制作apache镜像

```
#Dockerfile中所有的指令，必须是大写的(例如: FROM, RUN, COPY等)
#FROM 指定基础镜像，Dockerfile会对基础镜像进行编辑，生成新的镜像
#MAINTAINER 指定创建镜像者的信息
#RUN 指定制作命令，一条RUN，就代表一条要在容器内执行的命令
#ENV 指定环境变量
#EXPOSE 开启httpd服务要使用的端口，80和443
#WORKDIR 指定启动容器后的，默认工作目录
#ADD 指拷贝，Dockerfile目录下文件，拷贝到容器内(tar.gz,tar.bz2格式会自动解压)
#CMD 指定默认启动命令

js(1.252)主机的centos.tar.gz的镜像上传到msater主机
[root@js ~]# scp /root/project3/centos.tar.gz 192.168.1.21:/root/

编写httpd的Dockerfile文件
[root@master ~]# docker load -i centos.tar.gz
[root@master ~]# mkdir bb
[root@master ~]# cd bb/
[root@master bb]# cp /etc/yum.repos.d/CentOS-Base.repo ./
[root@master bb]# echo "hello world" > index.html
[root@master bb]# vim Dockerfile
FROM centos:latest
RUN  rm -rf /etc/yum.repos.d/*
ADD  CentOS-Base.repo /etc/yum.repos.d/CentOS-Base.repo
RUN  yum -y install httpd
ENV  LANG=C
EXPOSE 80
WORKDIR /var/www/html
ADD  index.html /var/www/html/index.html
CMD  ["/usr/sbin/httpd","-DFOREGROUND"]

build 创建新镜像；-t 指定新镜像名字和标签；. 指定Dockerfile文件所在的目录
[root@master bb]# docker build -t myos:httpd .

验证结果：
[root@master bb]# docker images
[root@master bb]# docker run -itd myos:httpd        #后台启动容器，因为是一个服务
[root@master bb]# docker ps                        #查看正在使用的容器
[root@master bb]# docker inspect 800b21aa9736        #查看容器的详细信息
[root@master bb]# curl http://172.17.0.2
hello world

上传镜像到harbor主机
[root@master bb]# docker tag myos:httpd 192.168.1.100:80/library/myos:httpd
[root@master bb]# docker push 192.168.1.100:80/library/myos:httpd

制作filebea镜像，随便找一台机器，安装filebeat的软件包
[root@js ~]# scp /root/project3/ELK/filebeat-1.2.3-x86_64.rpm 192.168.1.72:/root/

[root@es-0002 ~]# yum -y install filebeat-1.2.3-x86_64.rpm
修改filebeat的配置文件
[root@es-0002 ~]# vim /etc/filebeat/filebeat.yml
15     - /var/weblog/access_log      #指定filebeat要读取的日志文件
72    document_type: apache_log     #修改日志标签，方便用户区分日志来源；不同日志修改不同标签
```

```
183 # elasticsearch:                    #注释掉elasticsearch
188 # hosts: ["localhost:9200"]         #注释掉hosts
278  logstash:                          #取消logstash的注释
280   hosts: ["192.168.1.75:5044"]      #指定logstash主机IP地址
[root@es-0002 ~]# systemctl restart filebeat
[root@es-0002 ~]# ps -C filebeat
  PID TTY          TIME CMD
 1321 ?        00:00:00 filebeat

master主机制作filebeat镜像并上传
[root@master bb]# mkdir /root/cc
[root@master bb]# cd /root/cc
[root@master cc]# cp /etc/yum.repos.d/CentOS-Base.repo ./

拷贝filebeat配置文件到master主机
[root@es-0002 ~]# scp /etc/filebeat/filebeat.yml /root/filebeat-1.2.3-x86_64.rpm
192.168.1.21:/root/cc
[root@master cc]# vim Dockerfile
FROM centos:latest
RUN  rm -rf /etc/yum.repos.d/*
ADD  CentOS-Base.repo /etc/yum.repos.d/CentOS-Base.repo
ADD  filebeat-1.2.3-x86_64.rpm  ./
RUN  yum -y install ./filebeat-1.2.3-x86_64.rpm
ADD  filebeat.yml /etc/filebeat/filebeat.yml
CMD  ["/usr/bin/filebeat", "-c", "/etc/filebeat/filebeat.yml"]

[root@master cc]# docker build -t myos:filebeat .
[root@master cc]# docker run -itd myos:filebeat    #后台启动容器，因为是一个服务
[root@master cc]# docker ps  | grep filebeat       #查看容器是否运行
[root@master cc]# docker tag myos:filebeat 192.168.1.100:80/library/myos:filebeat
[root@master cc]# docker push 192.168.1.100:80/library/myos:filebeat
```

登录harbor查看，可以看到此镜像

```
从私有仓库harbor中pull镜像的时候，k8s集群使用类型为docker-registry的Secret进行认证。
现在创建一个Secret，名称为regcred：
master主机认证，登录harbor
[root@master ~]# kubectl create secret docker-registry regcred --docker-server=192.168.1.100:80
--docker-username=admin --docker-password=Harbor12345

查看regcred的详细信息，其中.dockerconfigjson的值包含了登录harbor的用户名和密码等信息
[root@master ~]# kubectl get secret regcred --output=yaml

通过以下命令进行查看：
[root@master ~]# kubectl get secret regcred --output="jsonpath={.data.\.dockerconfigjson}" |
base64 -d
```

```
创建文件baseos.yaml，使用Secret regcred
[root@master ~]# vim baseos.yaml
---
kind: Pod
apiVersion: v1
metadata:
  name: mypod
spec:
  containers:
  - name: mylinux
    image: 192.168.1.100:80/library/myos:httpd
  imagePullSecrets:
  - name: regcred
[root@master ~]# kubectl create -f baseos.yaml

查看没有任何问题，访问也是没有问题
[root@master ~]# kubectl get pod -o wide
NAME   READY STATUS   RESTARTS AGE IP          NODE       NOMINATED NODE   READINESS GATES
mypod 1/1    Running   0       11m 10.244.3.2  node-0002  <none>       <none>
[root@master ~]# curl 10.244.3.2
```

```
编写apachelog.yaml资源清单文件，收集apache日志，到elk可以进行分析
[root@master ~]# vim apachelog.yaml
---
kind: Deployment
apiVersion: apps/v1
metadata:
  name: weblog
spec:
  selector:
    matchLabels:
      myapp: weblog
  replicas: 1
  template:
    metadata:
      labels:
        myapp: weblog
    spec:
      volumes:
      - name: log-data
        hostPath:
          path: /var/weblog
          type: DirectoryOrCreate
      containers:
      - name: apache
        image: 192.168.1.100:80/library/myos:httpd
```

