# 安装[Elasticsearch](https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html)

yum install java

YUM

Download and install the public signing key:

rpm --import https://packages.elastic.co/GPG-KEY-elasticsearch

Add the following in your /etc/yum.repos.d/elasticsearch.repo directory in a file with a .repo suffix, for example elasticsearch.repo

[elasticsearch-2.x]

name=Elasticsearch repository for 2.x packages

baseurl=https://packages.elastic.co/elasticsearch/2.x/centos

gpgcheck=1

gpgkey=https://packages.elastic.co/GPG-KEY-elasticsearch

enabled=1

And your repository is ready for use. You can install it with:

yum -y install elasticsearch

chkconfig --add elasticsearch

Otherwise if your distribution is using systemd:

sudo /bin/systemctl daemon-reload

sudo /bin/systemctl enable elasticsearch.service

默认端口9200

配置文件/etc/sysconfig/elasticsearch

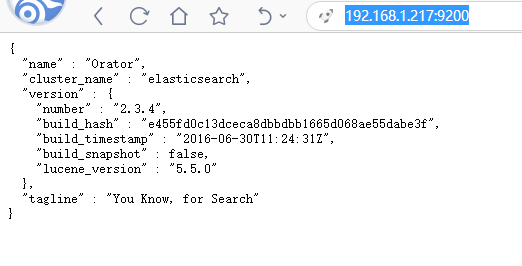
配置文件/etc/elasticsearch/elasticsearch.yml

配置network.host: 10.10.10.210

启动 service elasticsearch start

测试curl -X GET http://10.10.10.210:9200/

或者浏览器访问：

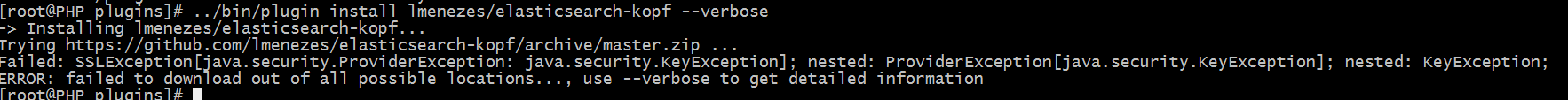


### 安装elasticsearch 插件

先使用plugin install命令安装

/usr/share/elasticsearch/bin/plugin install lmenezes/elasticsearch-kopf

会报如下错误，初步判断是因为访问https的时候ssl的问题，但是我们可以看到此插件的地址，然后手动下载压缩包、手动安装



下载插件

wget <https://github.com/lmenezes/elasticsearch-kopf/archive/master.zip>

安装插件

/usr/share/elasticsearch/bin/plugin install file:/usr/share/elasticsearch/plugins/master.zip

如果你把安装包放在plugins目录下，则安装后需要删除plugins/master.zip 否则elasticsearch启动报错。

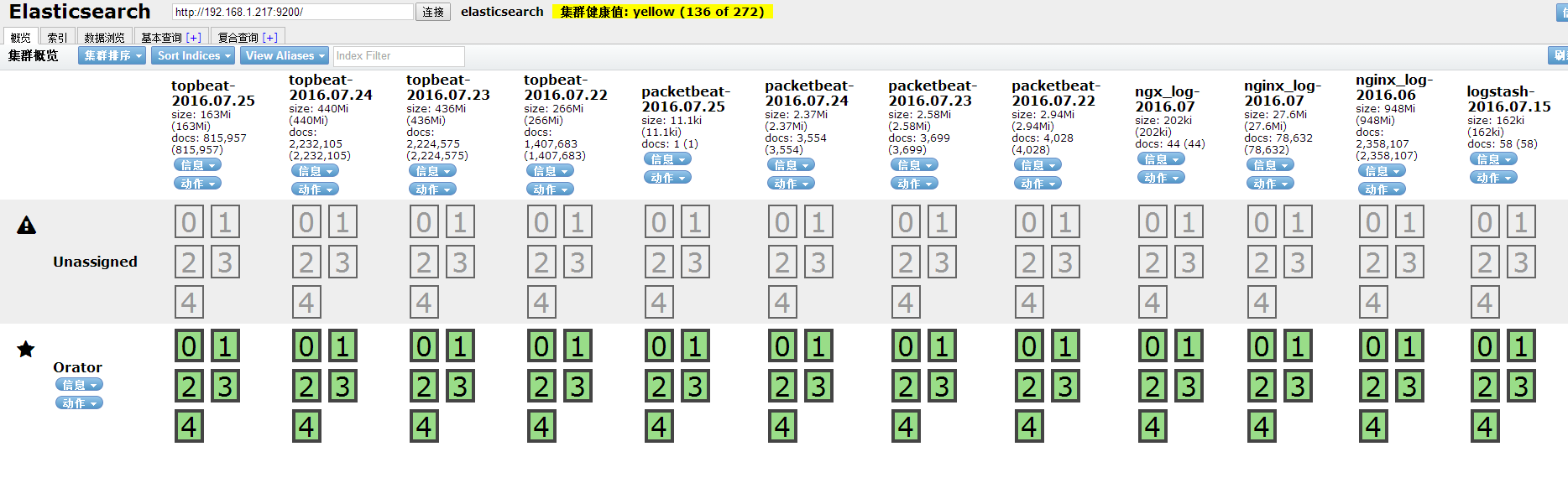
重启

service elasticsearch restart

访问这个插件地址：（不是所有插件都能通过浏览器访问）

<http://192.168.1.217:9200/_plugin/kopf>

head插件



# 安装[Kibana](https://www.elastic.co/guide/en/kibana/4.5/index.html)

Installing Kibana with yum[**edit**](https://github.com/elastic/kibana/edit/4.5/docs/kibana-repositories.asciidoc)

Warning

The repositories set up in this procedure are not compatible with distributions using version 3 of rpm, such as CentOS version 5.

1. Download and install the public signing key:

rpm --import https://packages.elastic.co/GPG-KEY-elasticsearch

1. Create a file named kibana.repo in the /etc/yum.repos.d/kibana.repo directory with the following contents:
2. [kibana-4.5]
3. name=Kibana repository for 4.5.x packages
4. baseurl=http://packages.elastic.co/kibana/4.5/centos
5. gpgcheck=1
6. gpgkey=http://packages.elastic.co/GPG-KEY-elasticsearch

enabled=1

1. Install Kibana by running the following command:

yum install kibana

Configure Kibana to automatically start during bootup. If your distribution is using the System V version of init (check with ps -p 1), run the following command:

chkconfig --add kibana

1. 配置文件

/opt/kibana/config/kibana.yml

配置elasticsearch.url: "http://10.10.10.210:9200"

1. If your distribution is using systemd, run the following commands instead:
2. sudo /bin/systemctl daemon-reload

sudo /bin/systemctl enable kibana.service

service kibana start

That’s it! Kibana is now running on port 5601.

<http://192.168.1.217:5601>

### 安装kibana插件

/opt/kibana/bin/kibana plugin --install elastic/sense

Service kibana restart 安装后重启kibana

# 安装Logstash

Logstash requires Java 7 or later. Use the [official Oracle distribution](http://www.oracle.com/technetwork/java/javase/downloads/index.html) or an open-source distribution such as [OpenJDK](http://openjdk.java.net/).

To check your Java version, run the following command:

java -version

On systems with Java installed, this command produces output similar to the following:

java version "1.7.0\_45"

Java(TM) SE Runtime Environment (build 1.7.0\_45-b18)

Java HotSpot(TM) 64-Bit Server VM (build 24.45-b08, mixed mode)

YUM

Download and install the public signing key:

rpm --import https://packages.elastic.co/GPG-KEY-elasticsearch

Add the following in your /etc/yum.repos.d/logstash.repo directory in a file with a .repo suffix, for example logstash.repo

[logstash-2.3]

name=Logstash repository for 2.3.x packages

baseurl=https://packages.elastic.co/logstash/2.3/centos

gpgcheck=1

gpgkey=https://packages.elastic.co/GPG-KEY-elasticsearch

enabled=1

And your repository is ready for use. You can install it with:

yum install logstash

配置文件/etc/sysconfig/logstash

vim /etc/logstash/conf.d/

默认端口57400

### 安装logstash插件

查看可用插件

bin/logstash-plugin list

## 匹配实例

匹配文件放在/opt/logstash/patterns目录下，然后在配置文件的grok中用patterns\_dir => "/opt/logstash/patterns"

指定。

URIPARM1 [A-Za-z0-9$.+!\*'|(){},~@#%&/=:;\_?\-\[\]]\*

URIPATH1 (?:/[A-Za-z0-9$.+!\*'(){},~:;=@#%&\_\- ]\*)+

URI1 (%{URIPROTO}://)?(?:%{USER}(?::[^@]\*)?@)?(?:%{URIHOST})?(?:%{URIPATHPARAM})?

