

大型架构及配置技术

NSD ARCHITECTURE

DAY04

内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	Kibana使用
	10:30 ~ 11:20	
	11:30 ~ 12:00	
下午	14:00 ~ 14:50	Logstash配置 扩展插件
	15:00 ~ 15:50	
	16:10 ~ 17:10	
	17:20 ~ 18:00	总结和答疑



Kibana使用

Kibana使用

Kibana

批量导入数据

数据批量查询

map 映射

Kibana 部分

Kibana修改时间

Kibana数据展示

Kibana展示方式

Kibana



批量导入数据

- 使用_bulk批量导入数据
 - 批量导入数据使用POST方式，数据格式为json，url编码使用data-binary
 - 导入含有index配置的json文件


```
# gzip -d logs.jsonl.gz
# curl -XPOST 'http://192.168.4.14:9200/_bulk' --data-binary @logs.jsonl

# gzip -d shakespeare.json.gz
# curl -XPOST 'http://192.168.4.14:9200/_bulk' --data-binary @shakespeare.json
```



批量导入数据（续1）

- 使用_bulk批量导入数据
 - 导入没有index配置的json文件
 - 我们需要在ur里面制定index和type

```
# gzip -d accounts.json.gz
```

```
# curl -XPOST
```

```
'http://192.168.4.14:9200/accounts/act/_bulk?pretty' --data-binary @accounts.json
```



数据批量查询

- 数据批量查询使用GET

```
# curl -XGET 'http://192.168.4.11:9200/_mget?pretty' -d '{
  "docs":[
    {
      "_index": "accounts",
      "_type": "act",
      "_id": 1
    },
    {
      "_index": "accounts",
      "_type": "act",
      "_id": 2
    },
  ],
}
```



数据批量查询（续1）

- 数据批量查询使用GET
 - 续上一页

```
{  
  "_index": "shakespeare",  
  "_type": "scene",  
  "_id": 1  
}  
]  
'
```



map 映射

- mapping :
 - 映射：创建索引的时候，可以预先定义字段的类型及相关属性
 - 作用：这样会让索引建立得更加的细致和完善
 - 分类：静态映射和动态映射
 - 动态映射：自动根据数据进行相应的映射
 - 静态映射：自定义字段映射数据类型



案例1：导入数据

1. 批量导入数据并查看



Kibana 部分

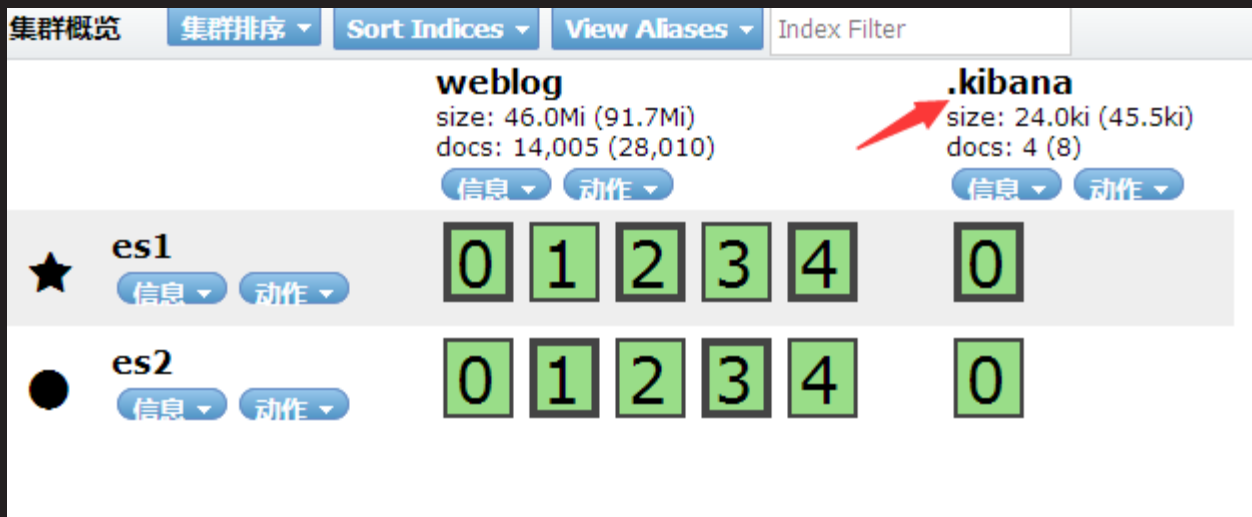
- 数据导入以后查看logs是否导入成功

logstash-2015.05.20	logstash-2015.05.19	logstash-2015.05.18
size: 27.3Mi (57.7Mi) docs: 4,750 (9,500)	size: 25.4Mi (58.6Mi) docs: 4,624 (9,248)	size: 26.8Mi (58.5Mi) docs: 4,631 (9,262)
信息 动作	信息 动作	信息 动作
<div>0 1 2 3</div> <div>4</div>	<div>0 1 2 3</div> <div>4</div>	<div>0 1 2 3</div> <div>4</div>
<div>0 1 2 3</div> <div>4</div>	<div>0 1 2 3</div> <div>4</div>	<div>0 1 2 3</div> <div>4</div>



Kibana 部分 (续1)

- 修改Kibana的配置文件后启动Kibana，然后查看ES集群，如果出现.kibana Index表示Kibana与ES集群连接成功



Kibana 部分 (续2)

- Kibana里选择日志
 - 支持通配符 *
 - 我们这里选择logstash-*
 - 在下面的Time-field选择@timestramp作为索引
 - 然后点create按钮



Kibana 部分 (续3)

Configure an index pattern

In order to use Kibana you must configure at least one index pattern. Index patterns are used to identify the Elasticsearch index to run search and analytics against. They are also used to configure fields.

☒ Index contains time-based events

☐ Use event times to create index names [DEPRECATED]

Index name or pattern

Patterns allow you to define dynamic index names using * as a wildcard. Example: logstash-*

logstash-*

☐ Do not expand index pattern when searching (Not recommended)

By default, searches against any time-based index pattern that contains a wildcard will automatically be expanded to query only the indices that contain data within the currently selected time range.