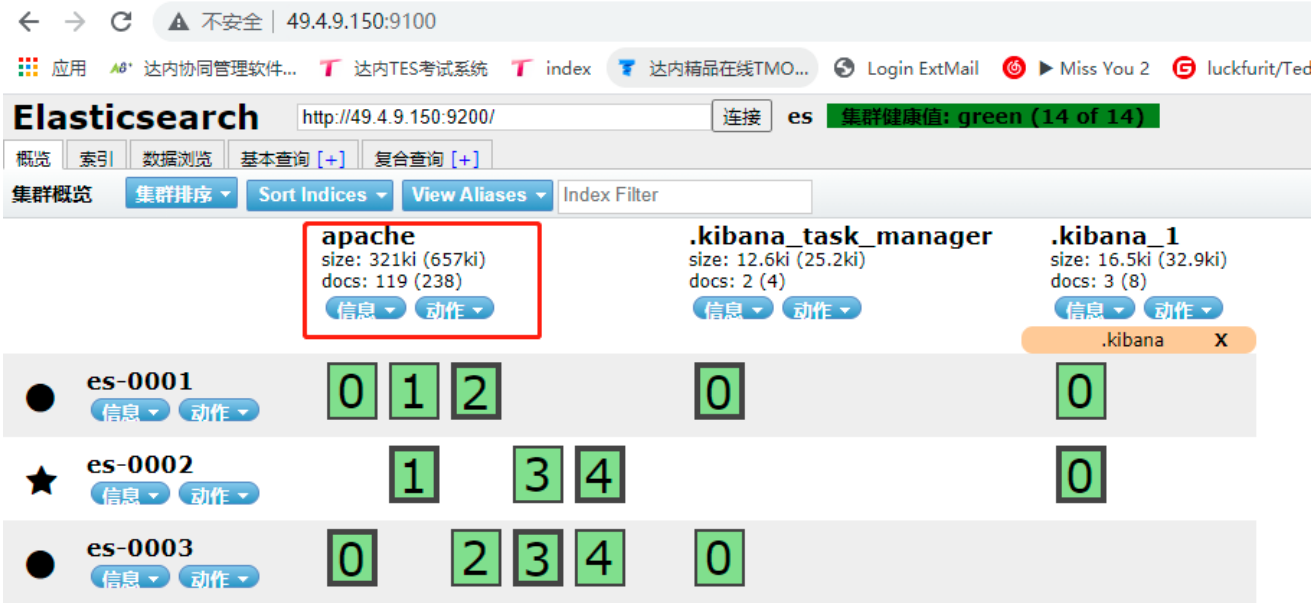
```
        volumeMounts:
        - name: log-data
          mountPath: /var/log/httpd
        ports:
        - protocol: TCP
          containerPort: 80
      - name: filebeat-backend
        image: 192.168.1.100:80/library/myos:filebeat
        volumeMounts:
        - name: log-data
          mountPath: /var/weblog
      restartPolicy: Always
      imagePullSecrets:         #新添加，关于harbor认证的操作
      - name: regcred
[root@master ~]# kubectl apply -f apachelog.yaml    #创建资源

查看在哪台机器上面启动，就去哪台机器上面看日志
[root@master ~]# kubectl get pod -o wide
NAME   READY  STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES
weblog-688dd9768c-vwkbs 2/2 Running 0  4m24s  10.244.3.4  node-0002  <none>      <none>

[root@node-0002 ~]# ls /var/weblog/
access_log error_log
[root@node-0002 ~]# cat /var/weblog/access_log
```

```
安装logstash服务1.75
[root@logstash ~]# yum -y install logstash
配置logstash
[root@logstash ~]# vim /etc/logstash/logstash.conf
input{
 stdin{ codec => "json" }
 file {
   path => ["/tmp/a.log","/tmp/b.log"]
  start_position => "beginning"
  sincedb_path => "/var/lib/logstash/sincedb"
 }

 beats{
  port => 5044
 }
 }

filter{
  if [type] == "apache_log" {
  grok {
   match => { "message" => "%{COMBINEDAPACHELOG}" }
  }
 }
}

output{
 stdout{ codec => "rubydebug" }
 if [type] == "apache_log" {
 elasticsearch {
   index => "apache"
   hosts => ["es-0001:9200","es-0002:9200","es-0003:9200"]
  }
 }
 }
[root@logstash ~]# /usr/share/logstash/bin/logstash -f /etc/logstash/logstash.conf
```

