:2375/v1.15/info: dial tcp 10.9.19.86:2375: connection refused"
time="2015-04-22T18:47:58Z" level=error msg="Get http://paas-container-6.coding.local:2375/v1.15/info: dial tcp 10.9.24.110:2375: connection refused"
time="2015-04-22T18:47:58Z" level=error msg="Get http://paas-container-2.coding.local:2375/v1.15/info: dial tcp 10.9.19.61:2375: connection refused"
time="2015-04-22T18:47:58Z" level=error msg="Get http://paas-container-12.coding.local:2375/v1.15/info: dial tcp 10.9.31.236:2375: connection refused"
time="2015-04-22T18:47:58Z" level=error msg="Get http://qc-db-1.coding.local:2375/v1.15/info: dial tcp 10.10.31.77:2375: connection refused"

监控体系

Prometheus 是SoundClound开发的一个开源的系统监控和报警工具

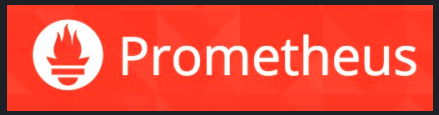
- 多维度数据模型存储 (基于时间序列的 key/value)
- 灵活简单的多维度查询语句
- 不依赖于复杂的分布式的存储模型，单节点服务即装即用
- 数据收集可以通过 HTTP 拉取，也可以通过 Gateway 推送
- 静态配置和动态发现监控目标
- 支持多种主流的图形和仪表模型

监控体系


build and view dashboards
PromDash
use built-in UI
for graphing
and querying
query
long-lived jobs
client library
scrape metrics
Prometheus
server
active alerts
alertmanager
email
PagerDuty
...
pushgateway
scrape metrics
Local
Storage
node / container
push metrics
client library
short-lived job