NGINXACCESSAPI %{IPORHOST:remote\_addr} %{IPORHOST:server\_addr} (%{USERNAME:user}|-) \[%{HTTPDATE:log\_timestamp}\] %{BASE10NUM:http\_status} (?:%{BASE10NUM:body\_bytes\_sent}|-) %{HOSTNAME:http\_host} (%{BASE16FLOAT:request\_time}|-) \"(?:%{WORD:method} %{NOTSPACE:request}(?: HTTP/%{NUMBER:http\_version})?|-)\" \"(?:%{URI1:http\_referrer}|-)\" (?:%{QUOTEDSTRING:user\_agent}|-) (%{WORD:x\_forword}|-)

## 配置实例

input{

file{

type=>"api\_log"

path=>"/var/log/nginx/access.log"

start\_position =>"beginning"

}

}

filter{

grok{

patterns\_dir => "/opt/logstash/patterns"

match => {

"message" => "%{NGINXACCESSAPI}"

}

overwrite => ["message"]

}

date {

match => ["log\_timestamp","dd/MMM/YYYY:HH:mm:ss Z"]

target => "@timestamp"

}

}

output{

elasticsearch{

hosts=>"192.168.1.217:9200"

index=>"api\_log-%{+YYYY.MM}"

}

}

# Beat

## 安装Packetbeat

在前面两篇文章中记录了使用logstash来收集mysql的慢查询日志，然后通过kibana以web的方式展示出来，但在生产环境中，需求会更复杂一些，而且通过logstash写正则，实在是个费时费劲的事。例如在生产环境中会有要求分析某个时间段mysql或者mongodb的慢查询日志情况；还有I/O吞吐量；这个时间段内经常执行的查询语句，http访问情况等信息；然后将分析出来的结果以图表的形式展现出来。听起来是不是有点头晕，有点高大上的感觉，其实通过packetbeat，一切将变得简单高效。本文介绍使用packetbeat，elasticsearch,kibana实现这个需求。

### 安装 Packetbeat

To download and install Packetbeat on your application servers, use the commands that work with your system ([deb](https://www.elastic.co/guide/en/beats/packetbeat/current/packetbeat-installation.html#deb) for Debian/Ubuntu, [rpm](https://www.elastic.co/guide/en/beats/packetbeat/current/packetbeat-installation.html#rpm) for Redhat/Centos/Fedora, [mac](https://www.elastic.co/guide/en/beats/packetbeat/current/packetbeat-installation.html#mac) for OS X, and [win](https://www.elastic.co/guide/en/beats/packetbeat/current/packetbeat-installation.html#win) for Windows).

Note

If you use Apt or Yum, you can [install Packetbeat from our repositories](http://www.elastic.co/guide/en/beats/libbeat/1.2/setup-repositories.html) to update to the newest version more easily.

See our [download page](https://www.elastic.co/downloads/beats/packetbeat) for other installation options, such as 32-bit images.

**rpm:**

sudo yum install libpcap

curl -L -O https://download.elastic.co/beats/packetbeat/packetbeat-1.2.3-x86\_64.rpm

sudo rpm -vi packetbeat-1.2.3-x86\_64.rpm

### 配置输出到elasticsearch

配置文件/etc/packetbeat/packetbeat.yml

Set the IP address and port where Packetbeat can find the Elasticsearch installation:

# Configure what outputs to use when sending the data collected by the beat.

# Multiple outputs may be used.

output:

### Elasticsearch as output

elasticsearch:

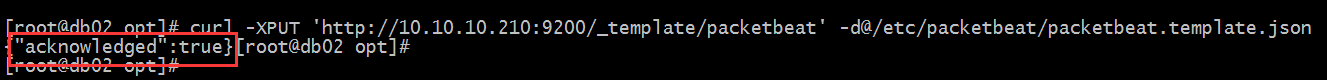
# Array of hosts to connect to.

hosts: ["192.168.1.42:9200"]

### 加载模板

curl -XPUT 'http://localhost:9200/\_template/packetbeat' [-d@/etc/packetbeat/packetbeat.template.json](mailto:-d@/etc/packetbeat/packetbeat.template.json)

返回{"acknowledged":true}证明加载成功



四：启动packetbeat服务

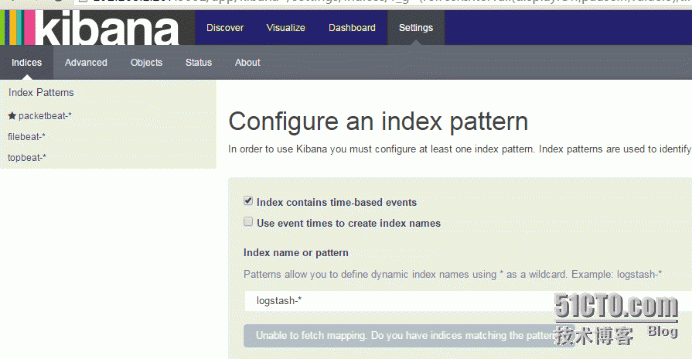
|  |  |
| --- | --- |
| 1 | # /etc/init.d/packetbeat start |

五：导入packetbeat-dashboards

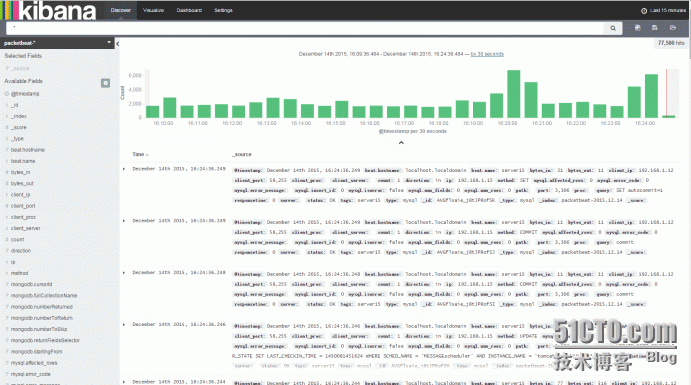
|  |  |
| --- | --- |
| 1  2  3 | # git clone https://github.com/elastic/packetbeat-dashboards  # cd packetbeat-dashboards  # sh load.sh -url http://192.168.1.217:9200 |

六：web展示

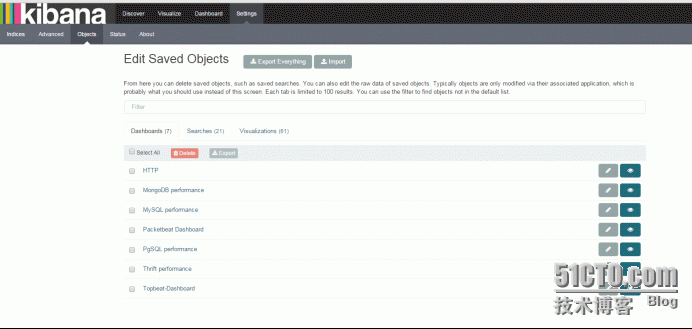
1: 配置索引，这个在执行完load.sh脚本之后，索引会自动创建

[](http://s3.51cto.com/wyfs02/M01/77/CD/wKiom1ZuiQ_wSfvGAACffDA-MDM552.png)

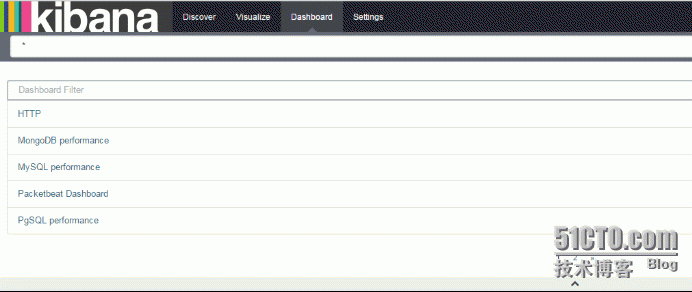
2: 查看客户端的数据推送情况

[](http://s1.51cto.com/wyfs02/M02/77/CE/wKiom1ZuiTnDruz2AADq8GDSF-E663.png)