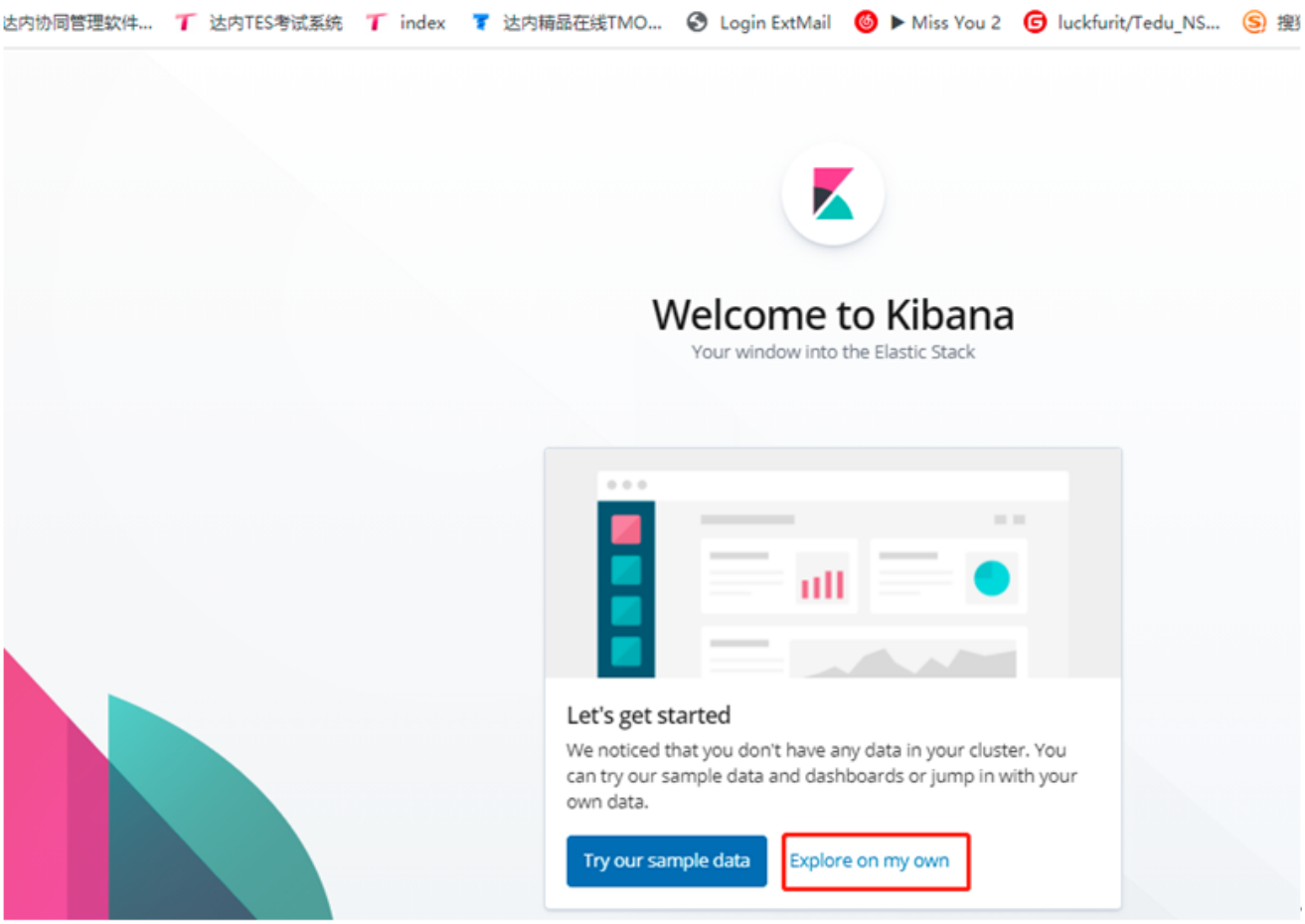
master主机访问容器，logstash可以看到数据输出信息

```
{
    "httpversion" => "1.1",
           "auth" => "-",
        "request" => "/",
          "agent" => "\"curl/7.29.0\"",
       "@version" => "1",
        "message" => "10.244.0.0 - - [03/Aug/2021:06:49:54 +0
000] \"GET / HTTP/1.1\" 200 12 \"-\" \"curl/7.29.0\"",
           "beat" => {
        "hostname" => "node-0002",
            "name" => "node-0002"
    },
         "offset" => 602,
           "tags" => [
            "beats_input_codec_plain_applied"
    ],
           "host" => "node-0002",
         "fields" => nil,
          "count" => 1,
         "source" => "/var/weblog/access_log",
          "bytes" => "12",
     "input_type" => "log",
       "referrer" => "\"-\"",
           "type" => "apache_log",
          "ident" => "-",
       "response" => "200",
           "verb" => "GET",
```