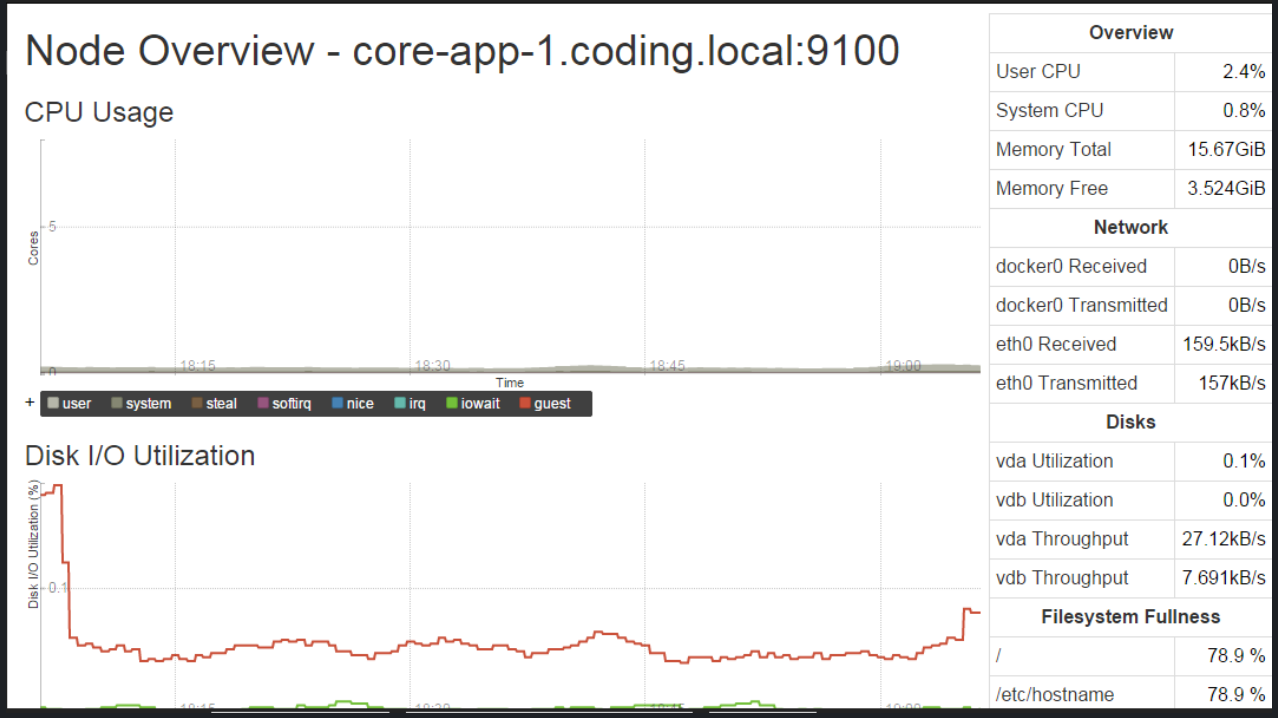
Prometheus & InfluxDB

- 存储事件的所有元数据
- 数据占用空间大
- 分布式集群设计，部署管理复杂
- 适用于大型数据处理



- 只存储数字化的数据
- 数据占用空间小
- 单节点即可运行，部署管理简单

监控体系


Node Overview - core-app-1.coding.local:9100
CPU Usage
Cores
5
0
18:15
18:30
18:45
19:00
Time
user
system
steal
softirq
nice
irq
iowait
guest
Disk I/O Utilization
Disk I/O Utilization (%)
0.1
Overview
User CPU 2.4%
System CPU 0.8%
Memory Total 15.67GiB
Memory Free 3.524GiB
Network
docker0 Received 0B/s
docker0 Transmitted 0B/s
eth0 Received 159.5kB/s
eth0 Transmitted 157kB/s
Disks
vda Utilization 0.1%
vdb Utilization 0.0%
vda Throughput 27.12kB/s
vdb Throughput 7.691kB/s
Filesystem Fullness
/ 78.9 %
/etc/hostname 78.9 %

配置代码化

master coding-ops / prod / monitoring / conf / consoles / node-overview.html

test 8天前 21acbc0964

Update Menu, importing Nodes graphs.

file | 5.51 KB 编辑 原始数据 按行查看 提交历史 删除

```
{{ template "head" . }}

{{ template "prom_right_table_head" }}
  <tr><th colspan="2">Overview</th></tr>
  <tr>
    <td>User CPU</td>
    <td>{{ template "prom_query_drilldown" (args (printf "sum(rate(node_cpu{job='node',instance='%s',mode='user'}[5m])) * 100 ,
  </tr>
  <tr>
    <td>System CPU</td>
    <td>{{ template "prom_query_drilldown" (args (printf "sum(rate(node_cpu{job='node',instance='%s',mode='system'}[5m])) * 10(
  </tr>
  <tr>
    <td>Memory Total</td>
    <td>{{ template "prom_query_drilldown" (args (printf "node_memory_MemTotal{job='node',instance='%s'}" .Params.instance) "B'
  </tr>
  <tr>
    <td>Memory Free</td>
    <td>{{ template "prom_query_drilldown" (args (printf "node_memory_MemFree{job='node',instance='%s'}" .Params.instance) "B"
  </tr>
<tr>
```

