生产日志管理系统部署实战

目录

文档信息

文档作者	房佳亮
文档版本	Version1.0
文档版权	内部资料禁止传播
文档归类	Linux 运维架构师系列
系统环境	CentOS-7.X-x86_64
作者邮箱	crushlinux@163.com
修订信息	2020-06-30
技术交流	

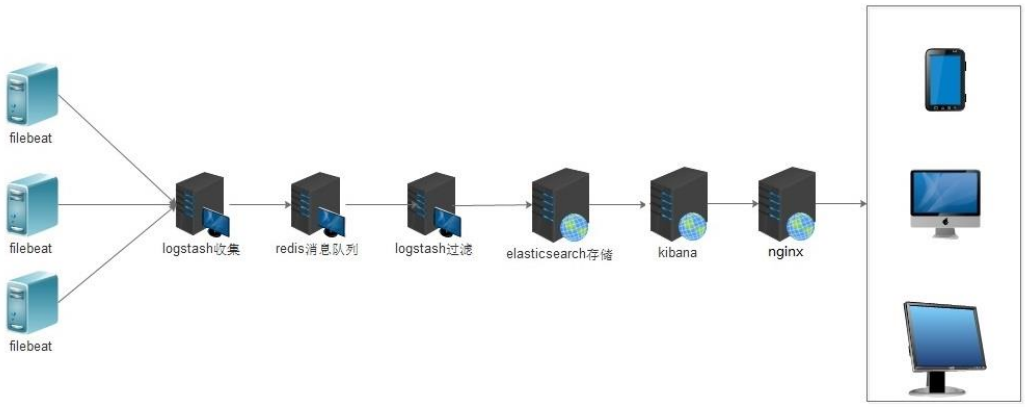
文档约定

[绿色背景] 知识重点
[红色背景] 错误警告
[黄色背景] 注意事项

执行命令

一、架构设计

拓扑结构



主机规划

操作系统	IP 地址	角色	CPU&内存	主机名
CentOS7.X-x86_64	192.168.200.111	ELK+Head+Nginx	2C4G	elk1
CentOS7.X-x86_64	192.168.200.112	ES	2C4G	elk2
CentOS7.X-x86_64	192.168.200.113	Redis	2C2G	redis
CentOS7.X-x86_64	192.168.200.114	Filebeat+Nginx	2C2G	web

二、系统初始化

2.1、关闭防火墙和 selinux

```
[root@localhost ~]# setenforce 0
[root@localhost ~]# iptables -F
[root@localhost ~]# systemctl stop firewalld
[root@localhost ~]# systemctl disable firewalld
```

2.2、关闭 NetworkManager 服务

```
[root@localhost ~]# systemctl stop NetworkManager
[root@localhost ~]# systemctl disable NetworkManager
```

2.3、配置阿里云 yum 源

```
[root@localhost ~]# wget http://mirrors.aliyun.com/repo/Centos-7.repo -O
/etc/yum.repos.d/aliyun.repo
```

2.4、配置主机名

```
[root@localhost ~]# hostname elk1    #各个主机根据表格配置对应的主机名
[root@localhost ~]# bash
[root@elk1 ~]#
```

2.5、配置 hosts 映射

```
[root@elk1 ~]# vim /etc/hosts
192.168.200.111 elk1
192.168.200.112 elk2
192.168.200.113 redis
192.168.200.114 web
[root@elk1 ~]# scp /etc/hosts 192.168.200.112:/etc/
[root@elk1 ~]# scp /etc/hosts 192.168.200.113:/etc/
[root@elk1 ~]# scp /etc/hosts 192.168.200.114:/etc/
```

三、安装部署 Elasticsearch 集群

3.1、检查 java 环境

在 elk1 和 elk2 两台主机上进行

```
[root@elk1 ~]# java -version
openjdk version "1.8.0_161"
OpenJDK Runtime Environment (build 1.8.0_161-b14)
OpenJDK 64-Bit Server VM (build 25.161-b14, mixed mode)
```