访问ES的head 插件查看结果，连接查看有apache的索引，里面也有数据，访问成功

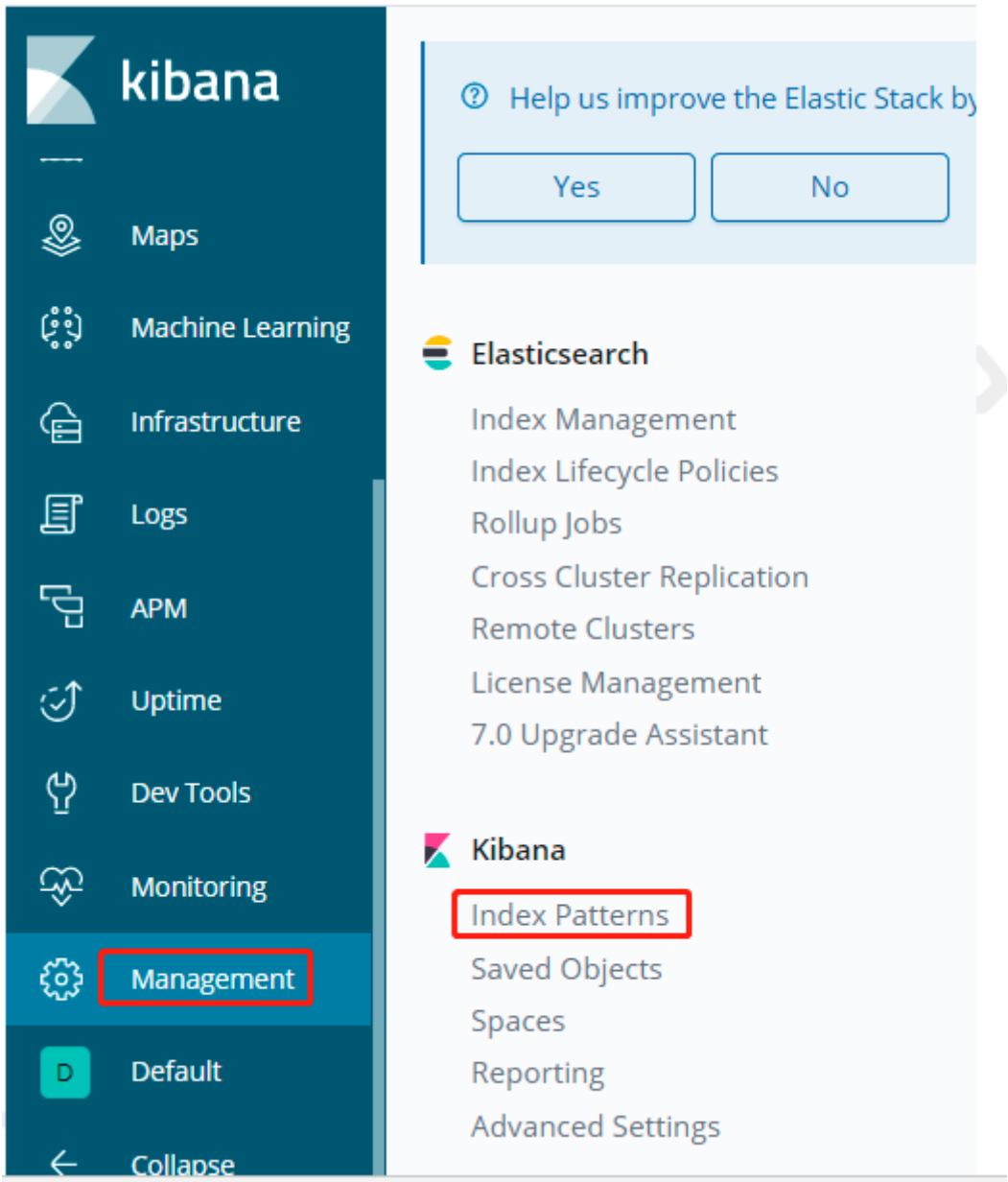


使用kibana 导入索引数据，绘制图形

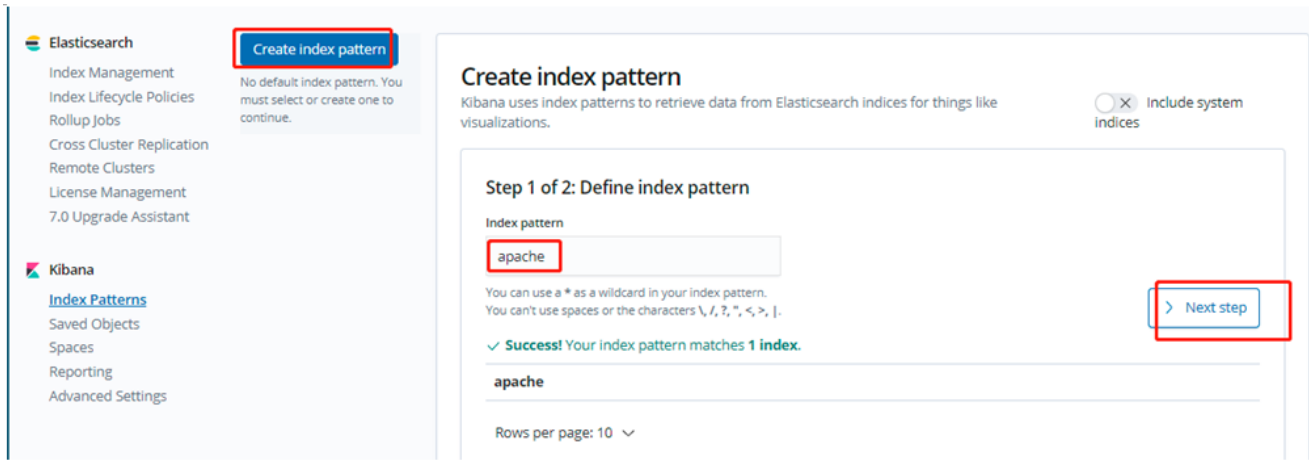


将前面案例收集的数据，进行可视化展示。并使用Metricbeat自带的仪表盘进行数据展示

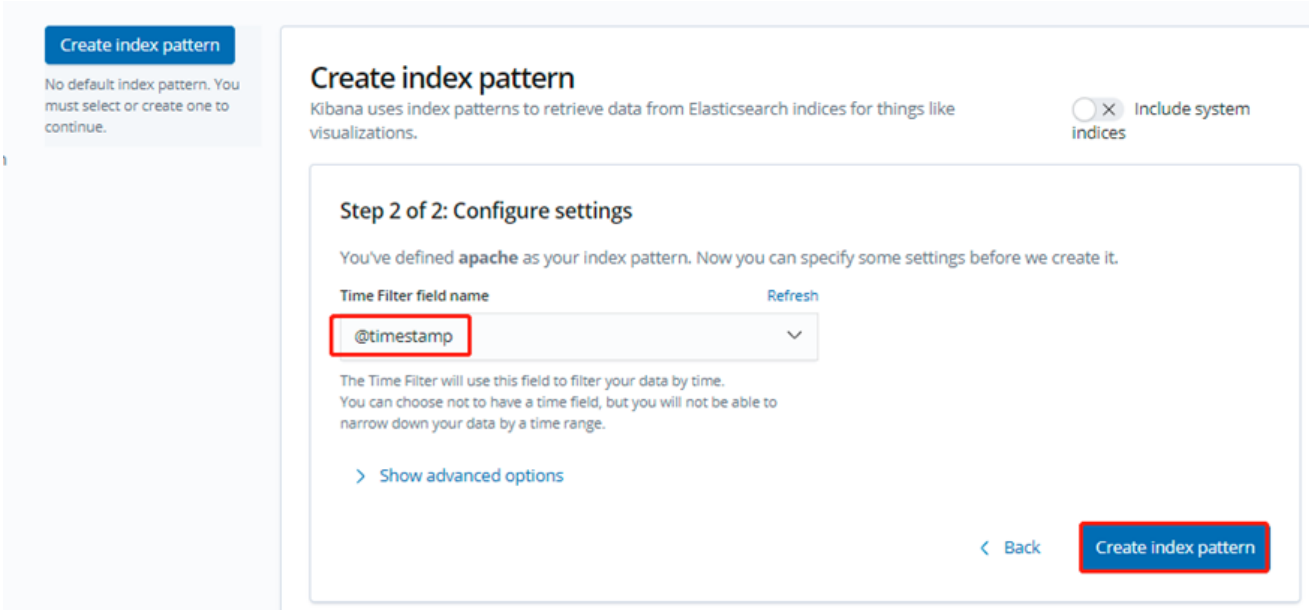
1）、设置索引。点击Management，单击”Index Patterns”



