

ELK 简介

因为本次培训的时间为 1 小时，所以相关的内容我会介绍的比较简略。

本文档的访问地址：



我是云平台部的系统架构师 蔡仲华

邮箱 `zhonghuacai@pateo.com.cn`

本文档的受众为：希望学习如何通过 Kibana 能够快速定位目标日志的开发、运维人员。

（有兴趣深入了解的话，推荐阅读官方文档，或者加入日志微信群里讨论。）

通过本次培训，你可以了解到：

- Elasticsearch 的设计简介
- ELK 的日志处理流程
- Kibana 的常用操作
- 日志接入的相关事宜

ElasticSearch 简介

TL;DR: ES 是一款搜索引擎，负责数据的存储和搜索，支持高度弹性的分布式部署。

数据结构

最重要的是数据结构，你需要知道数据是如何在 ES 中存储的。

可以简单的把 ES 的数据存储理解为一个关系型数据库。

所有的数据都以文档（document）的形式存储，按照粒度，从下往上区分 ES 的数据结构，可以分为：

- token：词组
- field：document 中的每一项
- document：文档
- type（mapping）：文档类型（可以理解成 table）
- shard：分片
- index：索引（可以理解成 db）

一个 document 的例子:

```
1 {
2   "_index": "sit-spring-logs-2018.05.18-1", index 名
3   "_type": "logs", type (mapping) 名
4   "_id": "AWOaok_wwbNxasisvUuN", 文档 id
5   "_version": 1,
6   "_score": null,
7   "_source": { 文档内容
8     "container_id": "31401636bd4ea1558f637cfe83399476eacab9dbfe14776cd064fcdb417eef3e",
9     "container_name": "/mesos-72bc65e1-09b1-4f97-bc3f-371819a4a18d-S1.fc938e-f536-40c1-9720-fbe1a8cea347",
10    "source": "stdout",
11    "app": "mcpweather", 每一行是一个 field: value
12    "level": "INFO",
13    "thread": "http-nio-8080-exec-8",
14    "class": "com.pateo.qingcloud.cp.adapter.utils.HttpUtil.doGet",
15    "line": "200", 每一个字符串, 又会被拆分为一组 terms
16    "message": "httpPath:http://telematics.autonavi.com/ws/mapapi/geo/reversecode/?",
17    "datasource": "spring",
18    "@timestamp": "2018-05-21T02:58:25.114000000+00:00"
19  },
20  "fields": {
21    "@timestamp": [
22      1526871505114
23    ]
24  }
25 }
```


和 RDBMS 做一个对比，可以简化理解为：

RDBMS	ES
db	index
table	type (mapping)
row	document
column	field

之所以把 ES 和 RDBMS 而不是 no-sql 对比，就是因为 ES 的数据是需要预先定义 mapping 的，这一点和 RDBMS 更为接近。

而且 ES 官方还建议一个 index 只建立一个 type，所以使其和 MySQL 中 table、row 的概念更像了。

分词

mapping 中会为每一个 field 定义类型，对于字符串类型的数据，ES 会对其进行分词。

不同的分词算法，对应不同的 analyzer，而一个标准的 analyzer，由三部分组成：

- character filter：过滤和转换字符；
- tokenizer：分词；
- token filter：过滤和转换词组；

这里不做详细展开了，需要了解的就是，任何日志消息，在 ES 中都是以词组（terms、tokens）的形式存储的。

（terms 是字符串的词组，tokens 是 terms 加上坐标等其他元数据）

所以你搜索的时候，其实搜索的也是词汇，ES 会根据你搜索的词汇对文档进行评分，然后返回给你评分最高的文档。

这种查询方式，被称为反向索引（倒排索引、inverted index）

反向索引长这样，通过词组去查找文档：

tokens	docu1	docu2
word1	✓	✓
word2		✓
word3		

集群

ES 是一个分布式的集群，所以每一个 index 都会按照配置，被拆分为数个 shards 分散存放于不同的机器上。

而且为了保证高可用，还可以为 shards 配置 replica，ES 会尽可能的将所有的 primary shards 和 replica 分散存储于不同的机器上