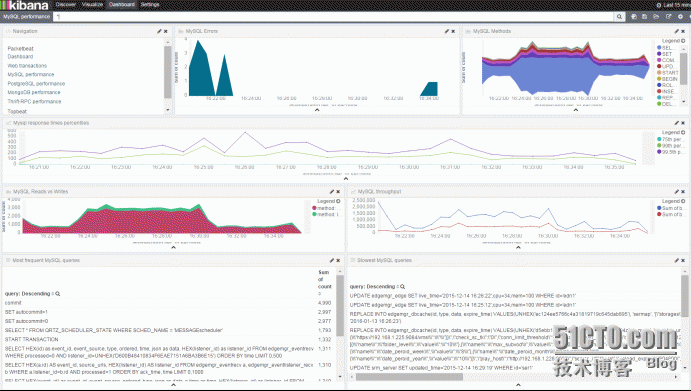
3: 查看导入的面板，可视化视图，点击setting-objects

[](http://s2.51cto.com/wyfs02/M02/77/CD/wKioL1ZuiW7jT1nlAABr3Brl0hc212.png)

4: 图形展示,点击dashboard-load save dashboards

[](http://s4.51cto.com/wyfs02/M00/77/CD/wKioL1ZuiZySBukgAABD8K7ZGHQ261.png)

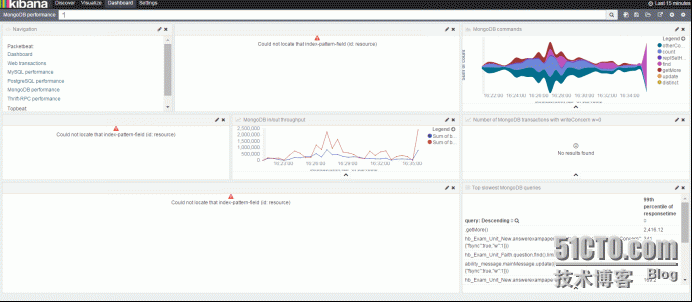
Mysql情况：

[](http://s1.51cto.com/wyfs02/M02/77/CE/wKiom1Zuib3B0N8GAAC-SiDsHqY275.png)

在有多台mysql服务的情况下，可以根据tags来区分，在搜索框中输入相应的tag，则只显示对应的数据

[](http://s4.51cto.com/wyfs02/M01/77/CE/wKiom1ZuieKx5OKOAADLhj9NrP8791.png)

Mongodb情况

[](http://s4.51cto.com/wyfs02/M00/77/CE/wKiom1ZuigDgzATgAABt0o2Rnfw187.png)

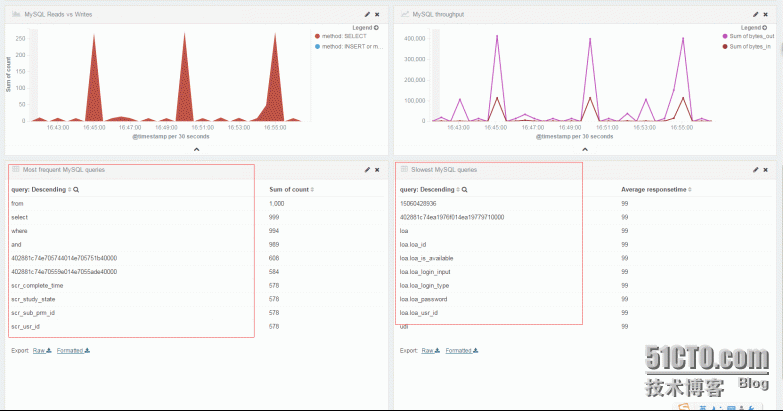
汇总情况：

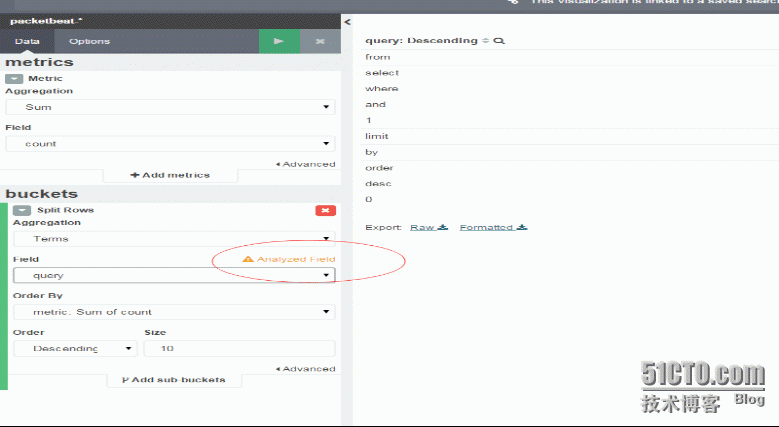
[](http://s3.51cto.com/wyfs02/M01/77/CE/wKiom1ZuihuCI8WaAACjCa0H8Ww300.png)

更多数据演示请访问packetbeat demo网址：<http://demo.elastic.co/packetbeat/>

七：故障排错

1: 在测试过程中曾经发现mysql里面的most frequent Mysql queries和slowest mysql queries数据显示不全，像是被截断的样子，排查后发现其实是模板的问题，删除模板后重新导入即可.

[](http://s2.51cto.com/wyfs02/M02/77/CD/wKioL1ZuilehmdvhAACK_2SQDeI261.png)

[](http://s4.51cto.com/wyfs02/M02/77/CE/wKiom1ZuilLyFRN8AABmQsQvIVk594.png)

|  |  |
| --- | --- |
| 1  2  3  4  5 | # curl -XDELETE 'http://192.168.1.207:9200/\*'   # curl -XPUT   'http://192.168.1.207:9200/\_template/packetbeat' -d@/etc/packetbeat/packetbeat.template.json  # cd packetbeat-dashboards  # sh load.sh -url http://192.168.1.207:9200 |

2: elasticsearch数据维护

搜索数据：(如果你有多个索引，可以把packetbeat-\*换成对应的索引名)：

|  |  |
| --- | --- |
| 1 | # curl -XGET 'http://192.168.1.226:9200/packetbeat-\*/\_search?pretty' |

删除数据(如果你有多个索引，可以把packetbeat-\*换成对应的索引名)：

|  |  |
| --- | --- |
| 1 | # curl -XDELETE 'http://192.168.1.207:9200/packetbeat-\*' |

## 安装TopBeat

是一个简单的服务器数据采集脚本，已经获取服务器进程的CPU，内存，磁盘使用数据。并将数据输出到ElasticSearch，Redis等服务中，非常的简单易用。

1.下载TopBeat  
    https://download.elastic.co/beats/topbeat/topbeat-1.2.3-x86\_64.rpm

2.解压并修改配置文件

    修改topbeat.yml,中的localhost，改成ES主节点IP 192.168.1.217

input:

period:10 此处可以调整数据采集频率

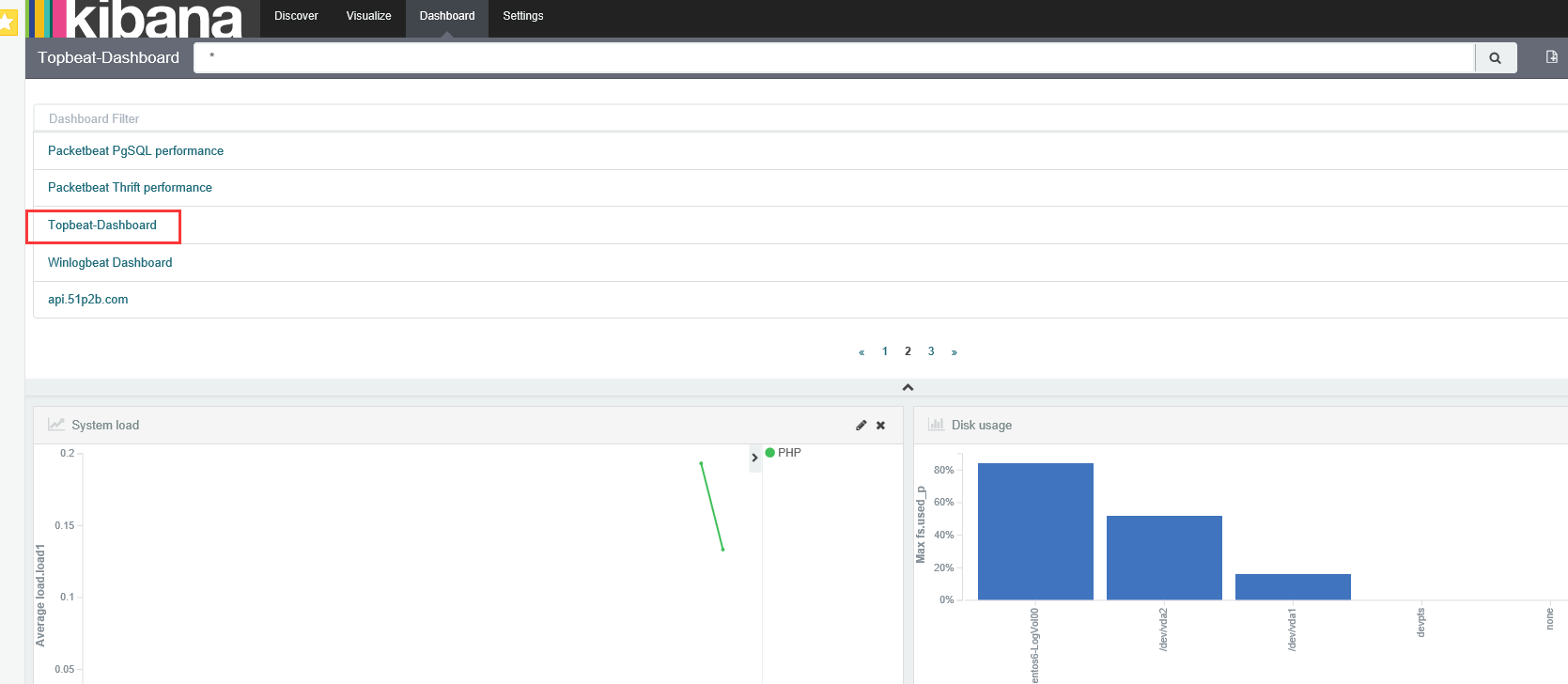
procs:["^$"] 此处可以过滤不需要的进程，采集全部进行 使用\*号

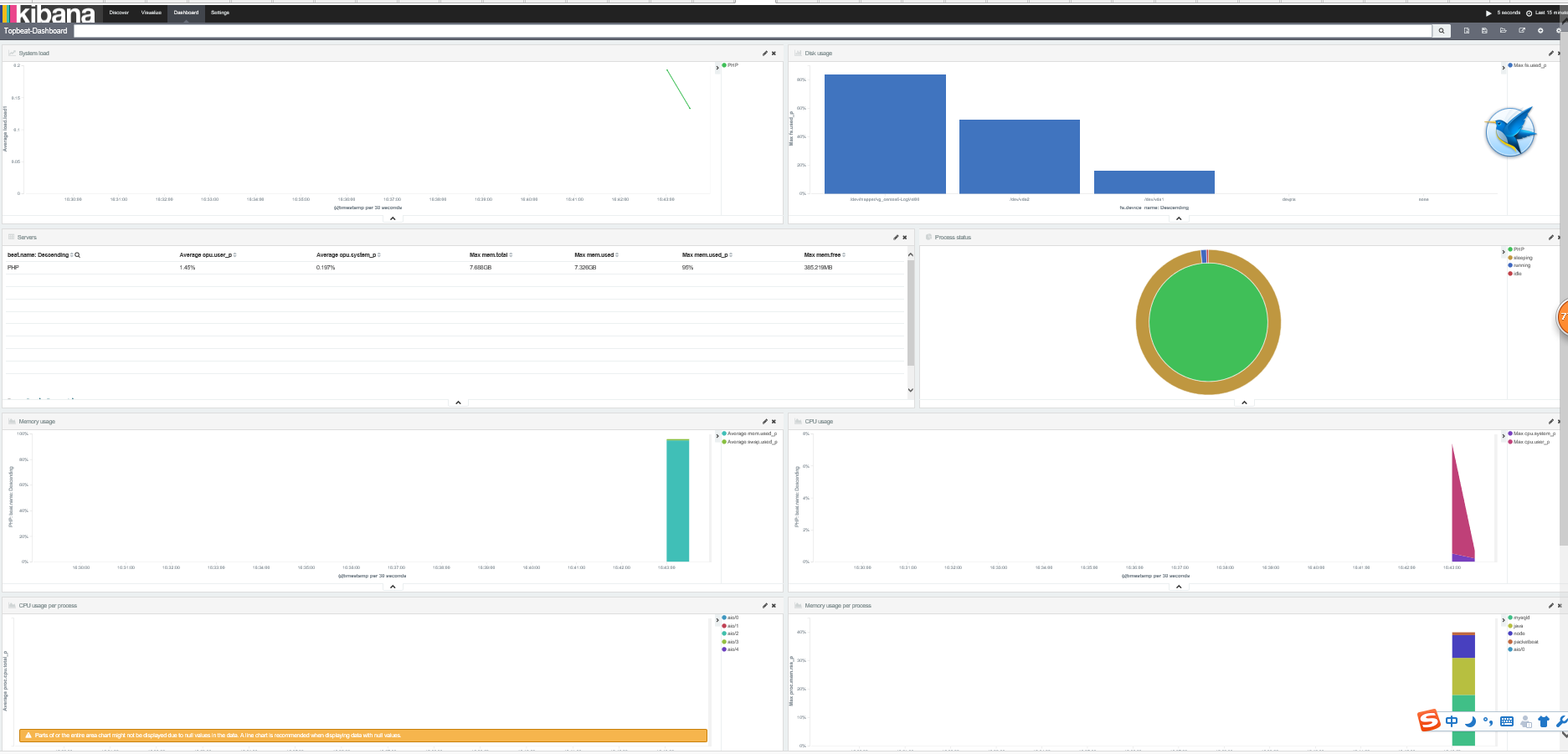
3.上传ElasticSearch数据模板  
    curl -XPUT 'http://192.168.1.217:9200/\_template/topbeat' -d@/etc/topbeat/topbeat.template.json      
      
4.启动topbeat服务  
 /etc/init.d/topbeat start

5.在kibana中查看

安装packetbeat时导入的仪表盘中包括了topbeat的仪表盘，显示如下

打开仪表盘里选择topbeat 仪表盘





## 安装filebeat

下载安装包：wget https://download.elastic.co/beats/filebeat/filebeat-1.2.3-x86\_64.rpm

安装：rpm -ivh filebeat-1.2.3-x86\_64.rpm

修改配置文件：/etc/filebeat/filebeat.yml

加载索引模板：curl -XPUT 'http://192.168.1.217:9200/\_template/filebeat' -d@/etc/filebeat/filebeat.template.json

启动：/etc/init.d/filebeat start

验证：在Kibana中可查看索引为filebeat-\*的日志

添加一个prospector：type字段然后使用document\_type:区分日志类型，如nginx日志，系统日志等等。

修改配置文件：/etc/filebeat/filebeat.yml 注意配置文件要严格缩进，否则会引起语法错误。

# Additional prospector

-

# Configuration to use stdin input

#input\_type: stdin

paths:

- /usr/local/nginx/logs/access\_71dai.log

document\_type: nginxlog

#这里的document\_type即对应logstash中的type字段。设置type便于在logstash中分类匹配，也便于在kibana中分类搜索和展示。

在logstash中按type过滤，如下，当type为nginxlog时执行grok匹配：

filter{

if [type] == "nginxlog"{

grok{

patterns\_dir => "/opt/logstash/patterns"

match => {

"message" => "%{NGINXACCESSAPI}"

}

overwrite => ["message"]

}

date {

match => ["log\_timestamp","dd/MMM/YYYY:HH:mm:ss Z"]

target => "@timestamp"

}

geoip {

source => "remote\_addr"

fields => ["country\_name", "region\_name", "city\_name", "real\_region\_name", "latitude", "longitude"]

remove\_field => [ "[geoip][longitude]","[geoip][latitude]","location","region\_name" ]

}

mutate {

convert => ["body\_bytes\_sent","integer"]

convert => ["request\_time","float"]

}

}

}

# ELK Packetbeat 协议和进程配置项

Packetbeat 协议部分包含支持每个协议的配置项，如ports, send\_request, send\_response 和 特定的协议选项。

目前Packetbeat支持以下协议：

* DNS
* HTTP
* Mysql
* PostgreSQL
* [Redis](http://www.ttlsa.com/redis/)
* Thrift-RPC
* [MongoDB](http://www.ttlsa.com/mongodb)
* [Memcache](http://www.ttlsa.com/nosql/memcache/)

配置如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | protocols:    dns:      ports: [53]      http:      ports: [80, 8080, 8000, 5000, 8002]      memcache:      ports: [11211]      mysql:      ports: [3306]      redis:      ports: [6379]      pgsql:      ports: [5432]      thrift:      ports: [9090] |

### 通用协议选项

下面的选项可用于所有协议：

**ports**

Packetbeat以该端口捕获具体协议流量。 Packetbeat 基于这个端口安装一个BPF过滤器。如果数据包不匹配该过滤器直接丢弃。Packetbeat还使用该端口以确定每个数据包使用哪个分析器解析。