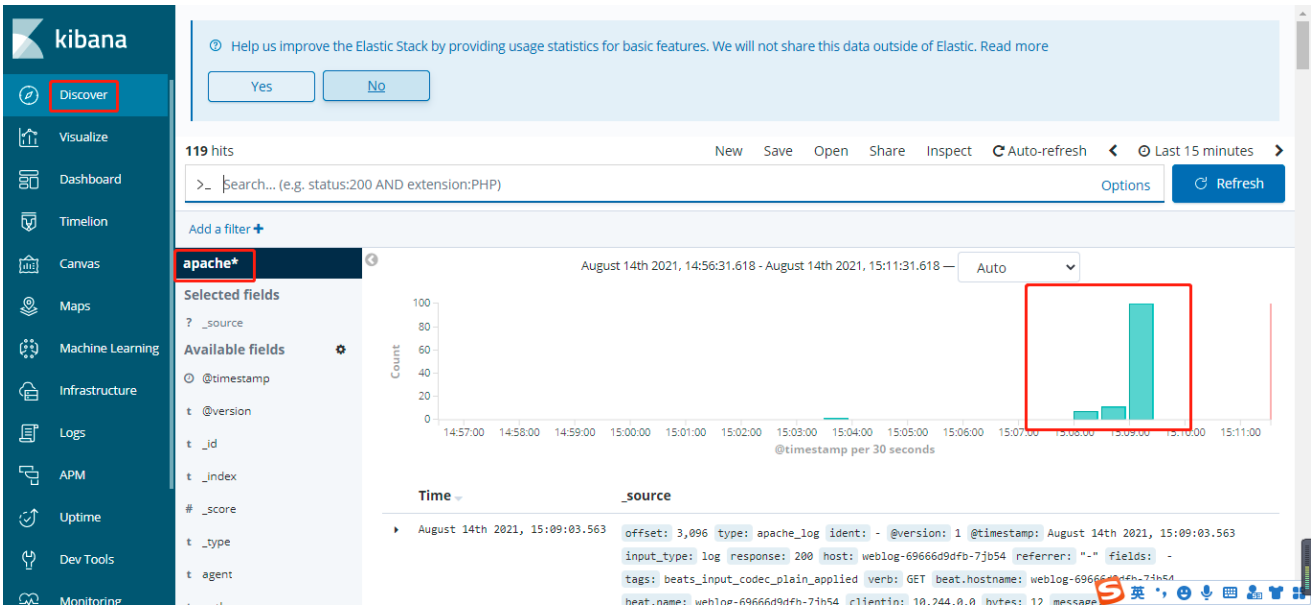
2）、设置Index name，可以采用通配符



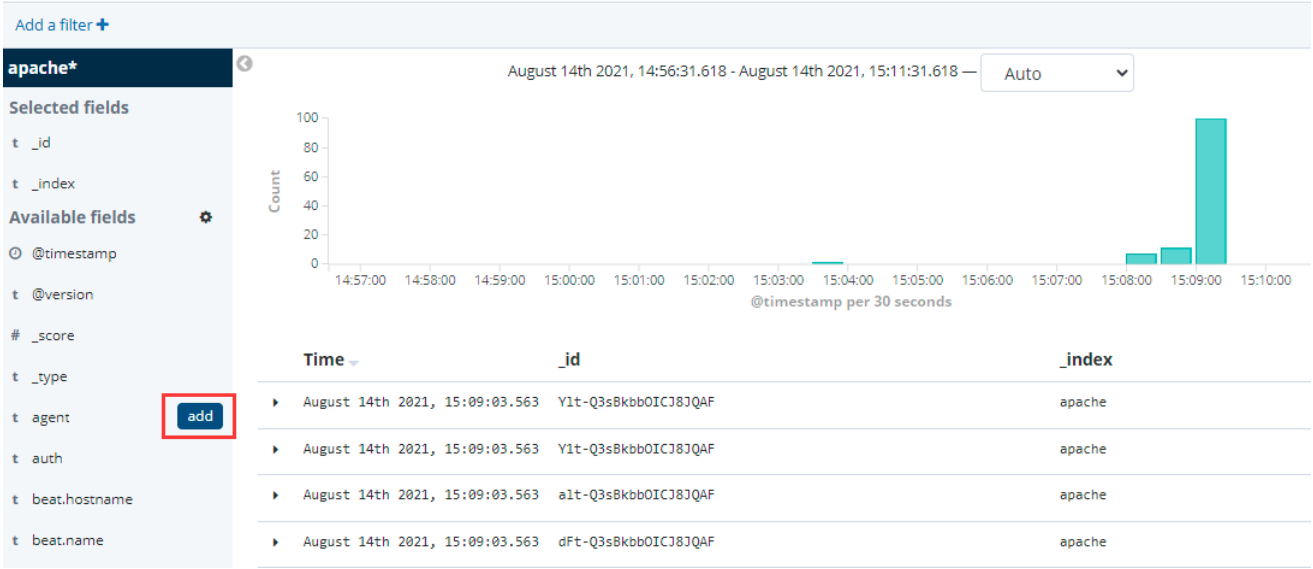
3）、时间字段选择@timestamp，选择完成后单击“Create Index pattern ”