Cluster

Node1

Shard1

Replica5

Shard3

Replica6

Node2

Shard2

Replica7

Shard4

Replica8

Node3

Shard5

Replica2

Node4

Shard7

Replica1

集群中的任何一个节点，都可以通过 RESTful 提供完全的访问。

每一个节点，可以有三种身份：

- master：管理节点，负责调度、恢复、选举等；
- data：数据存储节点；
- forwarder：转发节点，只响应请求。

默认情况下，节点的身份为 master & data，如果两者皆无，则称为 forwarder 节点。

ELK 日志处理流程

从搜集应用产生的日志，到集中式日志解析，再到最后的 Kibana 呈现，这一日志流处理过程中涉及的技术栈，称之为——ELK。

也就是 ElasticSearch & LogStash & Kibana 的首字母缩写。

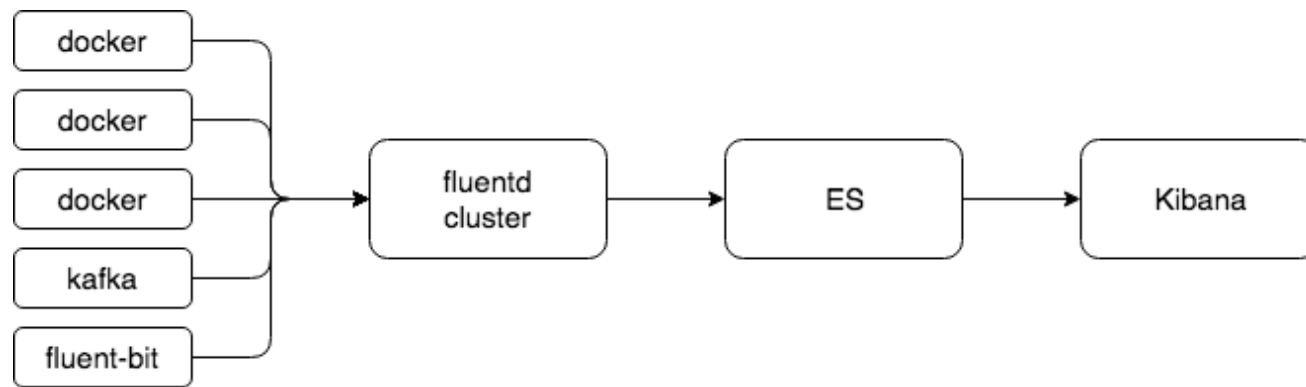
（不过我用 Fluentd 替换了 Logstash）

日志收集

我们绝大部分的应用以 docker 容器的形式运行，而 docker 原生支持 fluentd，所以直接在 marathon 里修改 docker 的配置即可：

```
"parameters": [  
  {  
    "key": "log-driver",  
    "value": "fluentd"  
  },  
  {  
    "key": "log-opt",  
    "value": "<FLUENTD_TAGS>"  
  },  
  {  
    "key": "log-opt",  
    "value": "fluentd-address=<FLUENTD_SERVER>"  
  }  
]
```