**send\_request**

如果启用了，请求的信息(request字段)将发送到elasticsearch。默认是 false。 如果想索引整个请求该选项要启用。注意对于HTTP，请求体默认不包含，仅仅是HTTP头部。

**send\_response**

如果启用了，响应的信息(response字段)将发送到elasticsearch。默认是 false。 如果想索引整个请求该选项要启用。注意对于HTTP，请求体默认不包含，仅仅是HTTP头部。

如下图

[](http://www.ttlsa.com/wp-content/uploads/2015/12/http.jpg)

**transaction\_timeout**

每个协议事务超时时间。过期事务将不再关联输入响应，但是立即发送到elasticsearch。

### DNS配置选项

配置实例如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | protocols:    dns:      ports: [53]        # include\_authorities controls whether or not the dns.authorities field      # (authority resource records) is added to messages.      # Default: false      include\_authorities: true      # include\_additionals controls whether or not the dns.additionals field      # (additional resource records) is added to messages.      # Default: false      include\_additionals: true |

**include\_authorities**

如果启用， dns.authority 字段 (权威资源记录) 将加入到DNS 事件。默认false。

**include\_additionals**

如果启用，dns.additionals字段 (额外资源记录) 将加入到DNS 事件。默认false。

### HTTP 配置选项

配置实例如下：用的最多是这个，注意看

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | protocols:    http:        # Configure the ports where to listen for HTTP traffic. You can disable      # the http protocol by commenting the list of ports.      ports: [80, 8080, 8000, 5000, 8002]        # Uncomment the following to hide certain parameters in URL or forms attached      # to HTTP requests. The names of the parameters are case insensitive.      # The value of the parameters will be replaced with the 'xxxxx' string.      # This is generally useful for avoiding storing user passwords or other      # sensitive information.      hide\_keywords: ["pass", "password", "passwd"]        # Uncomment the following to export a list of extra HTTP headers. By      default is none sent.      send\_headers: ["User-Agent", "Cookie", "Set-Cookie"]        # Uncomment the following to export Cookie or Set-Cookie headers. By      # default is false.      split\_coookie: true        # Configure the HTTP header that contains the real IP address.      real\_ip\_header: "X-Forwarded-For" |

**hide\_keywords**

查询参数列表，Packetbeat将自动审查已存的事务。与这些参数相关的值将被替代成 'xxxxx'。默认情况下，是不会更改HTTP信息的。

Packetbeat 有这个选项，是因为不同于 SQL 流量只含有密码的hash值，HTTP 流量可能包含敏感的数据。为了降低安全风险，可配置此选项避免HTTP POST发送特定的参数。

**redact\_authorization**

当启用该选项， Packetbeat 隐藏  Authorization值和 Proxy-Authorization HTTP头部,并检查响应的字符串。

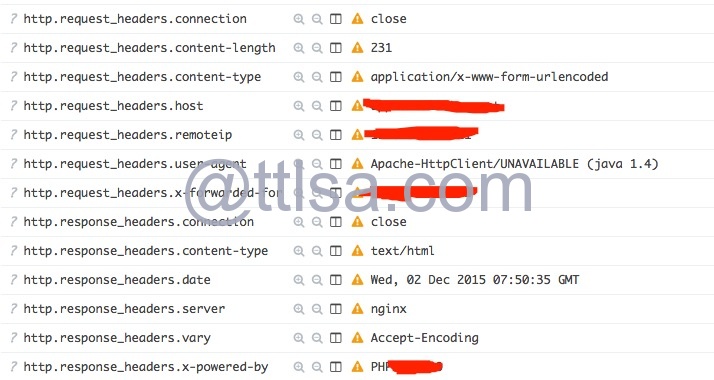
如果使用Basic Authentication验证，应该启用该选，因为可能包含base64未加密的用户名和密码。

**send\_headers**

捕获一系列头部名并发送到 Elasticsearch。这些头部以json格式放在headers字典里。

**send\_all\_headers**

发送头部白名单到 Elasticsearch，如果要发送所有头部，该选项设置为TRUE。默认是false。

[](http://www.ttlsa.com/wp-content/uploads/2015/12/send_all_headers.jpg)

include\_body\_for

列出内容类型列表，Packetbeat 将在response字段包含完整的HTTP有效载荷。该选项要和send\_response选项一起使用。

如：

|  |  |
| --- | --- |
| 1  2  3  4  5 | protocols:    http:      ports: [80, 8080]      send\_response: true      include\_body\_for: ["text/html"] |

**split\_cookie**

如果Cookie 或者 Set-Cookie 头部被发送, 该选项控制是否把它们分割成独自值。默认false。例如，设置了该选项， HTTP 响应将成下面的json格式结果：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | "response": {    "code": 200,    "headers": {      "connection": "close",      "content-language": "en",      "content-type": "text/html; charset=utf-8",      "date": "Fri, 21 Nov 2014 17:07:34 GMT",      "server": "gunicorn/19.1.1",      "set-cookie": {        "csrftoken": "S9ZuJF8mvIMT5CL4T1Xqn32wkA6ZSeyf",        "expires": "Fri, 20-Nov-2015 17:07:34 GMT",        "max-age": "31449600",        "path": "/"      },      "vary": "Cookie, Accept-Language"    },    "phrase": "OK"  } |

**real\_ip\_header**

提取真实IP地址。该信息用于 real\_ip 和 client\_location 索引字段。

### Memcache 配置选项

实例如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | memcache:      ports: [11211]      parseunknown: false      maxvalues: 0      maxbytespervalue: 100      transaction\_timeout: 200      udptransactiontimeout: 200 |

**parseunknown**

当启用该选项，迫使memcache文本协议分析器接受未知的命令。

**maxvalues**

信息(multi-get)存储值的最大数量。所有值base64编码。可能值如下：

* maxvalue: -1, 存储所有值 (基于文本协议 multi-get)
* maxvalue: 0, 不存储值 (默认)
* maxvalue: N, 最多可存 N 值

**maxbytespervalue**

复制每个值元素的最大字节数。值以base64编码，因此JSON文档中的实际大小将是maxbytespervalue的4倍。

**udptransactiontimeout**

事务的超时时间。默认 10000 毫秒。

### MySQL 和 PgSQL 配置选项

**max\_rows**

发送到elasticsearch的SQL信息的最大行数。默认10行。

**max\_row\_length**

发送到elasticsearch的SQL信息的最大字节。默认1024 字节。

### Thrift 配置选项

socket用的比较多，注意看。

实例如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | thrift:      transport\_type: socket      protocol\_type: binary      idl\_files: ["tutorial.thrift", "shared.thrift"]      string\_max\_size: 200      collection\_max\_size: 20      capture\_reply: true      obfuscate\_strings: true      drop\_after\_n\_struct\_fields: 100 |

**transport\_type**

Thrift 传输类型。 默认是 socket。

**protocol\_type**

Thrift 协议类型。 只接受二进制的TBinary协议, 这是默认的Thrift 协议。

**idl\_files**

Thrift 接口描述语言 (IDL)文件，Packetbeat 所监控的。 可选的，因为Thrift信息包含足够的信息去解码而无需IDL文件。 然而，提供 IDL使 Packetbeat 可包含参数和异常名字。

**string\_max\_size**

参数或返回值的字符串最大长度。如果超过该值，将会被自动截断，Packetbeat在字符串后面加上个点，标示被截断。默认200。

**collection\_max\_size**

Thrift list, set, map 或者 structure 最大元素个数。如果超过这个值，Packetbeat只捕获指定元素的个数。 Packetbeat 在集合后面添加一个虚拟的元素... 标明被截断。默认15。

**capture\_reply**

如果设置为false， Packetbeat 解码来自应答的方法名称，对剩余的响应信息简单的跳过。该选项对性能、磁盘使用情况或者数据保留很有用。默认TRUE。

**obfuscate\_strings**

如果设置为TRUE， Packetbeat 将以"\*"字符串取代方法参数、返回码或者异常结构发现的字符串。

**drop\_after\_n\_struct\_fields**

Packetbeat忽略整个事务前，可保留的最大字段数。这是个内存保护机制，不至于Packetbeat内存无限增长。因此通常设置为一个相对高的值。默认500。

### MongoDB 配置选项

用的也比较多，注意看

实例如下：

|  |  |
| --- | --- |
| 1  2  3  4  5 | mongodb:      send\_request: true      send\_response: true      max\_docs: 0      max\_doc\_length: 0 |