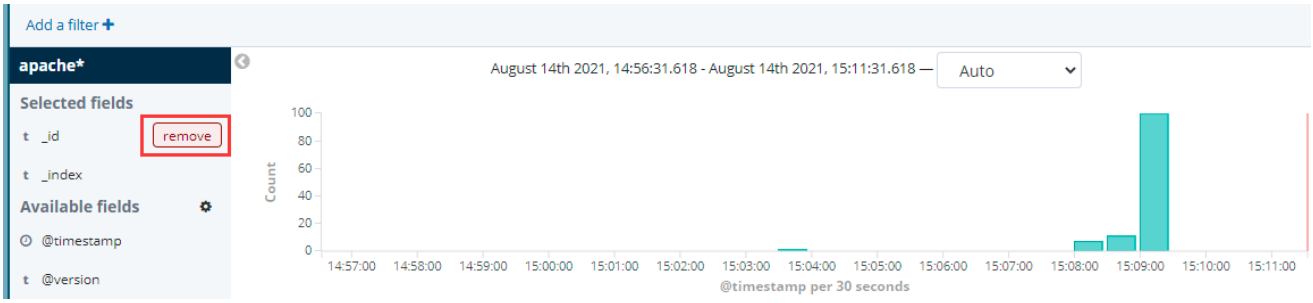


4)、单击Discover等待Searching完成后，可以看到数据

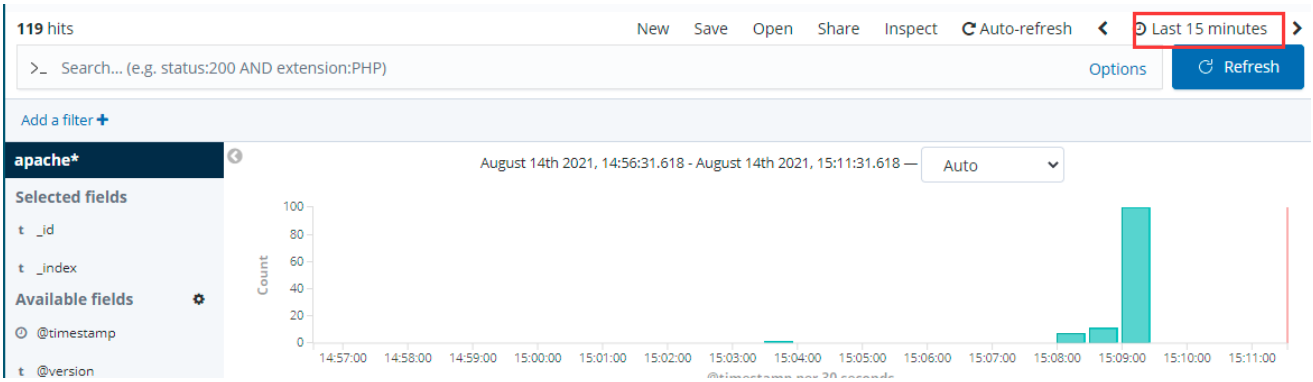


有时候只想关注一些指定的字段，那么可以将鼠标移动到索引下面的字段上，然后选add即可。同样的移动上面已经选择的字段选择remove进行移除

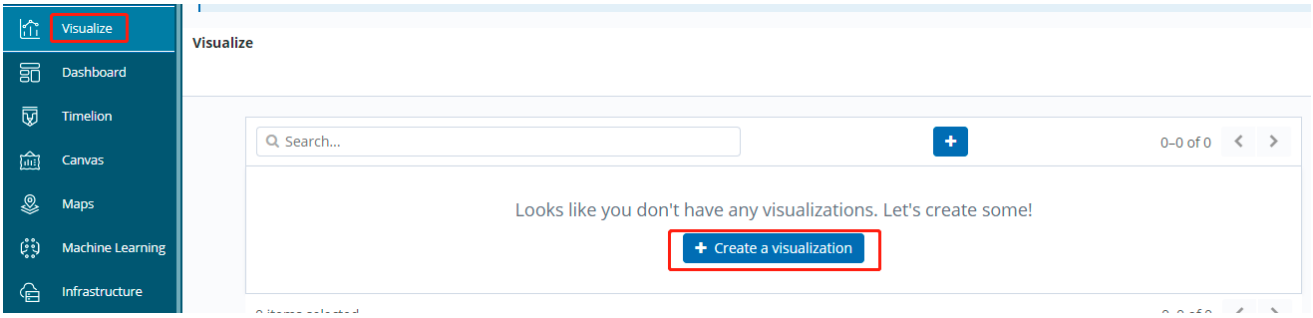


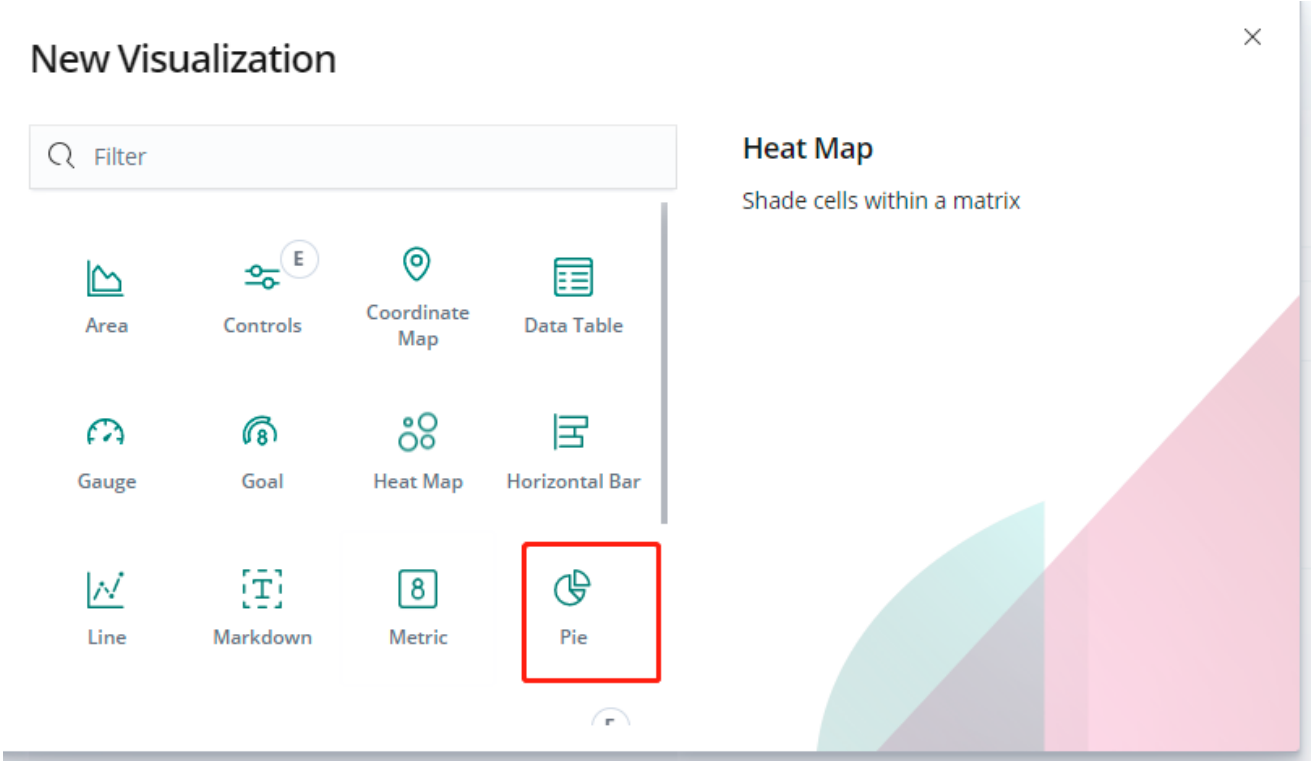


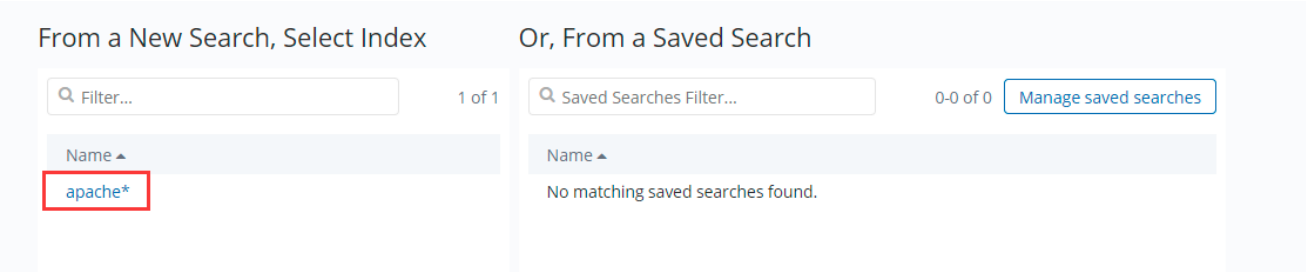
以表格的形式进行展示，同时可选择什么时间段内的数据，以及数据刷新时间