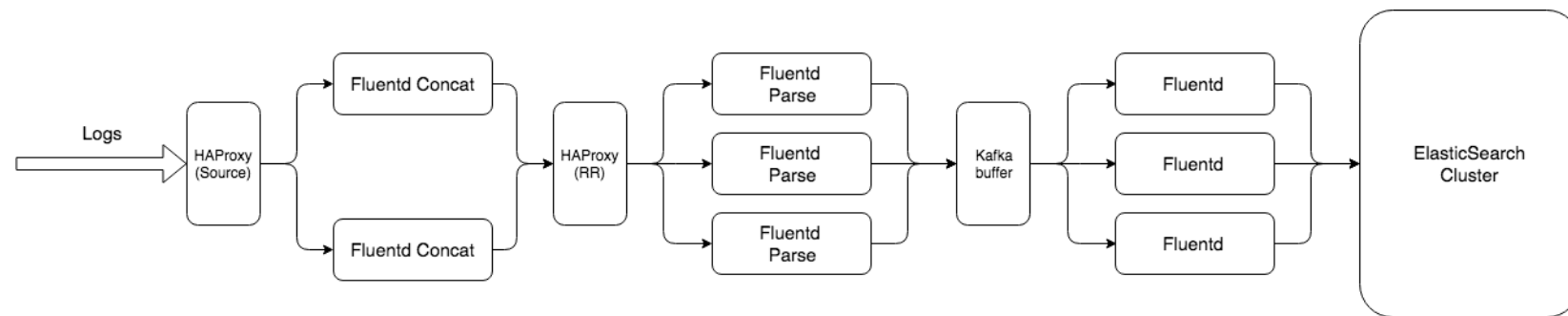
ELK (EFK) 的流程：



fluentd cluster 这边，需要对每一个应用的单独配置，包括：

- 日志的接入点
- 日志的解析格式
- 日志的存储规则

为了满足复杂的解析需求，并且保证尽可能高的解析性能，我们目前的 fluentd cluster 实际是这样的：



所以如果有新的项目组，想要接入日志平台，需要确定好以下事宜：

- 固定的日志格式（可支持 json 解析）
- 日志量的评估

然后来商讨确定：日志 tag、存储日期、解析规则。

等一切都配置好了以后，才能最终在 Kibana 上查阅到日志。

更多日志接入的配置信息，可以在公司 confluence 里搜索【日志平台】查看：

<http://confluence.pateo.com.cn/pages/viewpage.action?pageId=5636896>



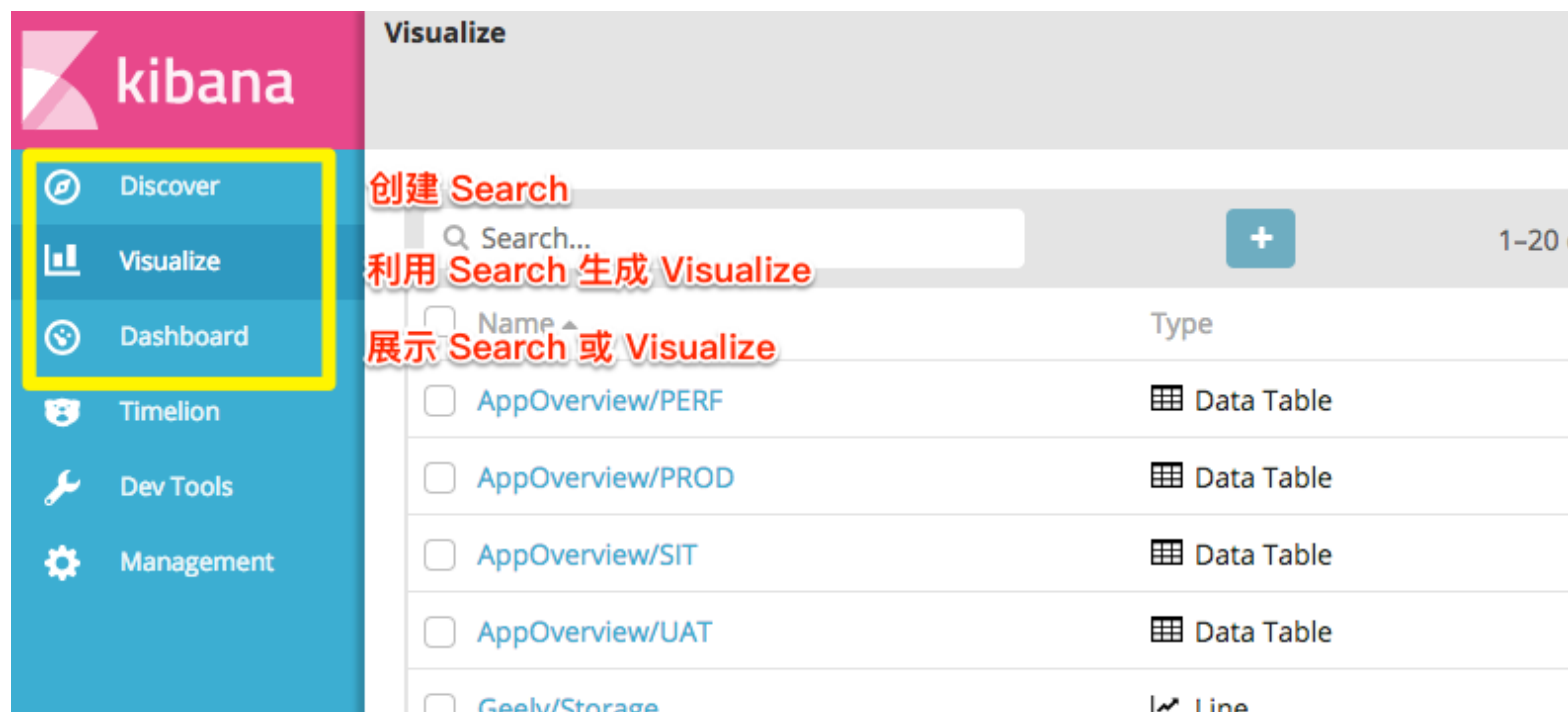
(该文档内，有日志微信群的 QR 码)