下面两个设置用于限制Packetbeat索引中的response字段的数据量。

**max\_docs**

响应索引中的response字段最大的文档数。默认10。 可以设置为0，不限制文档数。

Packetbeat 在最后添加 [...] 一行标明未保存其它文档，是因为该设置。

**max\_doc\_length**

单个文档索引中的response字段的最大字符数。默认5000。 可以设置为0不限制。

如果因为该设置文档被修剪，Packetbeat在文档末尾添加... 字符串以标示。

注意，以该方式限制文档意味着它们将不再是正确的json格式对象了。

### 维护网络拓扑的实时状态

Packetbeat知道每个事务的源服务器和目标服务器的名称。这个之前说过的，会维护主机名、ip映射的。不清楚的看前面文章。

Packetbeat存储拓扑信息到elasticsearch索引。要保持网络拓扑，使用elasticsearch作为输出，并设置save\_topology为TRUE。

### 进程 (可选)

可选的。捕获进程的通讯包。默认情况下，进程匹配是禁用的。只支持[Linux](http://www.ttlsa.com/linux/)系统。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | procs:    enabled: true    monitored:      - process: mysqld        cmdline\_grep: mysqld        - process: pgsql        cmdline\_grep: postgres        - process: nginx        cmdline\_grep: nginx        - process: app        cmdline\_grep: gunicorn |

**process**

指定进程名称，可随意定义以便可识。

**cmdline\_grep**

该名字用于识别正在运行的进程名。定期扫描进程表与之匹配，一旦匹配从 /proc/<pid>/cmdline读取信息。

# ELK beats通用配置说明

Beats配置文件是以YAML语法，该文件包含用于所有的beats的通用配置选项，以及其特点的选项。下面说说通用的配置，特定的配置要看各自beat文档。 通用的配置如下几部分：

* Shipper
* Output
* Logging(可选)
* Run Options（可选）

### Shipper

包含beat配置选项和一些控制其行为的常规设置。

其实每个配置选项的注释说明已经说的很清楚了，有些人就是视而不见。

如下所示：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | shipper:    # The name of the shipper that publishes the network data. It can be used to group    # all the transactions sent by a single shipper in the web interface.    # If this options is not defined, the hostname is used.    #name:      # The tags of the shipper are included in their own field with each    # transaction published. Tags make it easy to group servers by different    # logical properties.    tags: ["service-X", "web-tier"]      # Uncomment the following if you want to ignore transactions created    # by the server on which the shipper is installed. This option is useful    # to remove duplicates if shippers are installed on multiple servers.    ignore\_outgoing: true      # How often (in seconds) shippers are publishing their IPs to the topology map.    # The default is 10 seconds.    refresh\_topology\_freq: 10      # Expiration time (in seconds) of the IPs published by a shipper to the topology map.    # All the IPs will be deleted afterwards. Note, that the value must be higher than    # refresh\_topology\_freq. The default is 15 seconds.    topology\_expire: 15      # Configure local GeoIP database support.    # If no paths are not configured geoip is disabled.    #geoip:      #paths:      #  - "/usr/share/GeoIP/GeoLiteCity.dat"      #  - "/usr/local/var/GeoIP/GeoLiteCity.dat" |

#### name

beat名称，如果没设置以hostname名自居。该名字包含在每个发布事务的shipper字段。可以以该名字对单个beat发送的所有事务分组。

在启动时，每个beat将发送自己的IP、端口、名字到elasticsearch。这些信息存储在elasticsearch作为网络拓扑图，将每个beat的IP和端口与在这里你所指定的名字映射。

当一个beat接收到一个新的请求和响应称为事务，beat会查询elasticsearch查看网络拓扑是否包含该源服务器IP和端口以及目标服务器。如果该信息可用，在输出的client\_server字段被设置成运行在源服务器的beat名称，并且server字段被设置成运行在目标服务器的beat名称。

要在elasticsearch中使用拓扑图的话，必须设置save\_topology为TRUE并且elasticsearch为输出。



|  |  |
| --- | --- |
| 1  2 | shipper:    name: "ttlsa-shipper" |

#### tags

beat标签列表，包含在每个发布事务的tags字段。标签可用很容易的按照不同的逻辑分组服务器。例如，一个web集群服务器，可以对beat添加上webservers标签，然后在kibana的visualisation界面以该标签过滤和查询整组服务器。



|  |  |
| --- | --- |
| 1  2 | shipper:    tags: ["mysql-db", "aws", "rdb"] |

#### ignore\_outgoing

如果启用了ignore\_outgoing选项，beat将忽略从运行beat服务器上所有事务。不好描述，看下面的解释。

这是非常有用的，当两个beat发布相同的事务。因为一个beat认为是输出队列的事务，另一个beat认为是输入队列的事务。你可以结束这个重复的事务，启用该选项即可。

例如，有下面这个情景，三台服务器每台都安装了一个beat，t1在server1和server2之间交换事务，t2在server2和server3之间交换事务。

[beats](http://www.ttlsa.com/wp-content/uploads/2015/11/option_ignore_outgoing.png)

默认情况下，每个事务要被索引两次，因为beat2会看到两个事务。当ignore\_outgoing为false时，发布的事务是这样的：

* Beat1: t1
* Beat2: t1 and t2
* Beat3: t2

为了避免重复，需要强制beat只发送输入的事务，忽略本地服务器创建的事务。当ignore\_outgoing为true时，发布的事务是这样的：

* Beat1: none
* Beat2: t1
* Beat3: t2

#### refresh\_topology\_freq

拓扑图刷新的间隔。也就是设置每个beat向拓扑图发布其IP地址的频率。默认是10秒。

#### topology\_expire

拓扑的过期时间。在beat停止发布其IP地址时非常有用。当过期后IP地址将自动的从拓扑图中删除。默认是15秒。

#### geoip.paths

GeoIP数据库的搜索路径。beat找到GeoIP数据库后加载，然后对每个事务输出client的GeoIP位置。

推荐值为/usr/share/GeoIP/GeoLiteCity.dat 和/usr/local/var/GeoIP/GeoLiteCity.dat。

目前只有Packetbeat使用该选项。

### Output

可以配置多个输出来导出相关事务。当前支持的输出类型有：

* Elasticsearch
* Logstash
* [Redis](http://www.ttlsa.com/redis/) (不推荐)
* File
* Console

可以同时启用一个或多个输出。输出插件负责发送JSON格式化的事务数据到下一个管道。同时还维护网络拓扑。

#### Elasticsearch Output

当指定elasticsearch作为输出，beat通过elasticsearch HTTP API将事务直接发送到elasticsearch。



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | output:    elasticsearch:      # The Elasticsearch cluster      hosts: ["http://es.ttlsa.com:9200"]        # Comment this option if you don't want to store the topology in      # Elasticsearch. The default is false.      # This option makes sense only for Packetbeat      save\_topology: true        # Optional index name. The default is packetbeat and generates      # [packetbeat-]YYYY.MM.DD keys.      index: "packetbeat"        # List of root certificates for HTTPS server verifications      cas: ["/etc/pki/root/ca.pem"]        # TLS configuration.      tls:        # Certificate for TLS client authentication        certificate: "/etc/pki/client/cert.pem"          # Client Certificate Key        certificatekey: "/etc/pki/client/cert.key" |

启用SSL，在hosts配置项指定https。



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | output:    elasticsearch:      # The Elasticsearch cluster      hosts: ["https://localhost:9200"]        # Comment this option if you don't want to store the topology in      # Elasticsearch. The default is false.      # This option makes sense only for Packetbeat      save\_topology: true        # HTTP basic auth      username: "admin"      password: "s3cr3t" |

如果elasticsearch节点通过IP:PORT定义，需要加protocol: https，如下：



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | output:    elasticsearch:      # The Elasticsearch cluster      hosts: ["localhost"]        # Optional http or https. Default is http      protocol: "https"        # Comment this option if you don't want to store the topology in      # Elasticsearch. The default is false.      # This option makes sense only for Packetbeat      save\_topology: true        # HTTP basic auth      username: "admin"      password: "s3cr3t" |

#### hosts

可以指定连接的elasticsearch节点列表。事件将随机分配到这些节点。如果某个节点不可达，事件将自动发送到另一个节点。每个elasticsearch节点定义个格式：URL或者IP:PORT。如http://es1.ttlsa.com,https://es2.ttlsa.com或者10.0.0.1。如果没有指定端口默认是9200。

当以IP:PORT形式定义elasticsearch节点，则schema和path取自protocol和path配置项。如：



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | output:    elasticsearch:      # The Elasticsearch cluster      hosts: ["10.45.3.2:9220", "10.45.3.1:9230"]        # Optional http or https. Default is http      protocol: https        # HTTP Path at which each Elasticsearch server lives      path: /elasticsearch |