3.2、下载 elasticsearch 二进制安装包并解压安装

在 elk1 和 elk2 两台主机上进行

```
[root@elk1 ~]# wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.4.0-
linux-x86_64.tar.gz
[root@elk1 ~]# tar xf elasticsearch-7.4.0-linux-x86_64.tar.gz -C /usr/local/
[root@elk1 ~]# cd /usr/local/
[root@elk1 local]# mv elasticsearch-7.4.0/ elasticsearch
```

3.3、修改 elk1 配置文件

```
[root@elk1 ~]# vim /usr/local/elasticsearch/config/elasticsearch.yml
17 cluster.name: elk-cluster                    #定义集群名字，两台主机保持一致
23 node.name: elk1                              #定义当前节点名字
24 node.master: true                            #定义当前节点为主节点，否则为 false
34 path.data: /usr/local/elasticsearch/data     #定义数据路径
38 path.logs: /usr/local/elasticsearch/logs     #定义日志路径
56 network.host: 0.0.0.0                            #定义监听地址
60 http.port: 9200                              #定义监听端口
69 discovery.seed_hosts: ["elk1", "elk2"]       #定义集群节点
73 cluster.initial_master_nodes: ["elk1"]       #定义当前为主节点
```

3.4、修改 elk2 配置文件

```
[root@elk2 local]# vim /usr/local/elasticsearch/config/elasticsearch.yml
17 cluster.name: elk-cluster
23 node.name: elk2
24 node.master: false
34 path.data: /usr/local/elasticsearch/data
38 path.logs: /usr/local/elasticsearch/logs
56 network.host: 0.0.0.0
60 http.port: 9200
69 discovery.seed_hosts: ["elk1", "elk2"]
73 cluster.initial_master_nodes: ["elk1"]
```

3.5、启动 ES 集群

```
[root@elk1 ~]# useradd -s /sbin/nologin -r elk
[root@elk1 ~]# chown -R elk /usr/local/elasticsearch/
[root@elk1 ~]# sudo -u elk /usr/local/elasticsearch/bin/elasticsearch -d

[root@elk1 ~]# echo "vm.max_map_count=262144" >> /etc/sysctl.conf
[root@elk1 ~]# sysctl -p

[root@elk1 ~]# netstat -lnpt | grep java
tcp6       0      0 :::9200                 :::*                    LISTEN
64052/java
tcp6       0      0 :::9300                 :::*                    LISTEN
64052/java
```

3.6、故障案例

max file descriptors [4096] for elasticsearch process is too low, increase to at least [65535]

解决方案：修改/etc/security/limits.conf 文件，增加以下两行配置，重新执行启动任务

```
[root@elk1 ~]# vim /etc/security/limits.conf
*       soft    nofile          65536
*       hard    nofile          65536
```

max virtual memory areas vm.max_map_count [65530] is too low, increase to at least [262144]

解决方案：修改/etc/sysctl.conf 文件，增加配置 vm.max_map_count=262144，保存退出后执行 sysctl -p 生效

max number of threads [3795] for user [elk] is too low, increase to at least [4096]

解决方案：修改配置文件/etc/security/limits.conf（和问题 1 是一个文件），增加以下两行配置，保存退出

```
*       soft    noproc          4096
*       hard    noproc          4096
```