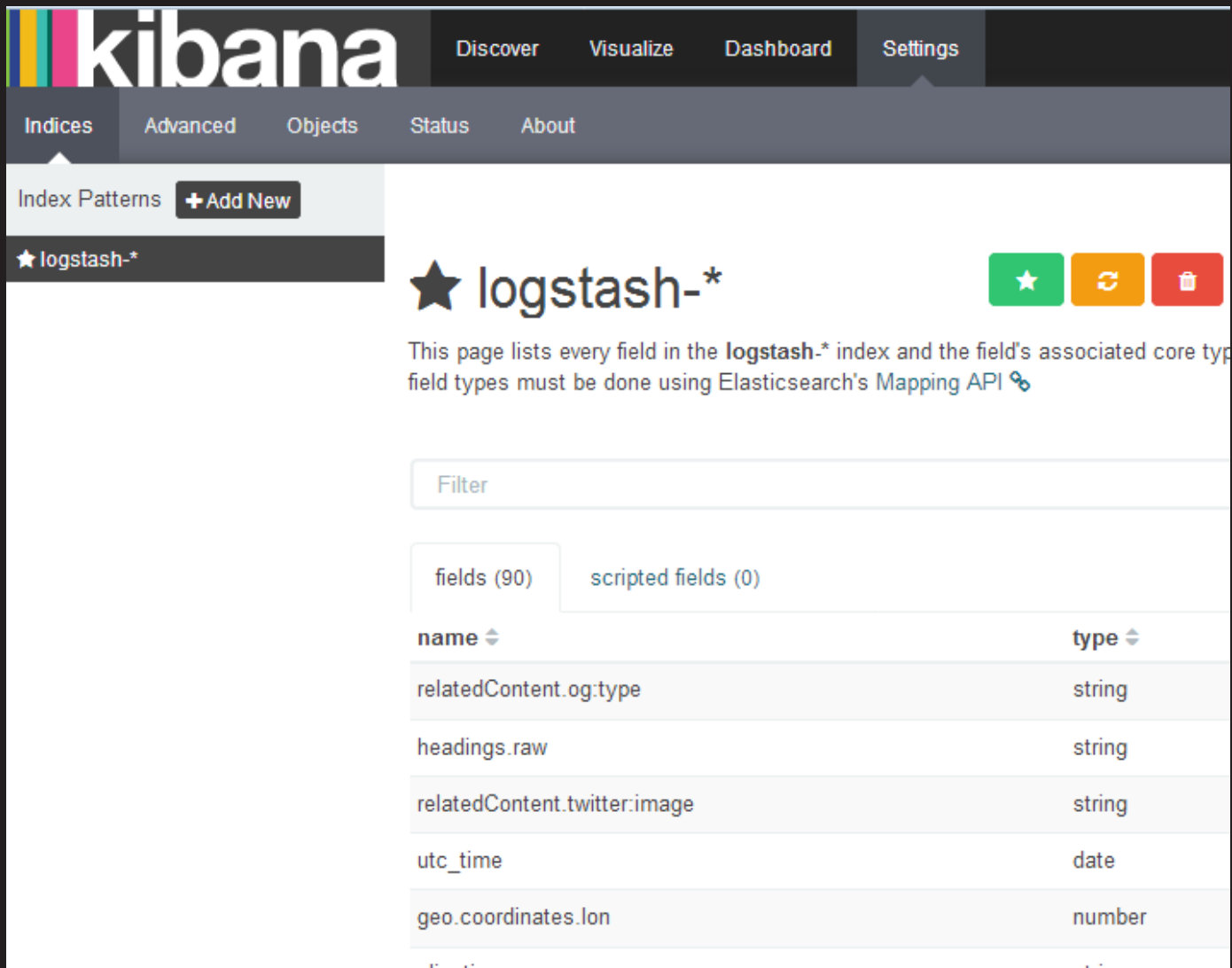
Searching against the index pattern *logstash-** will actually query elasticsearch for the specific matching indices (e.g. *logstash-2015.12.21*) that fall within the current time range.

Time-field name ⓘ refresh fields

Create



Kibana 部分 (续4)



kibana Discover Visualize Dashboard Settings

Indices Advanced Objects Status About

Index Patterns **+ Add New**

★ logstash-*

★ logstash-*

This page lists every field in the **logstash-*** index and the field's associated core type. Field types must be done using Elasticsearch's [Mapping API](#).

Filter

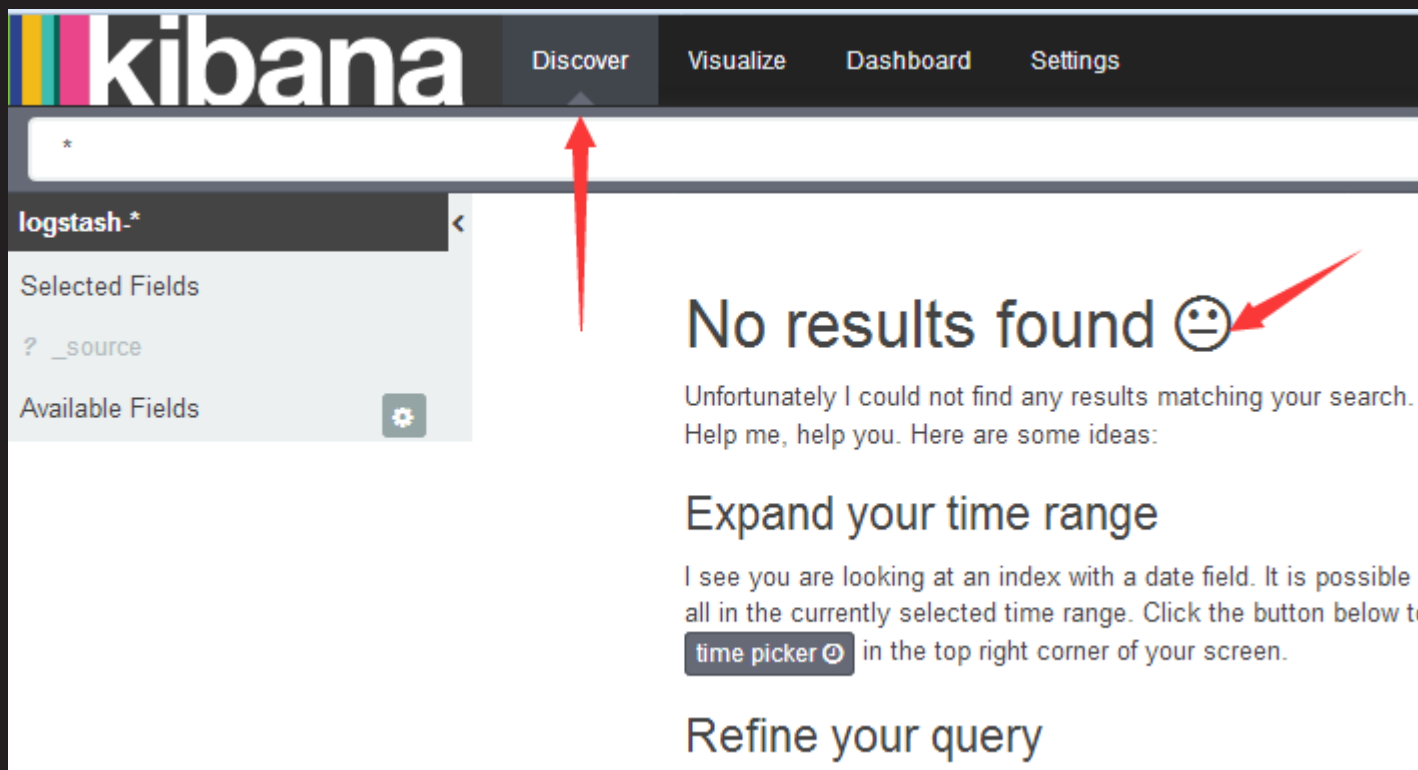
fields (90) scripted fields (0)

name	type
relatedContent.og:type	string
headings.raw	string
relatedContent.twitter:image	string
utc_time	date
geo.coordinates.lon	number
clientip	string



Kibana 部分 (续5)