在上面的例子中，Elasticsearch可用节点是https://10.45.3.2:9220/elasticsearch和https://10.45.3.1:9230/elasticsearch。

#### worker

配置每台主机发送事件到elasticsearch的worker数量。在负载均衡模式下最好启用。例如，2台主机和3个worker，一共将启动6个worker，每台主机3个worker。

#### host (不推荐)

elasticsearch服务的主机。该选项不建议使用，已经被hosts替换。

#### port (不推荐)

elasticsearch服务的端口。该选项不建议使用，已经被hosts替换。

#### username

连接elasticsearch的基础验证用户名。

#### password

连接elasticsearch的基础验证密码。

#### protocol

定义哪种协议可达elasticsearch。选项有http或者https。默认是http。但是，如果在hosts配置项指定了URL，URL中指定的协议将覆盖protocol值。

#### path

调用HTTP API的前置路径前缀。一般用在elasticsearch监听在HTTP反向代理，同时又自定义API前缀。

#### index

指定写入事件的索引根名称。默认是beat名称。例如Packetbeat，根索引名称是[packetbeat-]YYYY.MM.DD (如, packetbeat-2015.11.29)。

#### max\_retries

发送到特定logstash的最大尝试次数。如果达到该次数仍不成功，事件将被丢弃。默认是3。

值0表示禁用重试。值小于0将无限重试知道事件已经发布。

如果输出插件把事件丢弃，每个beat要实现必须去顶是否要丢失事件或者尝试再次发送。如果到达max\_retries后发送操作还是不成功，beat可选通知。

#### bulk\_max\_size

单个elasticsearch批量API索引请求的最大事件数。默认是50。

#### timeout

elasticsearch请求超时事件。默认90秒。

#### flush\_interval

新事件两个批量API索引请求之间需要等待的秒数。如果bulk\_max\_size在该值之前到达，额外的批量索引请求生效。

#### save\_topology

elasticsearch是否保持拓扑。默认false。该值只支持Packetbeat。

#### topology\_expire

elasticsearch保存拓扑信息的有效时间。默认15秒。

#### tls

配置TLS参数选项，如证书颁发机构等，用于基于https的连接。如果tls丢失，主机的CAs用于https连接elasticsearch。

#### Logstash Output

logstash输出通过使用lumberjack协议将事件直接发送到logstash。要使用此选项，必须在logstash上安装和配置logstash-input-beats插件。logstash允许额外的处理和生成事件路由。

每个发送到logstash事件包含额外的索引和过滤元数据。如：



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | {      ...      "@metadata": {        "beat": "<beat>",        "type": "<event type>"      }  } |

在logstash，你可以配置elasticsearch输出插件使用元数据和事件类型进行索引。

下面的logstash1.5配置文件设置logstash使用beat报告的索引和文档类型将事件索引到elasticsearch。索引使用取决于logstash确定的@timestamp字段。



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | input {    beats {      port => 5044    }  }    output {    elasticsearch {      host => "localhost"      port => "9200"      protocol => "http"      index => "%{[@metadata][beat]}-%{+YYYY.MM.dd}"      document\_type => "%{[@metadata][type]}"    }  } |

logstash 2.x 相同的配置：



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | input {    beats {      port => 5044    }  }    output {    elasticsearch {      hosts => ["http://localhost:9200"]      index => "%{[@metadata][beat]}-%{+YYYY.MM.dd}"      document\_type => "%{[@metadata][type]}"    }  } |

事件被索引到elasticsearch，类似于将事件通过beats直接索引到elasticsearch。如下配置，如何配置beat使用logstash：



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | output:    logstash:      hosts: ["localhost:5044"]        # index configures '@metadata.beat' field to be used by Logstash for      # indexing. By Default the beat name is used (e.g. filebeat, topbeat, packetbeat)      index: mybeat |

#### hosts

要连接logstast的服务器列表。每个列表项可以包含端口号。如果没有指定端口，将使用默认值。

#### worker

配置每个主机发布事件的worker数量。在负载均衡模式下最好启用。例如，如果2台主机和3个worker，一共6个worker将启动，每台3个worker。

#### loadbalance

如果设置为TRUE和配置了多台logstash主机，输出插件将负载均衡的发布事件到所有logstash主机。如果设置为false，输出插件发送所有事件到随机的一台主机上，如果选择的不可达将切换到另一台主机。默认是false。



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | output:    logstash:      hosts: ["localhost:5044", "localhost:5045"]        # configure index prefix name      index: mybeat        # configure logstash plugin to loadbalance events between the logstash instances      loadbalance: true |

#### port

hosts配置项如果没有指定端口好将使用的默认端口。默认是10200。

#### index

如上解释

#### tls

如上解释

#### timeout

等待logstash响应的超时时间，默认30秒。

#### max\_retries

如上解释

##### Redis Output (不推荐)

被beats代替，不推荐使用了。不再此做介绍了。

### File Output

文件输出将事务转存到一个文件，每个事务是一个JSON格式。主要用于测试。也可以用作logstash输入。



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | output:      # File as output    # Options:    # path: where to save the files    # filename: name of the files    # rotate\_every\_kb: maximum size of the files in path    # number of files: maximum number of files in path    file:      path: "/tmp/packetbeat"      filename: packetbeat      rotate\_every\_kb: 1000      number\_of\_files: 7 |

#### path

指定文件保存的路径。必须的。

#### filename

文件名。默认是 Beat 名称。上面配置将生成 packetbeat, packetbeat.1, packetbeat.2 等文件。

#### rotate\_every\_kb

定义每个文件最大大小。当大小到达该值文件将轮滚。默认值是1000 KB。

#### number\_of\_files

保留文件最大数量。文件数量到达该值将删除最旧的文件。默认是7，一星期。

### Console Output

标准输出，JSON 格式。

|  |  |
| --- | --- |
| 1  2  3 | output:    console:      pretty: true |

#### pretty

如果设置为TRUE，事件将很友好的格式化标准输出。默认false。

### Logging (Optional)

配置beats日志。日志可以写入到syslog也可以是轮滚日志文件。默认是syslog。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | logging:    level: warning      # enable file rotation with default configuration    to\_files: true      # do not log to syslog    to\_syslog: false      files:      path: /var/log/mybeat      name: mybeat.log      keepfiles: 7 |

### Logging options

#### to\_syslog

如果启用发送所有日志到系统日志。

#### to\_files

日志发送到轮滚文件。

#### level

日志级别。debug, info, warning, error 或 critical。如果使用debug，但没有配置selectors，\* selectors将被使用。默认error。

#### selectors

The list of debugging-only selector tags used by different Beats components. Use \* to enable debug output for all components. For example add publish to display all the debug messages related to event publishing. When starting the Beat, selectors can be overwritten using the -d command line option (-dalso sets the debug log level).

#### files.path

日志文件目录。

#### files.name

日志文件名称。默认是Beat 名称。

#### files.rotateeverybytes

日志文件的最大大小。默认 10485760 (10 MB)。

#### files.keepfiles

保留日志周期。 默认 7。值范围为2 到 1024。

### Logging Format

每个日志类型有不同的日志格式：

* to syslog: 系统日志加上自己的时间戳。
* to file: RFC 3339 格式用于时间戳2006-01-02T15:04:05Z07:00 WARN log-message. 该给事包含时区和日志级别。
* to stderr: UTC  格式用于时间戳 2015/11/12 09:03:37.369262 geolite.go:52: WARN log-message。该格式包括UTC时间戳和毫秒，主要用于调试。

### Run Options (Optional)

beats创建套接字后放权。打开套接字需要root访问权限，但不是所有都需要该权限。因此，建议以普通用户运行beats。可以通过uid、gid来指定。

在[Linux](http://www.ttlsa.com/linux/)上，setuid不会改变所有线程的uid，所以Go garbage收集器还将以root用户运行。另外注意，进程监控需要以root权限运行。

|  |  |
| --- | --- |
| 1  2  3 | runoptions:    uid=501    gid=501 |

# 使用Shield保护ELK平台兼权限控制

插件文档：https://www.elastic.co/downloads/shield

ELK系统默认并不含有用户认证功能，基本上任何人都可以随意读写ElasticSearch的API并获取数据，这时该如何对ELK系统做好防护工作呢？

## 目标

在读完这篇教程后，您可以学会：

* 阻拦未授权的用户对ELK平台的访问
* 允许不同用户访问不同的index

## 方法

这里我们使用Elastic公司的Shield来完成这个工作

## Shield是什么

Shield是Elastic公司为ElasticSearch开发的一个安全插件。在安装此插件后，Shield会拦截所有对ElasticSearch的请求，并加上认证与加密，保障ElasticSearch及相关系统的安全性。   
它支持下面的功能：