日志收集

Logstash conf

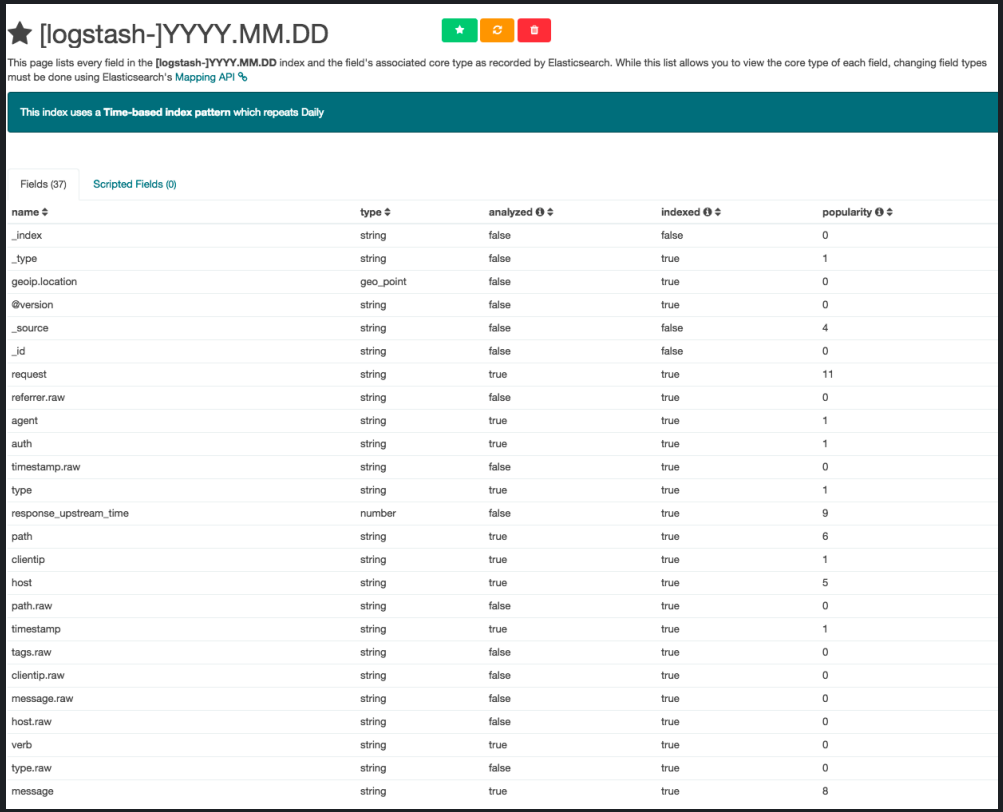
```
input {
  file {
    path => "/data/log/nginx/access.log"
    type => "nginx-access"
  }
}
filter {
  grok {
    type => "nginx-access"
    match => {
        "message" => "(?:%{IPORHOST:clientip}|-) - %{NUMBER:response_time:float} - (?:%
{NUMBER:response_upstream_time:float}|-) - %{USER:auth} \[%{HTTPDATE:timestamp}\] \"(?:
%{WORD:verb} %{NOTSPACE:request}(?: HTTP/%{NUMBER:httpversion})?|%
{DATA:rawrequest})\" %{NUMBER:response:int} (?:%{NUMBER:bytes:int}|-) %{QS:referrer} %
{QS:agent}"
    }
  }
  date {
    type => "nginx-access"
    match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ]
  }
}
output {
  elasticsearch {
        host => “xx.xx.xx.xx"
        protocol => transport
  }
}
```