- 导入成功以后选择Discover



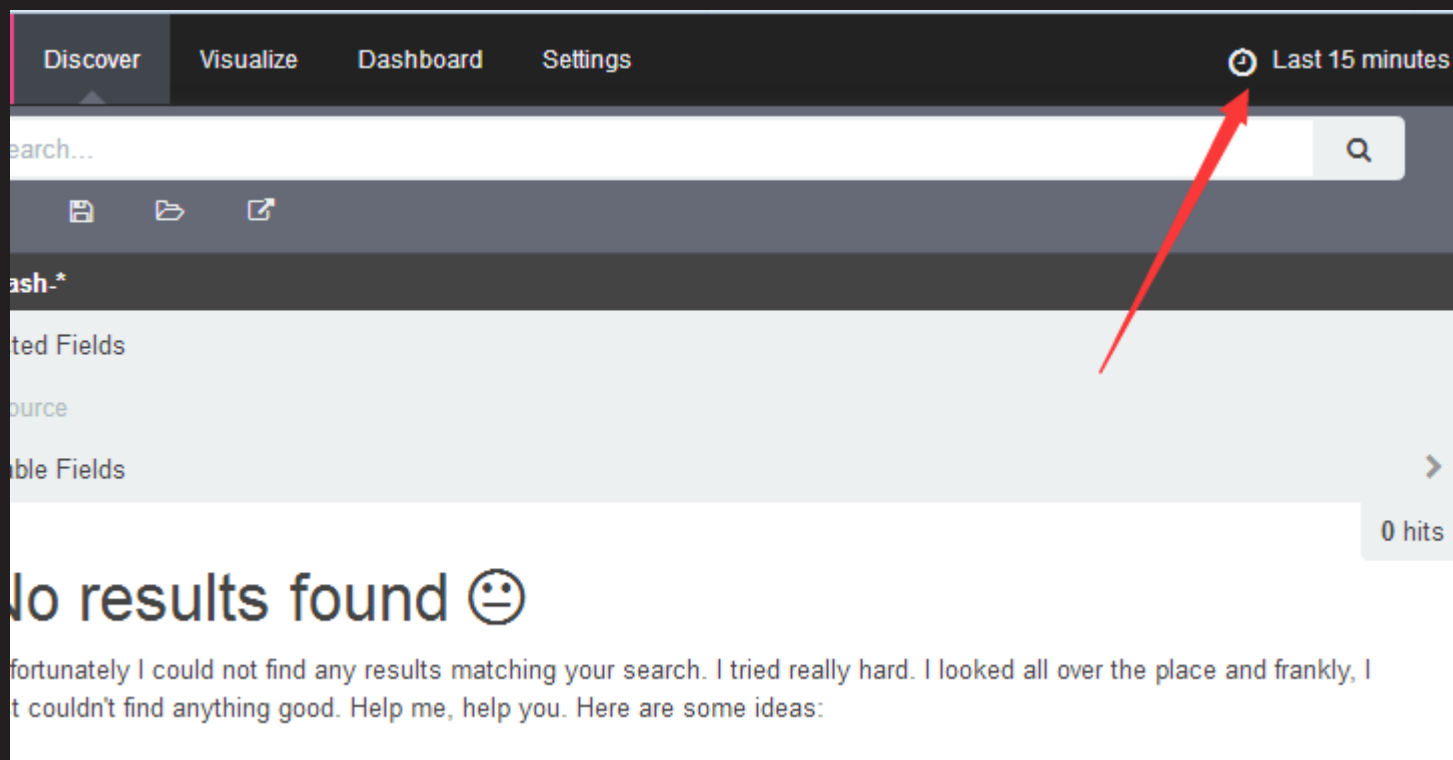
Kibana 部分（续6）

- 这里没有数据的原因是我们导入的日志是2015-05-10至2015-05-20的时间段，默认配置是最近15分钟，在这可以修改一下时间来显示



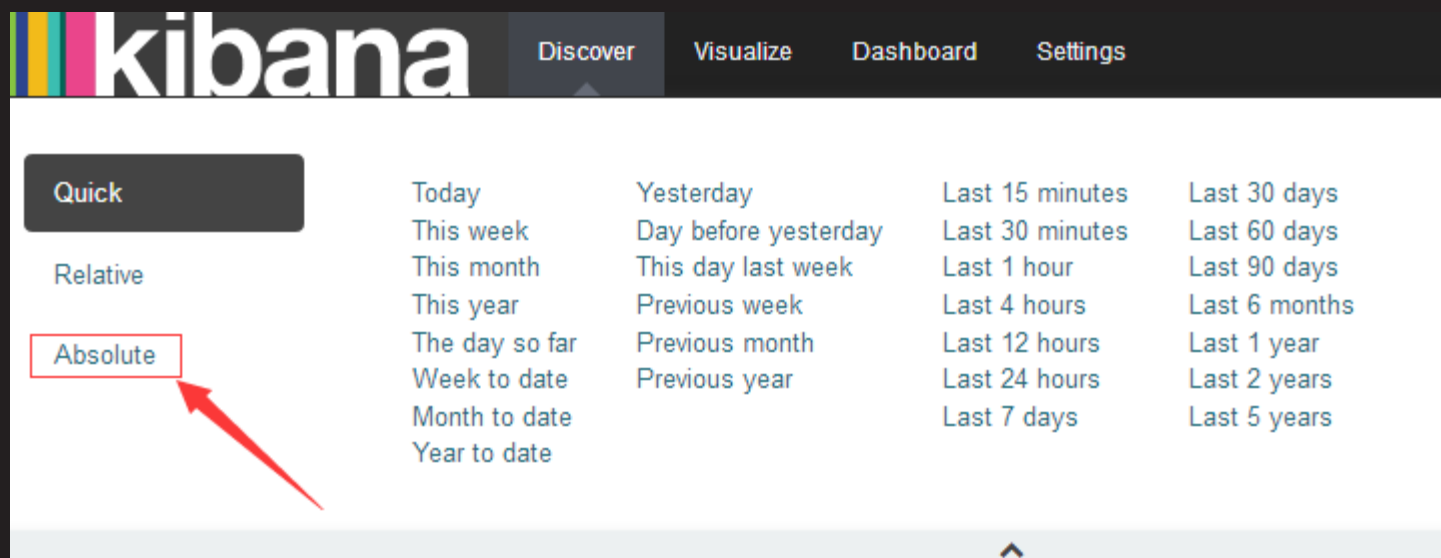
Kibana修改时间

- 修改时间



Kibana修改时间（续1）

- 修改时间



Kibana修改时间（续2）

- 修改时间

From:

YYYY-MM-DD HH:mm:ss.SSS

<

May 2015

>

Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	01	02
03	04	05	06	07	08	09
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	01	02	03	04	05	06