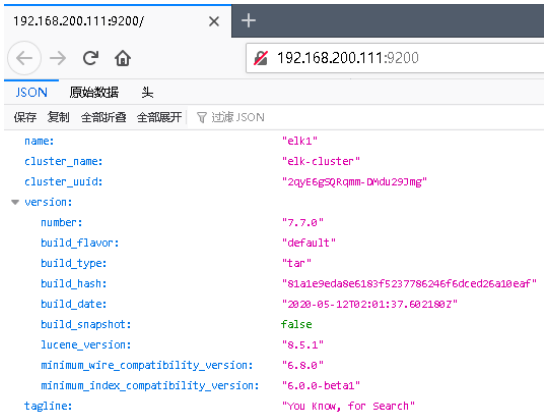
3.7、测试集群状态

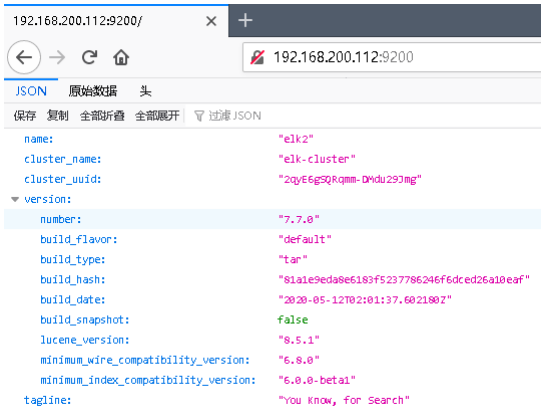
在 elk1 或 elk2 任意主机上执行以下命令

```
[root@elk1 ~]# curl http://192.168.200.111:9200/_cluster/health?pretty
{
  "cluster_name" : "elk-cluster",
  "status" : "green",
  "timed_out" : false,
  "number_of_nodes" : 2,
  "number_of_data_nodes" : 2,
  "active_primary_shards" : 0,
  "active_shards" : 0,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 0,
```

```
  "delayed_unassigned_shards" : 0,
  "number_of_pending_tasks" : 0,
  "number_of_in_flight_fetch" : 0,
  "task_max_waiting_in_queue_millis" : 0,
  "active_shards_percent_as_number" : 100.0
}
```

通过浏览器访问访问测试：http://192.168.200.111:9200 http://192.168.200.112:9200





四、安装部署 elasticsearch-head 插件

4.1、elk1 主机下载安装 nodejs 以获取 npm 模块

```
[root@elk1 ~]# wget https://nodejs.org/dist/v10.16.3/node-v10.16.3-linux-x64.tar.xz
[root@elk1 ~]# tar xf node-v10.16.3-linux-x64.tar.xz -C /usr/local/
[root@elk1 ~]# cd /usr/local/
[root@elk1 local]# mv node-v10.16.3-linux-x64/ nodejs
[root@elk1 local]# ln -s /usr/local/nodejs/bin/npm /bin/npm
[root@elk1 local]# ln -s /usr/local/nodejs/bin/node /bin/node
```

4.2、elk1 主机下载 elasticsearch-head 并安装

```
[root@elk1 ~]# wget https://github.com/mobz/elasticsearch-head/archive/master.zip
[root@elk1 ~]# unzip master.zip -d /usr/local
[root@elk1 ~]# cd /usr/local
[root@elk1 local]# mv elasticsearch-head-master/ es-head
[root@elk1 local]# cd es-head/
[root@elk1 es-head]# npm install -g grunt-cli     #如果报错可忽略此步骤
[root@elk1 es-head]# npm install
```

4.3、故障案例

错误 1：执行 npm install -g grunt-cli 时提示 /usr/bin/env: node: 不是目录

排除：在执行该命令时会同时调用 node 模块，而 node 模块调用的命令不在环境变量中，

故将 node 调用命令链接至环境变量即可：ln -s /usr/local/nodejs/bin/node /bin/node

错误 2：执行 npm install 时出现多个 "ERR"
排除方法 1：出现该原因是因为缺省了相应的类库，执行命令 npm install phantomjs-prebuilt -save 安装缺少的类库，随后重新执行前项任务；
排除方法 2：执行 npm install phantomjs-prebuilt@2.1.16 --ignore-script 忽略相关类库，直接进行安装。

解决依赖重新安装

```
[root@elk1 es-head]# npm install phantomjs-prebuilt@2.1.16 --ignore-script
[root@elk1 es-head]# npm install
```