Input Type

- file
- log4j
- eventlog
- redis
- syslog
- snmptrap
- websocket

…

日志收集



[logstash-]YYYY.MM.DD

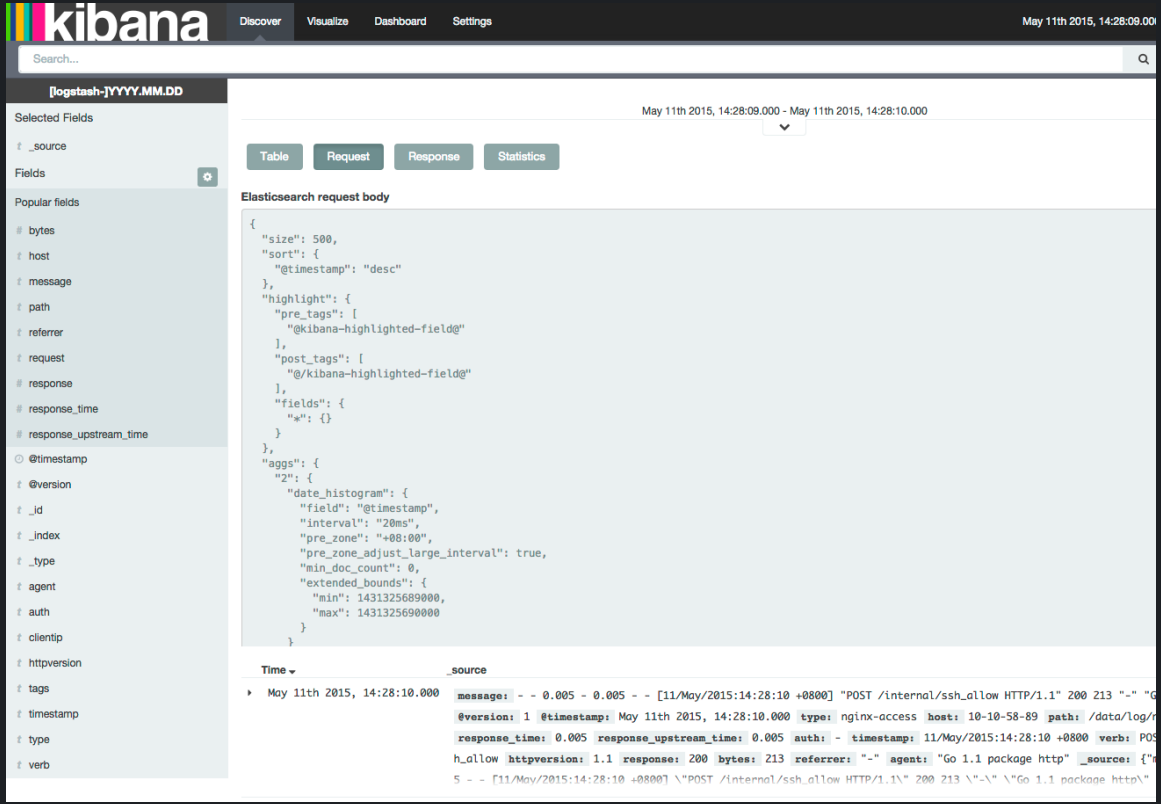
This page lists every field in the **[logstash-]YYYY.MM.DD** index and the field's associated core type as recorded by Elasticsearch. While this list allows you to view the core type of each field, changing field types must be done using Elasticsearch's Mapping API

This index uses a **Time-based index pattern** which repeats Daily

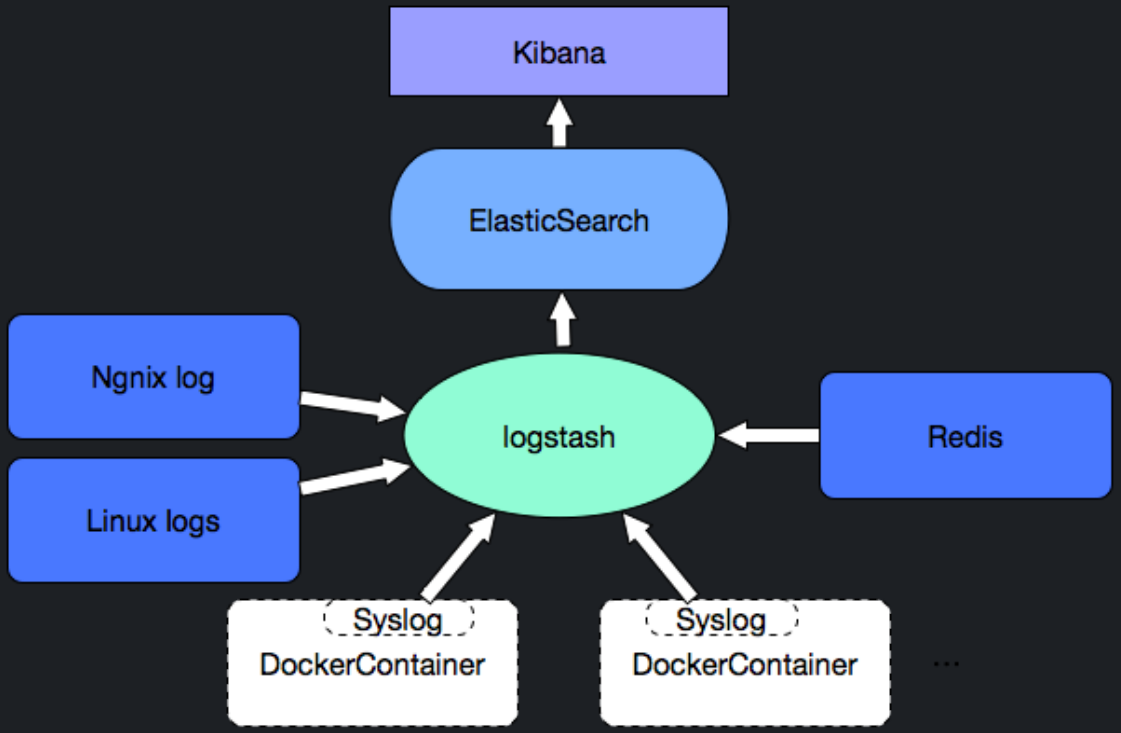
Fields (37) Scripted Fields (0)