To: Set To Now

YYYY-MM-DD HH:mm:ss.SSS

<

May 2015

>

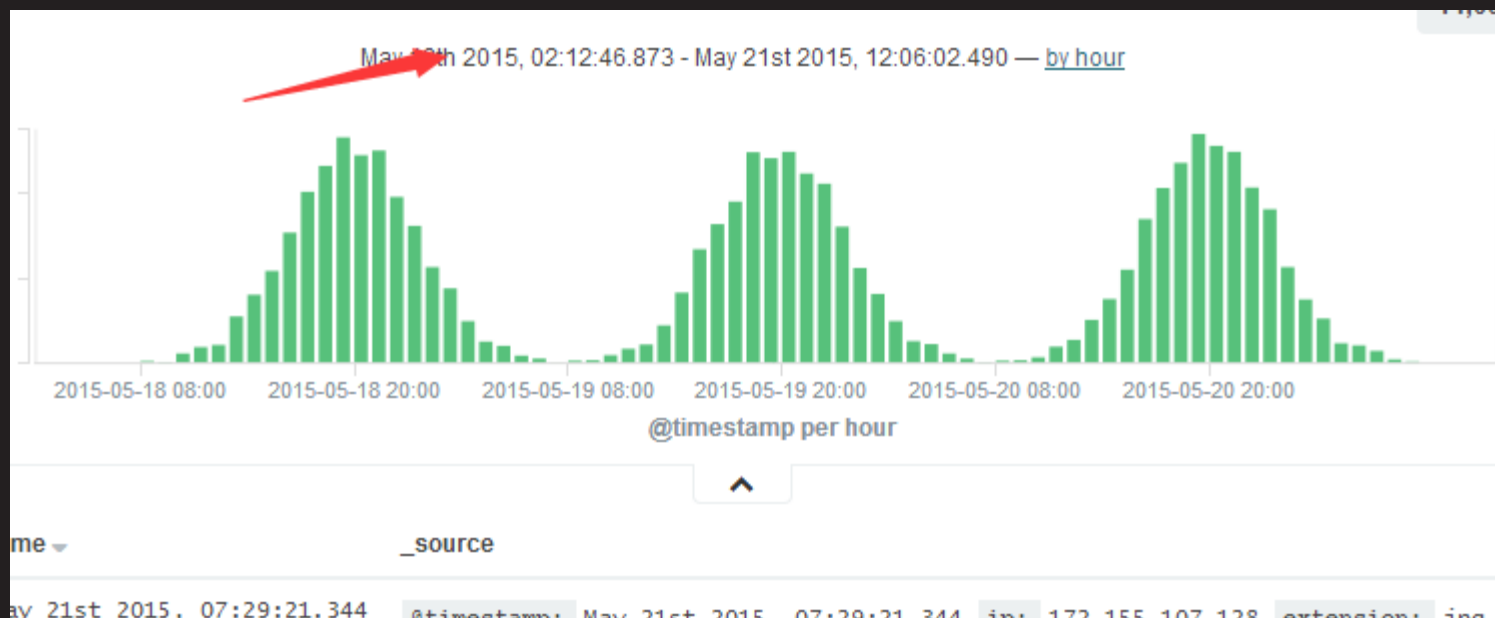
Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	01	02
03	04	05	06	07	08	09
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	01	02	03	04	05	06

Go











Kibana数据展示

- 数据展示



Kibana展示方式

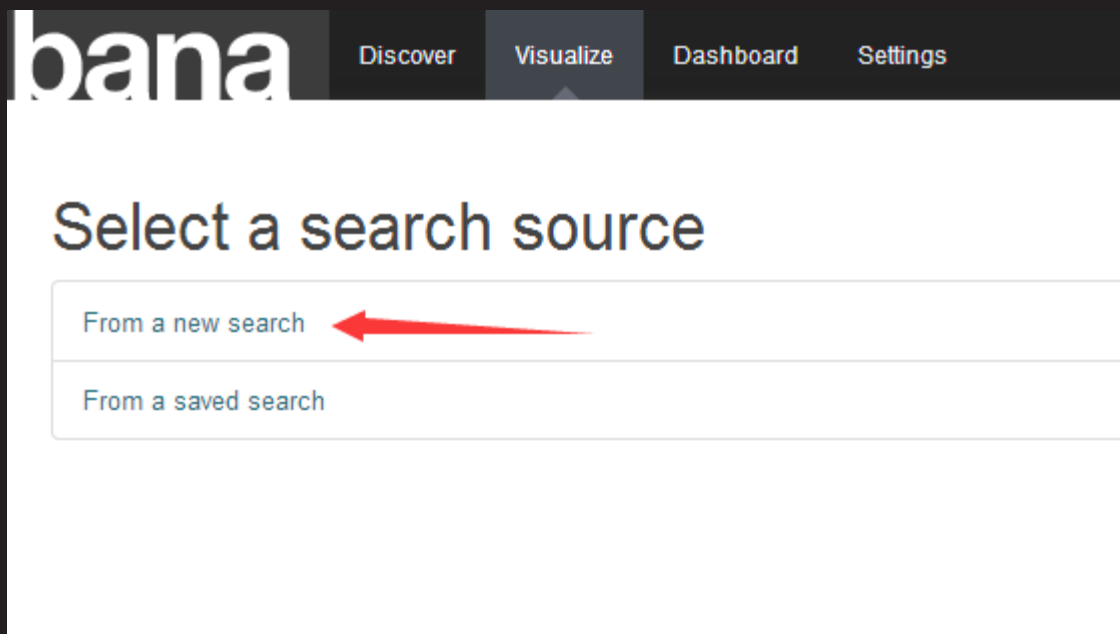
- 除了柱状图，Kibana还支持很多种展示方式

	Area chart	Great for stacked timelines in which the total of all series is more important than comparing any two or more series. Less useful for assessing the relative change of unrelated data points as changes in a series lower down the stack will have a difficult to gauge effect on the series above it.
	Data table	The data table provides a detailed breakdown, in tabular format, of the results of a composed aggregation. Tip, a data table is available from many other charts by clicking grey bar at the bottom of the chart.
	Line chart	Often the best chart for high density time series. Great for comparing one series to another. Be careful with sparse sets as the connection between points can be misleading.
	Markdown widget	Useful for displaying explanations or instructions for dashboards.
	Metric	One big number for all of your one big number needs. Perfect for showing a count of hits, or the exact average a numeric field.
	Pie chart	Pie charts are ideal for displaying the parts of some whole. For example, sales percentages by department.Pro Tip: Pie charts are best used sparingly, and with no more than 7 slices per pie.
	Tile map	Your source for geographic maps. Requires an elasticsearch geo_point field. More specifically, a field that is mapped as type:geo_point with latitude and longitude coordinates.
	Vertical bar chart	The goto chart for oh-so-many needs. Great for time and non-time data. Stacked or grouped, exact numbers or percentages. If you are not sure which chart you need, you could do worse than to start here.