Kibana 操作简介

地址： <http://172.16.4.11/service/kibana>

(需要链接百度 VPN)

最主要操作的几个 tab



Discover 页

The screenshot shows the Kibana Discover interface. The left sidebar contains navigation links: Discover, Visualize, Dashboard, Timelion, Dev Tools, and Management. The main area displays search results for the query `status:200 AND extension:PHP`. The interface includes a search bar, a list of filters, and a table of results. The bottom of the screen shows a time range selector and a visualization of the data.

Annotations in the image:

- 符合查询的条目** (Items matching the query): Points to the search results table.
- 查询页面** (Query page): Points to the Discover link in the sidebar.
- 添加查询条件 filter** (Add query condition filter): Points to the "Add a filter" button.
- 查询已保存的搜索** (Query saved searches): Points to the "Use saved query syntax" link.
- 请从这里寻找自己要看的搜索** (Please find the search you want to see from here): Points to the saved searches dropdown menu.
- 如果没有的话, 联系管理员添加** (If not, contact the administrator to add): Points to the saved searches dropdown menu.
- 面板** (Dashboard): Points to the Dashboard link in the sidebar.

一般搜索的步骤：

1. 首先点击 discover



perf-baidubot-logs 3 hits

New Save

Open

Share

<

Last 7 days

>

Open Search



Q Saved Searches Filter...



Name ^



perf-baidubot-logs



perf-cp-logs



perf-geely-logs



prod-cp-logs



sit-cp-logs



sit-geely-logs



uat-cp-logs

uat-spark-logs

UAT spark logs from kafka

3. 选择自己要查看的日志



2. 点击 Open




4. 选择时间区间





1-8 of 8


Manage saved searches





kibana


 Discover

 Visualize

 Dashboard

 Timelion

 Dev Tools

 Management

点击 +
可以增加 filter
仅显示该字段为当前值的

Executor-0

▶	April 4th 2018, 16:06:17.398	DEBUG	DiscoveryClient-CacheRefreshExecutor-0	com.netflix.discovery.DiscoveryClient	1,074	deviceconnectorsms
▼	April 4th 2018, 16:06:17.398	DEBUG	DiscoveryClient-CacheRefreshExecutor-0	com.netflix.discovery.DiscoveryClient	1,205	deviceconnectorsms