name	type	analyzed	indexed	popularity
_index	string	false	false	0
_type	string	false	true	1
geoip.location	geo_point	false	true	0
@version	string	false	true	0
_source	string	false	false	4
_id	string	false	false	0
request	string	true	true	11
referrer.raw	string	false	true	0
agent	string	true	true	1
auth	string	true	true	1
timestamp.raw	string	false	true	0
type	string	true	true	1
response_upstream_time	number	false	true	9
path	string	true	true	6
clientip	string	true	true	1
host	string	true	true	5
path.raw	string	false	true	0
timestamp	string	true	true	1
tags.raw	string	false	true	0
clientip.raw	string	false	true	0
message.raw	string	false	true	0
host.raw	string	false	true	0
verb	string	true	true	0
type.raw	string	false	true	0
message	string	true	true	8

分析处理


kibana
Discover
Visualize
Dashboard
Settings
May 11th 2015, 14:28:09.000
Search...
[logstash-]YYYY.MM.DD
Selected Fields
_source
Fields
Popular fields
bytes
host
message
path
referrer
request
response
response_time
response_upstream_time
@timestamp
@version
_id
_index
_type
agent
auth
clientip
httpversion
tags
timestamp
type
verb
May 11th 2015, 14:28:09.000 - May 11th 2015, 14:28:10.000
Table
Request
Response
Statistics
Elasticsearch request body
{
"size": 500,
"sort": {
"@timestamp": "desc"
},
"highlight": {
"pre_tags": [
"@kibana-highlighted-field@"
],
"post_tags": [
"@/kibana-highlighted-field@"
],
"fields": {
"*": {}
}
},
"aggs": {
"2": {
"date_histogram": {
"field": "@timestamp",
"interval": "20ms",
"pre_zone": "+08:00",
"pre_zone_adjust_large_interval": true,
"min_doc_count": 0,
"extended_bounds": {
"min": 1431325689000,
"max": 1431325690000
}
}
Time
_source
May 11th 2015, 14:28:10.000
message: - - 0.005 - 0.005 - - [11/May/2015:14:28:10 +0800] "POST /internal/ssh_allow HTTP/1.1" 200 213 "-" "G
@version: 1 @timestamp: May 11th 2015, 14:28:10.000 type: nginx-access host: 10-10-58-89 path: /data/log/r
response_time: 0.005 response_upstream_time: 0.005 auth: - timestamp: 11/May/2015:14:28:10 +0800 verb: POS
h_allow httpversion: 1.1 response: 200 bytes: 213 referrer: "-" agent: "Go 1.1 package http" _source: {"m
5 - - [11/May/2015:14:28:10 +0800] \"POST /internal/ssh_allow HTTP/1.1\" 200 213 \"-\" \"Go 1.1 package http\"

日志收集&分析


Kibana
ElasticSearch
Ngnix log
Linux logs
logstash
Redis
Syslog
DockerContainer
Syslog
DockerContainer
...

wuke@coding.net