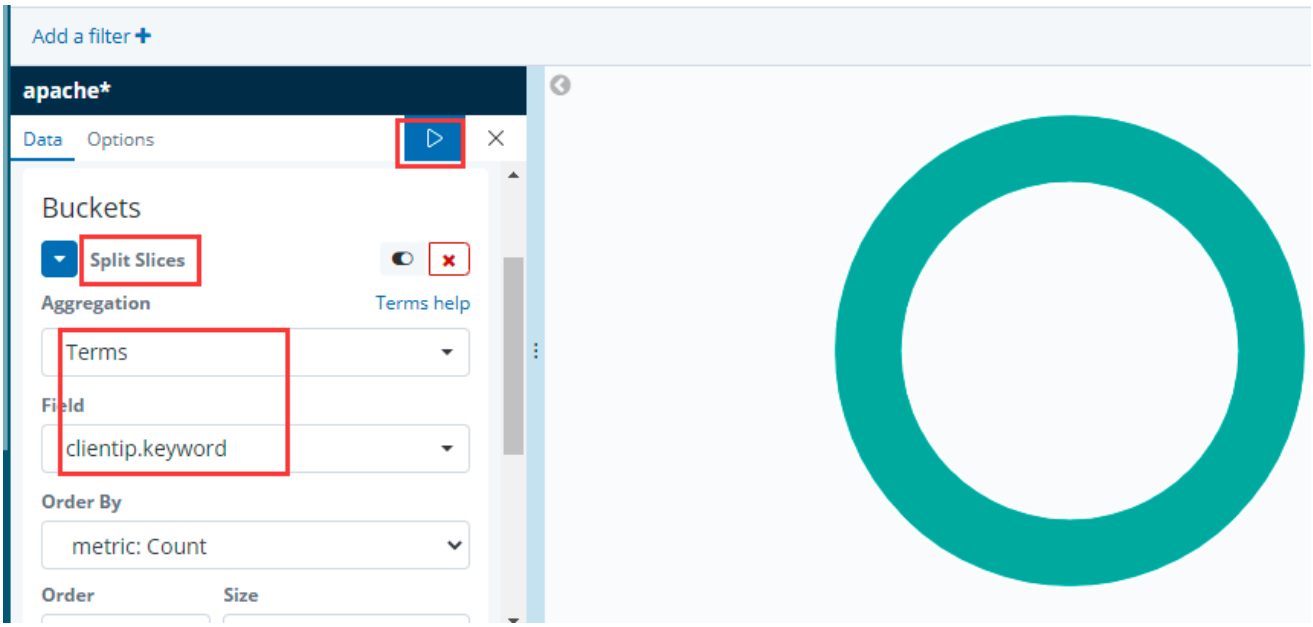


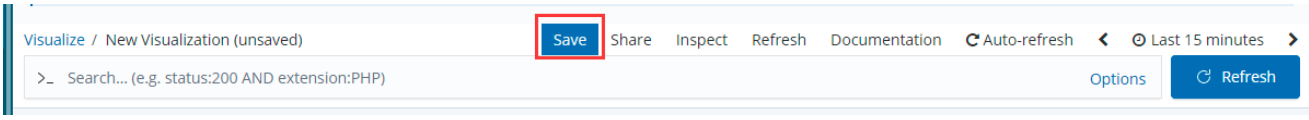
创建可视化图表

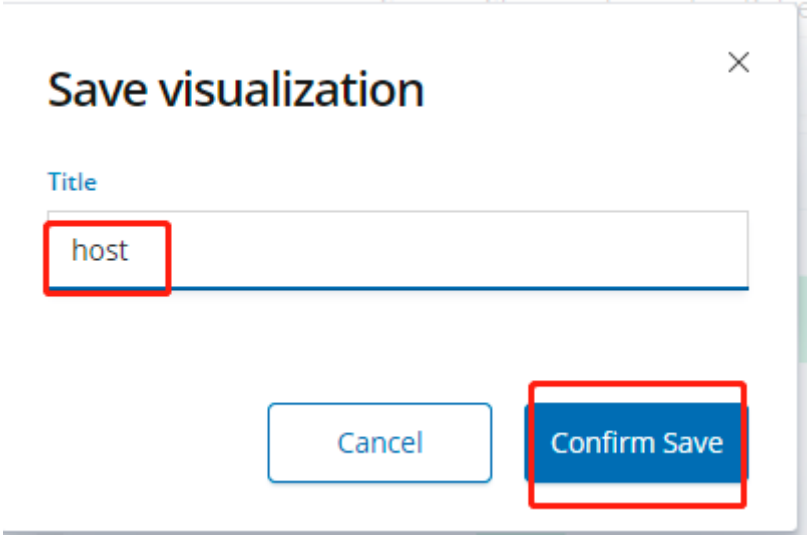




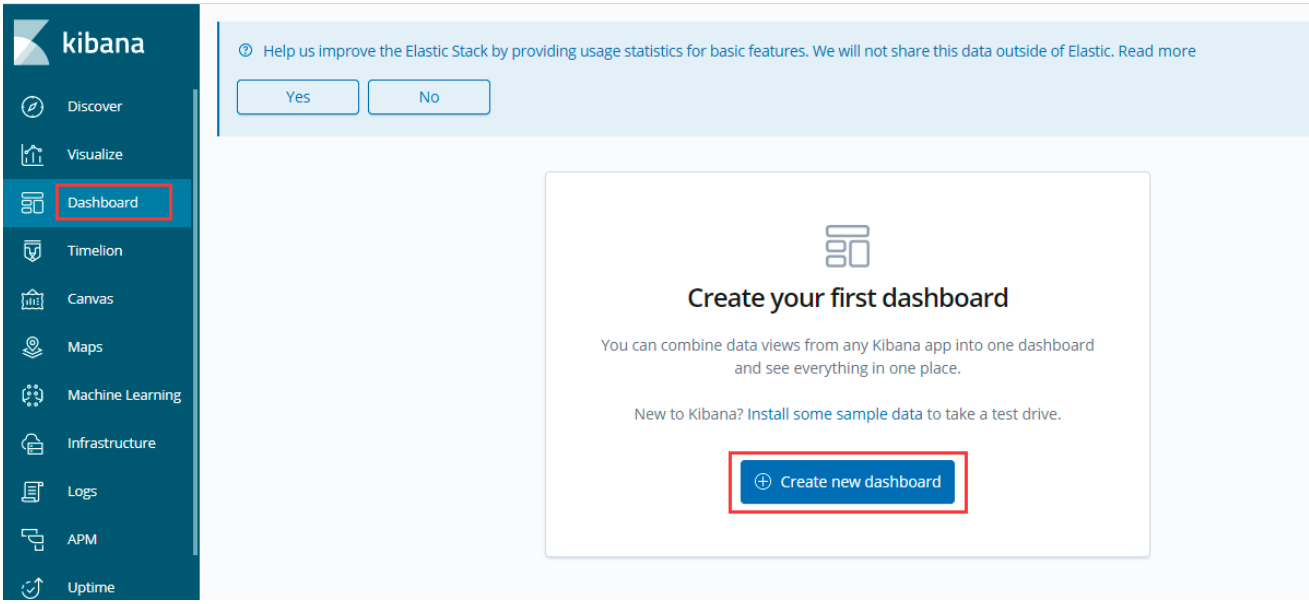




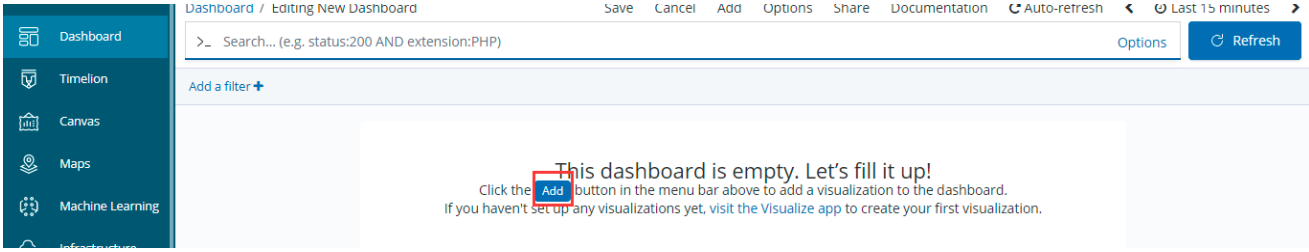


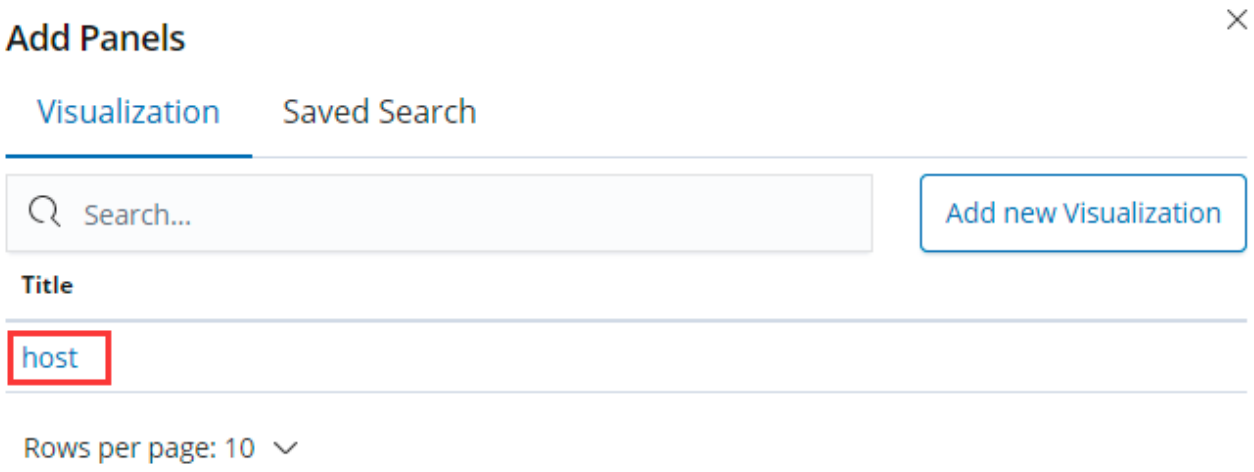


2）、点击Dashboard，会有很多仪表盘。在搜索框里写入关键词host，会出现和搜索关键词相关的表盘。可以通过它进行数据展示



点击add添加

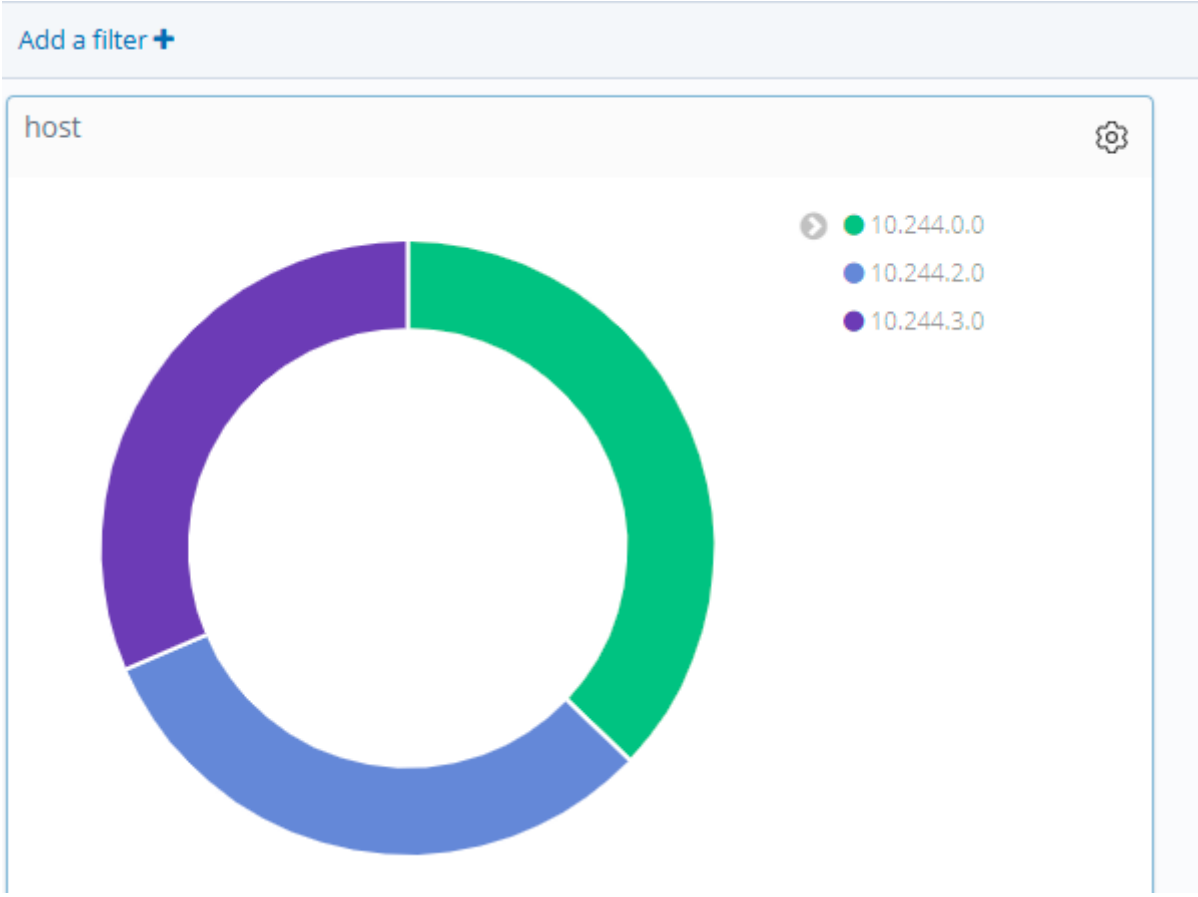




访问测试,编写脚本，用于每台机器随机访问网站测试，将脚本内容复制到 es1-es5 命令直接运行

```
[root@es1 ~]# vim test.sh
#!/bin/bash
K=$((RANDOM%90+10))                          #定义变量，得到10~100的随机数
for ((i=1;i<=K;i++));do
 curl -A "${HOSTNAME}" -s http://10.244.3.4/        #访问网站，-A 指定谁去访问网站
done
echo ${K}            #输出访问次数
```

在kibana 中查看饼图的变化


Add a filter +
host
10.244.0.0
10.244.2.0
10.244.3.0