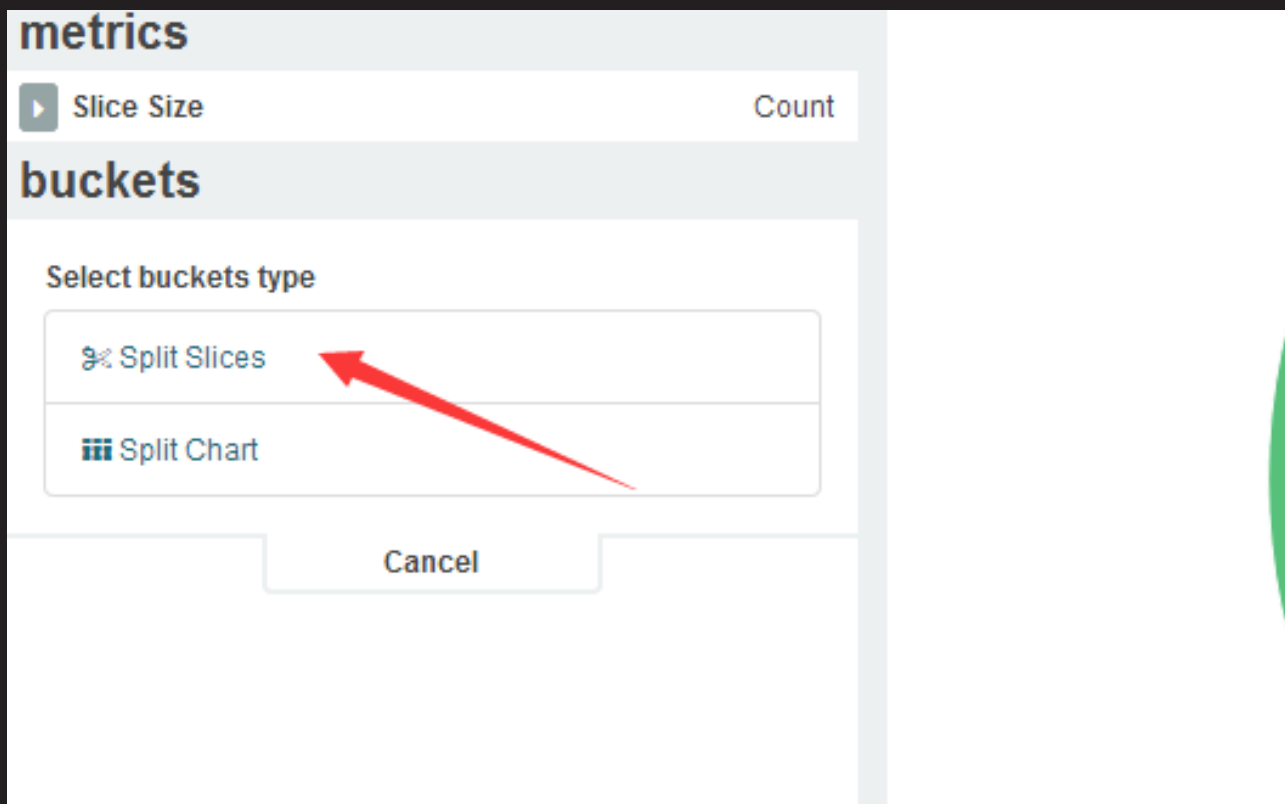
Kibana展示方式（续1）

- 饼图与列表，多种维度自定义统计分析



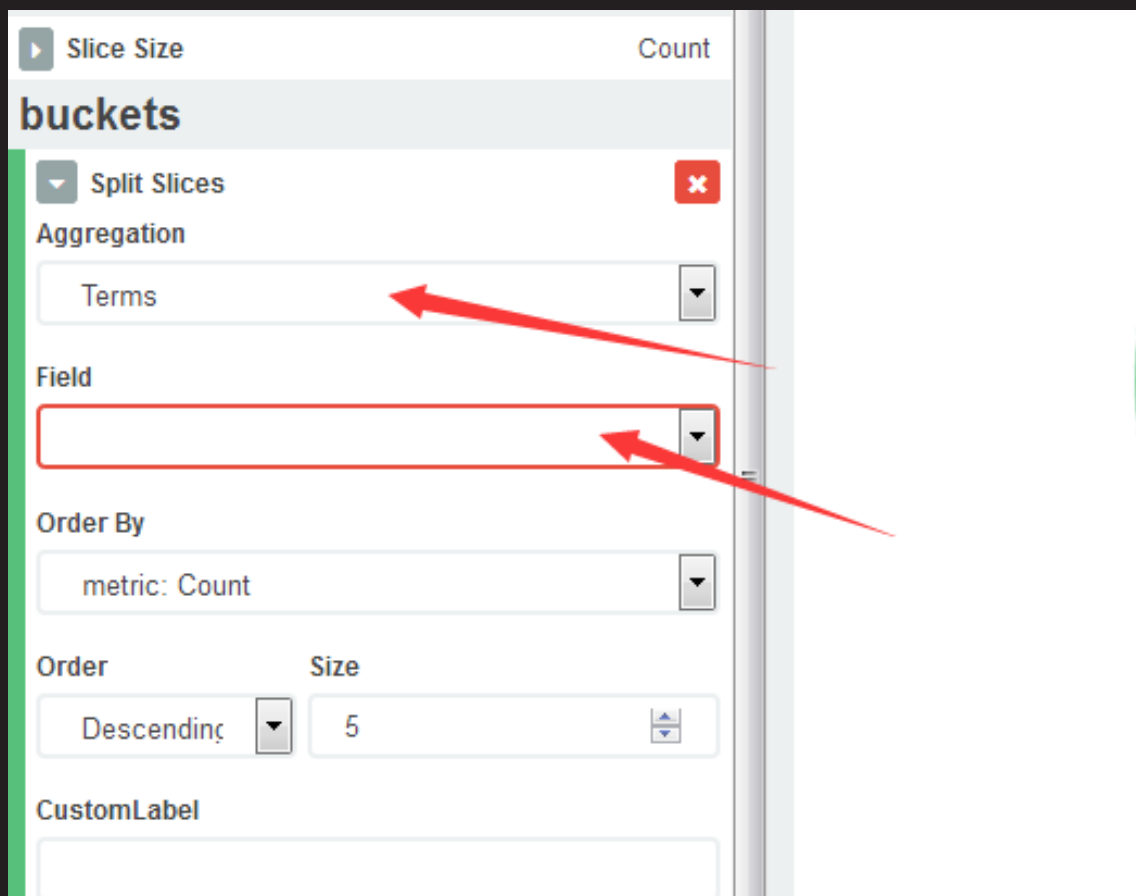
Kibana展示方式（续2）

- 饼图与列表，多种维度自定义统计分析



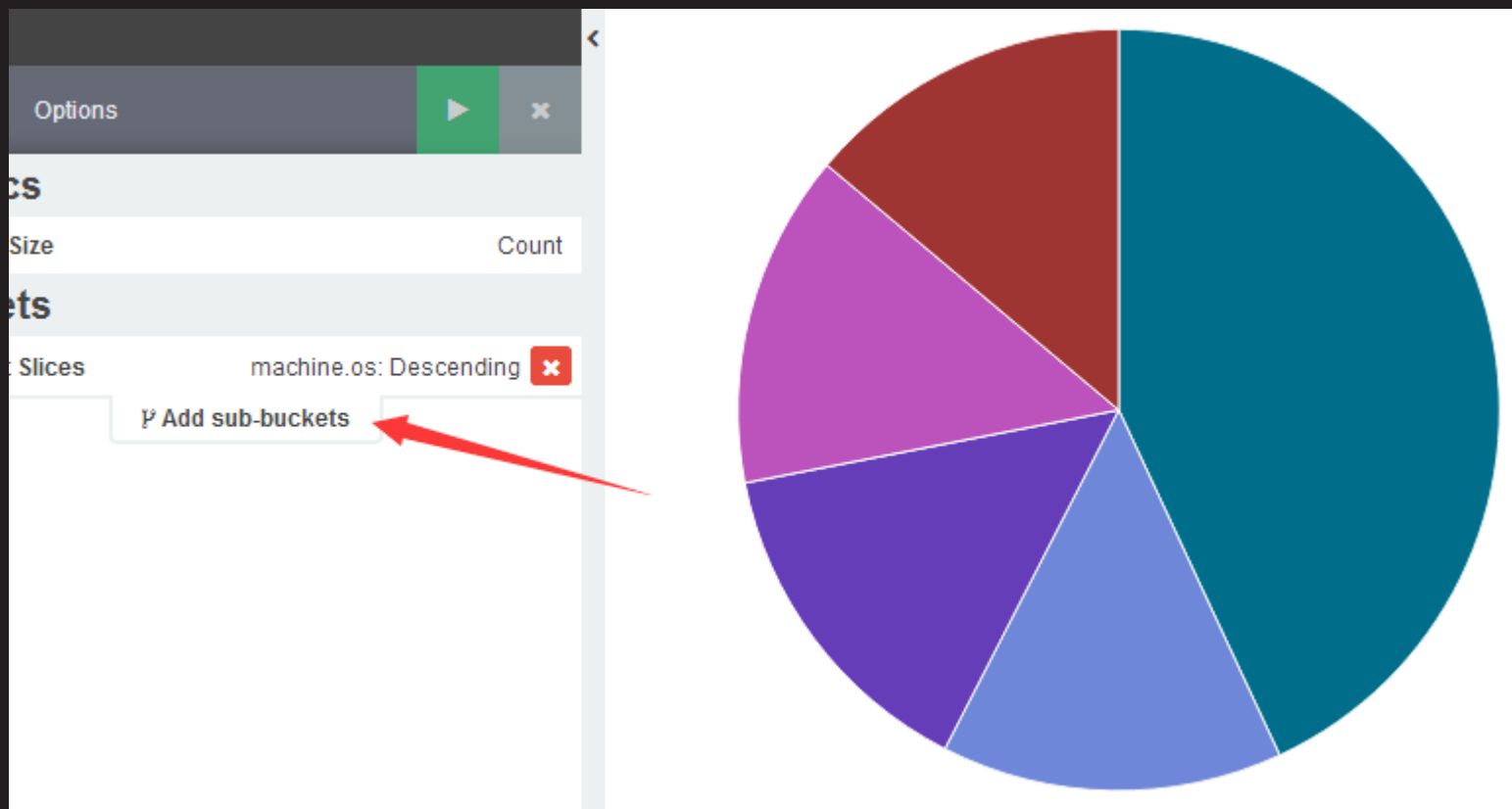
Kibana展示方式（续3）

- 饼图与列表，多种维度自定义统计分析