Table

JSON

 @timestamp

  * April 4th 2018, 16:06:17.398

 t

  * AWKPsT_YKp9Gt6740-A9

 t_index

  * sit-gateway-logs

 pre

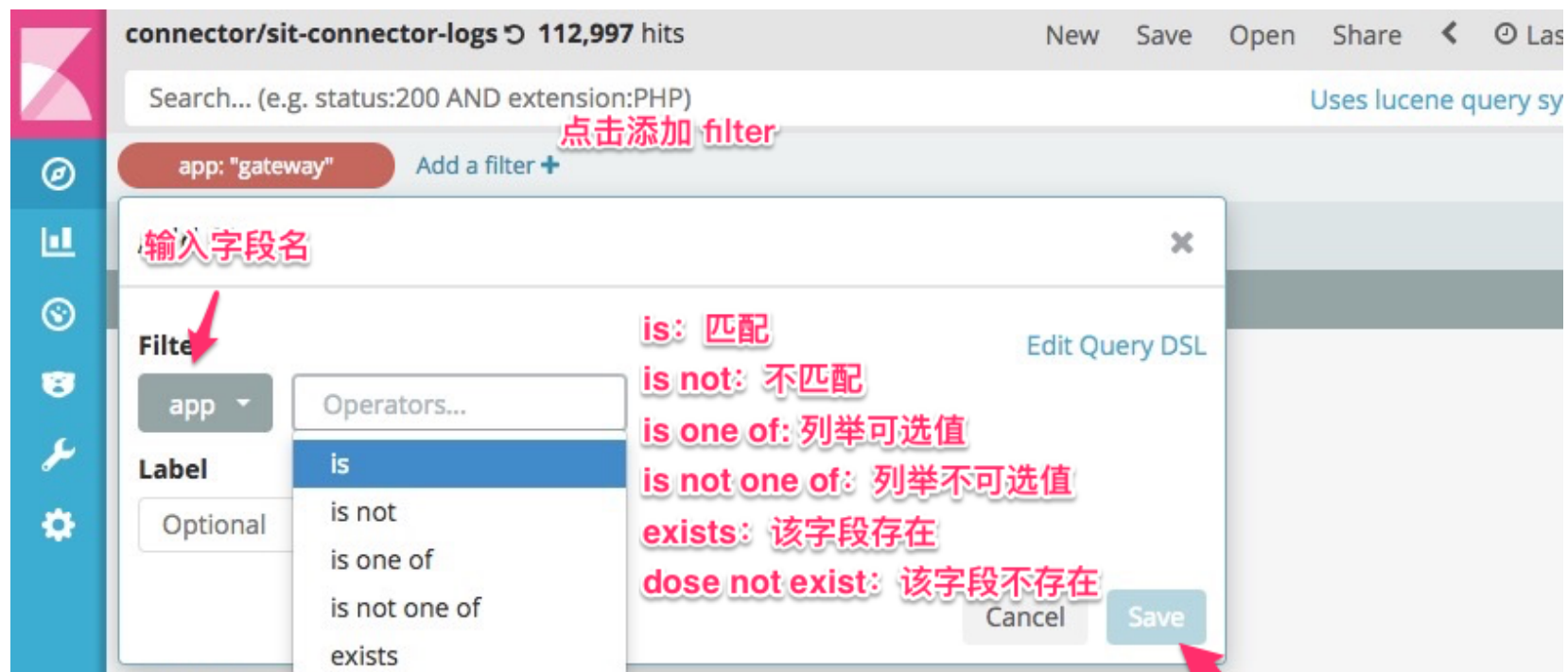
  *

点击后可展开

可以查看该日志的上下文

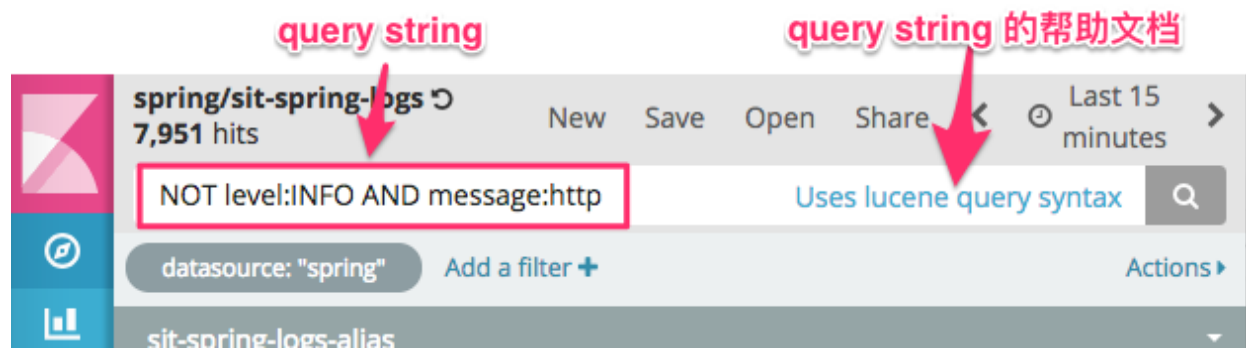
[View surrounding documents](#) [View single document](#)

善用 filter



最简单的方式就是实际操作一下

Query String 查询语句



常见的查询语句形如：

```
field1:val1 AND field2:val2 OR field3:val3 NOT field4:val4
```

```
datetime_field:[xxx TO xxx]  // 查询范围，日期也这么查
```

```
count:[xx TO xxx}  // 开闭区间
```

```
age:>10  // 对数字的比较
```

```
age:>=10
```

```
age:<10
```

```
age:<=10
```

```
field:"xxx?xxx?*"
```