### 用户认证

使用Shield可以定义一系列已知的用户，并用其认证用户请求。这些用户存在于抽象的“域”中。一个域可能是下面几种类型：

* LDAP服务
* Active Directory服务
* 本地esusers配置文件（类似/etc/passwd)

### 权限控制

Shield的权限控制包含下面几种元素：

* **被保护的资源Secured Resource**：权限所应用到的对象，比如某个index，cluster等等
* **特权Priviliege**：角色对对象可以执行的一种或多种操作，比如read,write等。还可以是indicies:/data/read/perlocate等某种对象特有的操作。
* **许可Permissions**：对被保护的资源拥有的一个或多个特权，如read on the "products" index。
* **角色Role**：由许可组成的有名字的集合
* **用户Users**：用户实体，可以被赋予0种，1种或多种角色，他们可以对被保护的资源执行相应角色所拥有的各种特权。

### 集群节点认证与信道加密

Shield使用SSL/TLS加密相应端口(9300)，防止集群被未授权的机器监听或干扰。

### IP 过滤

Shield支持基于IP的访问控制。

### 审计

Shield可以在ElasticSearch的日志中输出每次鉴权操作的详细信息，包括用户名，操作，操作是否被允许等等。

## 安装Shield

### 前提条件

* 您安装了Java7或更新版本
* 您将ElasticSearch 1.5.0+解压安装到了本机上。如果您使用APT或YUM安装，默认的安装目录可能在/usr/share/elasticsearch。

### 开始安装

1. 进入ElasticSearch安装目录：cd /usr/share/elasticsearch
2. 安装ElasticSearch许可插件：bin/plugin install license

Shield是商业插件，需要ElasticSearch的商业许可。第一次安装许可的时候，会提供30天的免费试用权限。30天后，Shield将会屏蔽cluster health, cluster stats, index stats这几个API，其余功能不受影响。

1. 下面安装Shield插件：bin/plugin install shield
2. 重启服务：service elasticsearch restart
3. 新建一个ElasticSearch管理员账户，这里会让您填写新密码：bin/shield/esusers useradd es\_admin -r admin 注意es\_admin是用户名 -r admin表示是一个管理员账号
4. 现在试试用RESTFUL API访问ElasticSearch，应该会被拒绝：curl -XGET 'http://localhost:9200/'
5. 在请求上加上用户名和密码：curl -u es\_admin -XGET 'http://localhost:9200/'

如果认证失败的话，可能需要在/etc/elasticsearch/elasticsearch.yml中加入下面的内容：

shield:

authc:

realms:

default:

type: esusers

order: 0

enabled: true

files:

users: "/etc/elasticsearch/shield/users"

users\_roles: "/etc/elasticsearch/shield/users\_roles"

到这里，Shield基本功能已经安装完毕。

## 配置其余软件

### 配置Logstash

1. 在ElasticSearch服务器上，使用esusers创建Logstash用户：/usr/share/elasticsearch/bin/shield/esusers useradd logstashserver -r logstash

密码leichi#123

1. 在Logstash服务器上，修改output模块的配置文件，例如：

output {

elasticsearch {

hosts=>["192.168.1.217:9200"]

index=>"nginx\_log-%{+YYYY.MM}"

user=>"logstashserver"

password=>" leichi#123" }

# stdout { codec => rubydebug }

}

配置/etc/elasticsearch/shield/roles.yml 使logstash对所有索引\*都有权限，这里names后边配置为\*；如果是对某个索引有权限，可以配置为nginx\_log-\*匹配索引名。

logstash:

cluster:

- manage\_index\_templates

indices:

- names: '\*'

privileges:

- write

- delete

- create\_index  
之后重启Logstash服务即可。

### 配置Kibana

基本配置

1. 在ElasticSearch服务器上，使用esusers创建属于kibana4\_server的用户：/usr/share/elasticsearch/bin/shield/esusers useradd kibanaserver -r kibana4\_server

密码leichi#123

列出现有用户

[root@PHP bin]# /usr/share/elasticsearch/bin/shield/esusers list

demo\_nginx : nginx\_user,kibana4\_server,kibana4\*

logserver : logstash

kibana4-server : kibana4\_server

demo\_mail : mail\_user,kibana4\_server,kibana4\*

logstashserver : logstash

es\_admin : admin

kibanaserver : kibana4\_server

1. 在Kibana服务器上，编辑/opt/kibana/config/kibana.yml,找到下面这一部分并修改：

在末尾增加两行

elasticsearch.username: "kibanaserver"

elasticsearch.password: "leichi#123"

（备注：在我的测试环境中如下红色字体部分没有修改也能正常使用）

# If your Elasticsearch is protected with basic auth, this is the user credentials

# used by the Kibana server to perform maintence on the kibana\_index at statup. Your Kibana

# users will still need to authenticate with Elasticsearch (which is proxied thorugh

# the Kibana server)

kibana\_elasticsearch\_username: kibanaserver #Kibana服务将用这个用户名访问ElasticSearch服务器。

kibana\_elasticsearch\_password: woshimima #密码

之后重启Kibana服务即可。您可能需要使用前面的es\_admin账户登录Kibana网页端。

权限控制

在实际的生产环境中，经常需要让不同的角色访问不同的index，比如让Nginx管理员只能看到nginx相关的日志，Mail管理员只看到mail相关的用户，这时候就需要使用到权限控制功能。   
首先我们编辑ElasticSearch服务器的/etc/elasticsearch/shield/roles.yml,

（备注：如下红色字体部分没有执行，也能正常控制权限：注释掉kibana4.indicies.\*部分，即去掉用户读取所有index的权限。如下：）

# The required permissions for kibana 4 users.

kibana4:

cluster:

- cluster:monitor/nodes/info

- cluster:monitor/health

indices:

# '\*':

# - indices:admin/mappings/fields/get

# - indices:admin/validate/query

# - indices:data/read/search

# - indices:data/read/msearch

# - indices:admin/get

'.kibana':

- indices:admin/exists

- indices:admin/mapping/put

- indices:admin/mappings/fields/get

- indices:admin/refresh

- indices:admin/validate/query

- indices:data/read/get

- indices:data/read/mget

- indices:data/read/search

- indices:data/write/delete

- indices:data/write/index

- indices:data/write/update

- indices:admin/create

之后再roles.yml的末尾加上相关用户的权限配置：

nginx\_user: #nginx\_user 角色定义

indices: #index部分

'api\_log-\*': read #指定nginx\_user可以读取所有匹配' api\_log-\*'的索引。

mail\_user: #mail\_user 角色定义，用法同上

indices:

'nginx\_log-\*': read

现在我们使用esuser新建两个用户，分属两个组

/usr/share/elasticsearch/bin/shield/esusers useradd demo\_nginx -r nginx\_user

/usr/share/elasticsearch/bin/shield/esusers useradd demo\_mail -r mail\_user

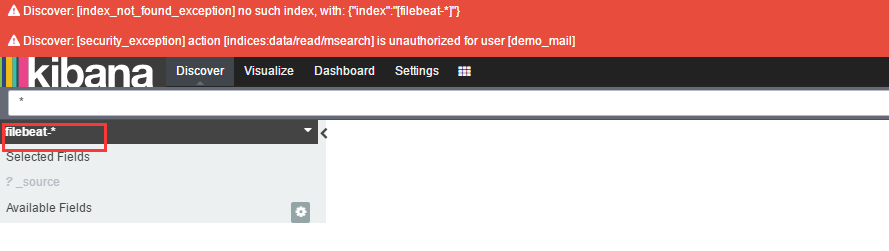
再把它们同时加入kibana4组中：

/usr/share/elasticsearch/bin/shield/esusers roles demo\_nginx -a kibana4\_server

/usr/share/elasticsearch/bin/shield/esusers roles demo\_mail -a kibana4\_server

这时再用不同的用户登录Kibana界面，就可以看到不同的内容了。

当用户访问没有权限的索引，即会提示如下：



### 配置packetbeat

vim /etc/packetbeat/packetbeat.yml

output:

### Elasticsearch as output

elasticsearch:

# Array of hosts to connect to.

# Scheme and port can be left out and will be set to the default (http and 9200)

# In case you specify and additional path, the scheme is required: http://localhost:9200/path

# IPv6 addresses should always be defined as: https://[2001:db8::1]:9200

hosts: ["192.168.1.217:9200"]

# Optional protocol and basic auth credentials.

#protocol: "https"

username: "logstashserver"

password: "leichi#123"