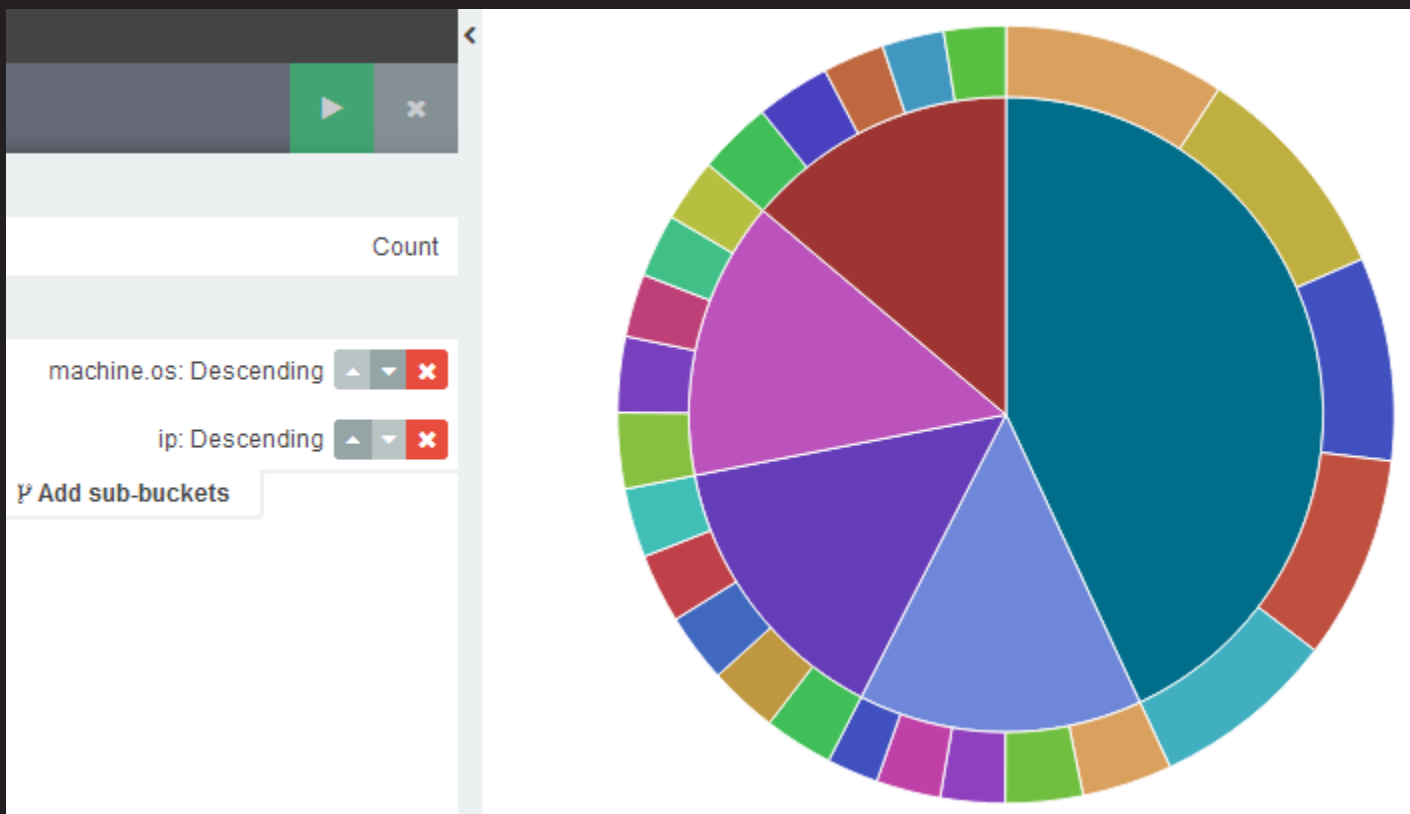
Kibana展示方式（续4）

- 饼图与列表，多种维度自定义统计分析



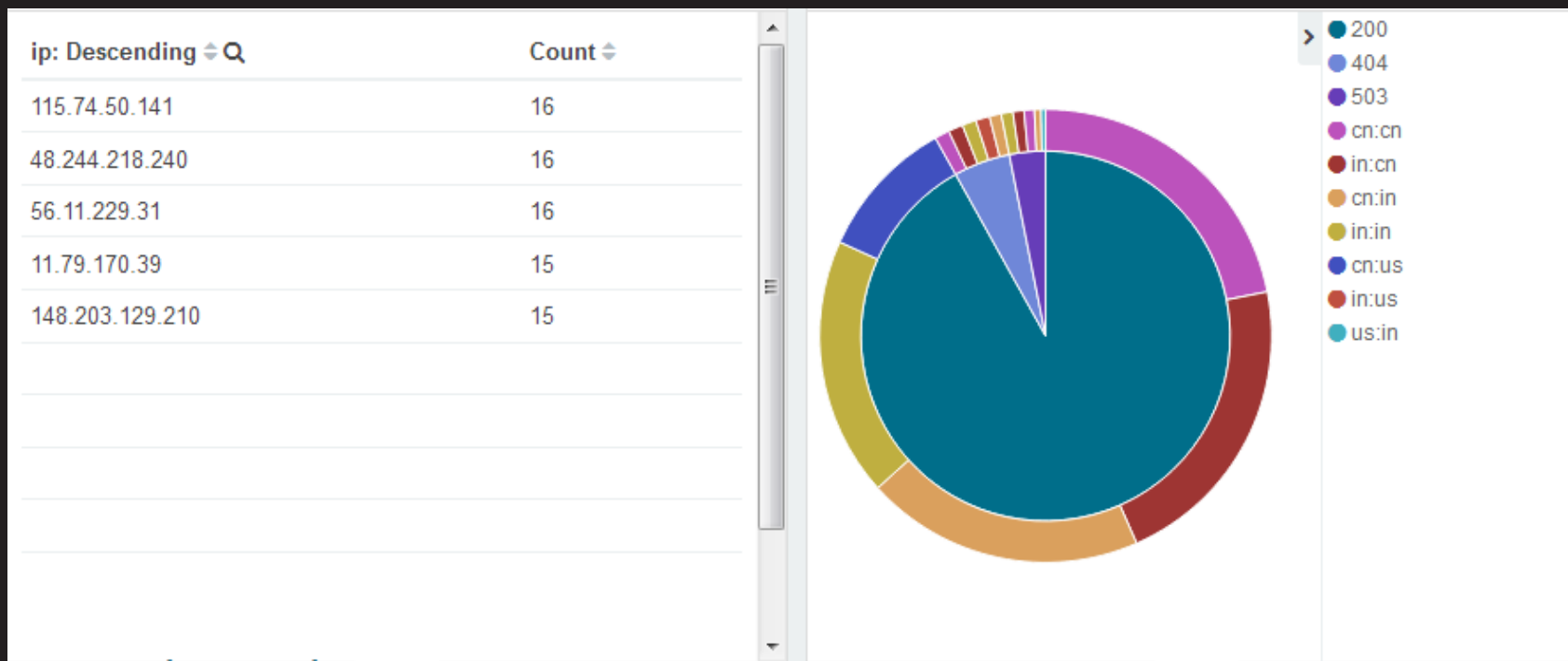
Kibana展示方式（续5）

- 饼图与列表，多种维度自定义统计分析



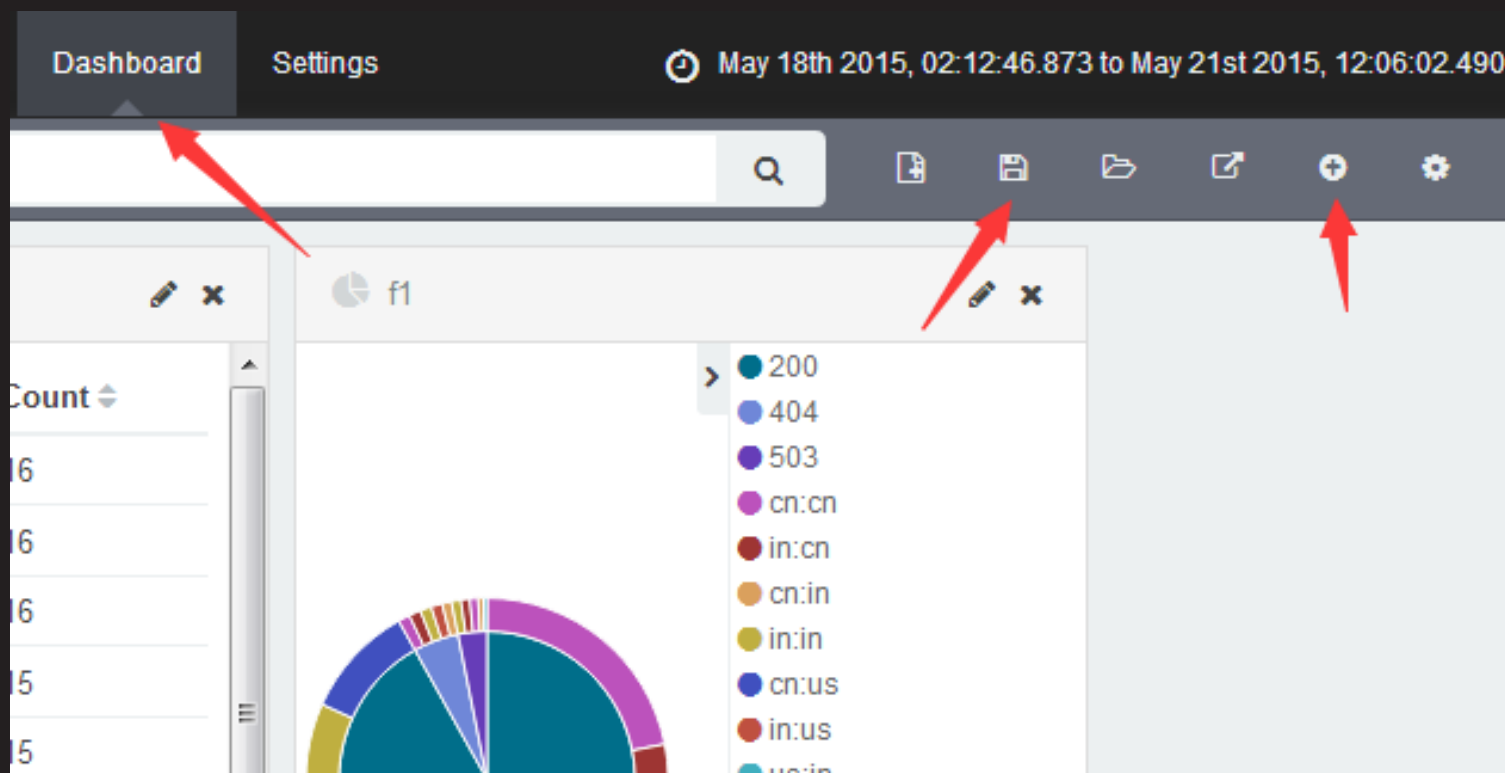
Kibana展示方式（续6）

- 列表与饼图



Kibana展示方式（续7）

- 保存后可以在Dashboard查看



Logstash配置扩展插件

Logstash配置扩展插件

Logstash

Logstash是什么

Logstash安装

Logstash类型及条件判断

Logstash配置文件

插件

filebeat安装配置

Logstash

Logstash是什么

- 是一个数据采集、加工处理以及传输的工具
- 特点
 - 所有类型的数据集中处理
 - 不同模式和格式数据的正常化
 - 自定义日志格式的迅速扩展
 - 为自定义数据源轻松添加插件