灵活的运用 query string 和 filter 可以满足绝大部分的查询需求。

Visualize

Kibana 上可以通过简单的操作，将存储的 Search 展示为交互式的数据图表

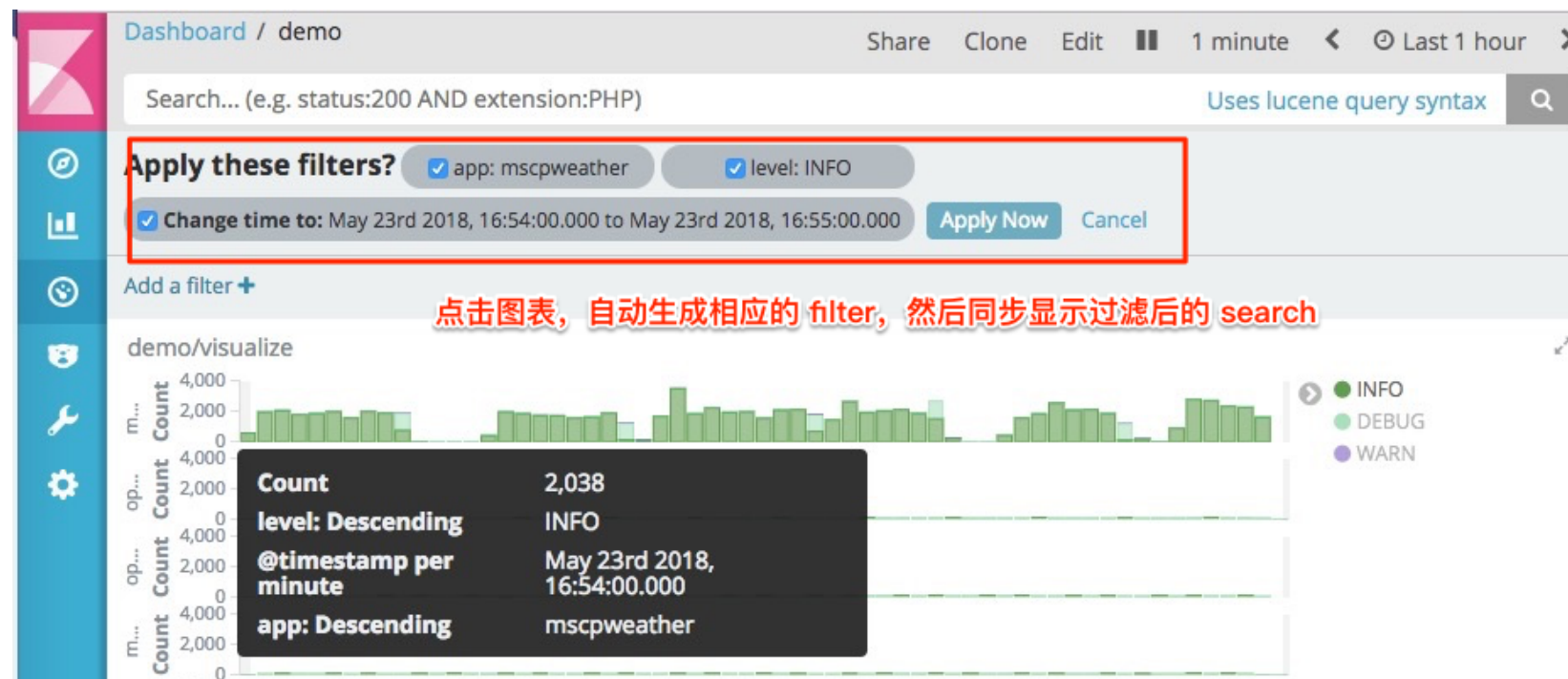
一般的图形的概念很简单，就是：

- 横轴：一般是时间
- 纵轴：一般是 count
- bucket：如何分组，分为图表内分组，和拆分图表的分组



设定好 Visualize 后，就可以在 Dashboard 里配置显示了

Dashboard 中可以将 visualize 和 search 放在一起，通过交互式的操作，点击图表自动生成 filter，并实时展现搜索结果



基于日志的二次开发

也许你希望从我们收集的日志里提取一些有效的信息，做一些二次的展示、分析工作。这样的话，你可以直接通过 Elasticsearch 的 HTTP API 拿取数据。具体的接口和操作方法，可以之后在日志群里询问。

Q&A