Logstash安装

- Logstash安装
 - Logstash依赖Java环境，需要安装java-1.8.0-openjdk
 - Logstash没有默认的配置文件，需要手动配置
 - Logstash安装在/opt/logstash目录下
`# rpm -ivh logstash-2.3.4-1.noarch.rpm`



Logstash安装（续1）

- Logstash工作结构

{ 数据源 } ==>

input { } ==>

filter { } ==>

output { } ==>

{ ES }



Logstash类型及条件判断

- Logstash里面的类型
 - 布尔值类型: `ssl_enable => true`
 - 字节类型: `bytes => "1MiB"`
 - 字符串类型: `name => "xkops"`
 - 数值类型: `port => 22`
 - 数组: `match => ["datetime","UNIX"]`
 - 哈希: `options => {k => "v",k2 => "v2"}`
 - 编码解码: `codec => "json"`
 - 路径: `file_path => "/tmp/filename"`
 - 注释: `#`



Logstash类型及条件判断（续1）

- Logstash条件判断

- 等于: ==
- 不等于: !=
- 小于: <
- 大于: >
- 小于等于: <=
- 大于等于: >=
- 匹配正则: =~
- 不匹配正则: !~



Logstash类型及条件判断（续2）

- Logstash条件判断
 - 包含: in
 - 不包含: not in
 - 与: and
 - 或: or
 - 非与: nand
 - 非或: xor
 - 复合表达式: ()
 - 取反符合: !()



Logstash配置文件

- Logstash的第一个配置文件
 - /etc/logstash/logstash.conf

```
input{  
  stdin}  
}  
filter{ }  
output{  
  stdout}  
}
```

- 启动并验证

```
# logstash -f logstash.conf
```



插件

- Logstash插件
 - 上页的配置文件使用了logstash-input-stdin和logstash-output-stdout两个插件，Logstash还有filter和codec类插件，查看插件的方式是


```
# /opt/logstash/bin/logstash-plugin list
```
 - 插件及文档地址

<https://github.com/logstash-plugins>
- 练习
 - Logstash配置从标准输入读取输入源，然后将结果输出到屏幕



插件（续1）

- codec类插件
 - 常用的插件：plain、json、json_lines、rubydebug、multiline等
 - 使用刚刚的例子，这次输入json数据
 - 设置输入源的codec是json，在输入的时候选择rubydebug



插件（续2）

- codec类插件

```
input{
  stdin{ codec => "json" }
}
filter{ }
output{
  stdout{ codec => "rubydebug" }
}
```

- 输入普通数据和json对比

```
{"a": 1, "c": 3, "b": 2}
```



插件（续3）

- codec类插件
 - 练习output和input配置
 - 练习在input不指定类型json输出结果
 - 练习在output不指定rubydebug的输出结果
 - 同时指定以后的输出结果



插件（续4）

- 练习input file插件

```
file{
    start_position => "beginning"
    sincedb_path => "/var/lib/logstash/sincedb-access"
    path => [ "/tmp/alog" , "/tmp/blog" ]
    type => 'filelog'
}
```

- sincedb_path记录读取文件的位置
- start_position配置第一次读取文件从什么地方开始



插件（续5）

- 练习input tcp和udp插件

```
tcp{
    host => "0.0.0.0"
    port => 8888
    type => "tcplog"
}
udp{
    host => "192.168.4.16"
    port => 9999
    type => "udplog"
}
```



插件（续6）

- tcp&udp练习

- 使用shell脚本，对tcp指定端口发送数据

```
function sendmsg(){
    if (( $# == 4 )) && [ $1 == "tcp" -o $1 == "udp" ];then
        exec 9<>/dev/$1/$2/$3
        echo "$4" >&9
        exec 9<&-
    else
        echo "$0 (tcp|udp) ipaddr port msg"
    fi
}
```



插件（续7）

- tcp&udp 练习

- 发送tcp数据

- ```
sendmsg tcp 192.168.4.10 8888 'tcp msg'
```

- 发送udp数据

- ```
sendmsg udp 192.168.4.10 9999 'udp msg'
```



插件（续8）

- syslog插件练习

```
syslog{
    host => "192.168.4.10"
    port => 514
    type => "syslog"
}
```

- rsyslog.conf配置向远程发送数据

```
local0.info                @@192.168.4.10:514
```

- 写syslog，查看状态

```
logger -p local0.info -t test_logstash 'test message'
```



插件（续9）

- filter grok插件
 - 解析各种非结构化的日志数据插件
 - grok使用正则表达式把非结构化的数据结构化
 - 在分组匹配，正则表达式需要根据具体数据结构编写
 - 虽然编写困难，但适用性极广
 - 几乎可以应用于各类数据

```
grok{
    match => ["message", "%{IP:ip} , (?<key>reg) " ]
}
```



插件（续10）

- grok正则分组匹配

- 匹配ip时间戳和请求方法

```
"(?<ip>(\d+\.){3}\d+) \S+ \S+
(?<time>.*\])\s+\"(?<method>[A-Z]+)"
```

- 使用正则宏

```
%{IPORHOST:clientip} %{HTTPDUSER:ident} %{USER:auth}
\[%{HTTPDATE:timestamp}\] \"%{WORD:verb}
```

- 最终版本

```
%{COMMONAPACHELOG} \"(?<referer>[^\"]+)\\"
\"(?<UA>[^\"]+)\\"
```



插件（续11）

- output ES插件

```
if [type] == "filelog"{
  elasticsearch {
    hosts => ["192.168.4.15:9200"]
    index => "weblog"
    flush_size => 2000
    idle_flush_time => 10
  }
}
```

— 调试成功后，把数据写入ES集群



插件（续12）

- input filebeats插件

```
beats {  
  port => 5044  
  codec => "json"  
}
```

- 这个插件主要用来接收beats类软件发送过来的数据，由于logstash依赖JAVA环境，而且占用资源非常大，因此会使用更轻量的filebeat替代



filebeat安装配置

- filebeat安装与配置
 - 使用rpm安装filebeat

```
# rpm -ivh filebeat-1.2.3-x86_64.rpm
```
 - 修改配置文件/etc/filebeat/filebeat.yml
 - 设置开机运行

```
# systemctl enable filebeat
```
 - 开启服务

```
# systemctl start filebeat
```



filebeat安装配置（续1）

- 修改配置文件/etc/filebeat/filebeat.yml

```
paths:
  - /root/logs.jsonl
  document_type: weblog
... ..
paths:
  - /root/accounts.json
  document_type: account
output:
  logstash:
    hosts: ["192.168.4.10:5044"]
```



案例2：综合练习

1. 练习插件
2. 安装一台Apache服务并配置
3. 使用filebeat收集Apache服务器的日志
4. 使用grok处理filebeat发送过来的日志
5. 存入elasticsearch



总结和答疑