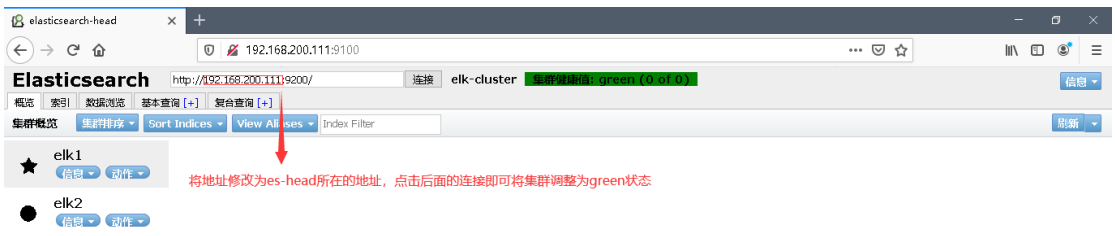
4.4、配置 elasticsearch 与 es-head 软件关联并重启
在 elk1 和 elk2 两台主机上进行

```
[root@elk1 ~]# vim /usr/local/elasticsearch/config/elasticsearch.yml
http.cors.enabled: true
http.cors.allow-origin: "*"
[root@elk1 ~]# pkill java
[root@elk1 ~]# sudo -u elk /usr/local/elasticsearch/bin/elasticsearch -d
```

4.5、启动 es-head
必须在 es-head 所在的安装目录下执行以下启动命令

```
[root@elk1 ~]# cd /usr/local/es-head/
[root@elk1 es-head]# nohup npm run start &
[root@elk1 es-head]# netstat -lnpt | grep 9[1-3]00
tcp        0      0 0.0.0.0:9100            0.0.0.0:*               LISTEN
65061/grunt
tcp6       0      0 :::9200                 :::*                    LISTEN
64941/java
tcp6       0      0 :::9300                 :::*                    LISTEN
64941/java
```

4.6、es-head 测试
通过浏览器访问测试： http://192.168.200.111:9100



五、安装部署 logstash 进行测试

5.1、下载安装 logstash

```
[root@elk1 ~]# wget https://artifacts.elastic.co/downloads/logstash/logstash-7.4.0.rpm
[root@elk1 ~]# rpm -ivh logstash-7.4.0.rpm
```

5.2、修改 logstash 的配置文件

```
[root@elk1 ~]# vim /etc/logstash/logstash.yml
19 node.name: logstash                  #定义 logstash 服务器对其他服务器可见名称
28 path.data: /var/lib/logstash         #定义数据目录
64 path.config: /etc/logstash/conf.d/   #定义子文件目录
290 http.host: "192.168.200.111"        #定义 xpack 服务器使用的地址
208 path.logs: /var/log/logstash        #定义日志目录
```

5.3、启动测试

将标准输入作为标准输出进行测试

```
[root@elk1 ~]# /usr/share/logstash/bin/logstash -e 'input {stdin {}} output {stdout {}}'
[INFO ] 2020-05-30 02:14:58.632 [Api Webserver] agent - Successfully started Logstash API
endpoint {:port=>9600}
www.crushlinux.com    #标准输入和标准输出
{
      "@timestamp" => 2020-05-29T18:15:09.507Z,
          "message" => "www.crushlinux.com",
        "@version" => "1",
              "host" => "elk1"
}
www.elk.com             #标准输入和标准输出
{
      "@timestamp" => 2020-05-29T18:15:20.742Z,
          "message" => "www.elk.com",
        "@version" => "1",
              "host" => "elk1"
}
```

测试成功，等待数据接入。

六、安装部署 kibana 并测试

6.1、下载安装 kibana

```
[root@elk1 ~]# wget https://artifacts.elastic.co/downloads/kibana/kibana-7.4.0-x86_64.rpm
[root@elk1 ~]# rpm -ivh kibana-7.4.0-x86_64.rpm
```

6.2、修改 kibana 配置文件

```
[root@elk1 ~]# vim /etc/kibana/kibana.yml
2 server.port: 5601                          #定义监听端口
7 server.host: "0.0.0.0"                     #定义监听地址
25 server.name: "kibana"                     #定义对外显示的服务名称
28 elasticsearch.hosts: ["http://192.168.200.111:9200"]  #定义 ES 服务器信息
37 kibana.index: ".kibana"                   #定义自身默认索引页
46 elasticsearch.username: "kibana"  #定义验证信息，安装 xpack 插件后才有效
47 elasticsearch.password: "123"
97 logging.dest: /var/log/kibana.log  #定义日志位置
115 i18n.locale: "zh-CN"                     #定义显示语言，7.x 系列直接支持中文
```

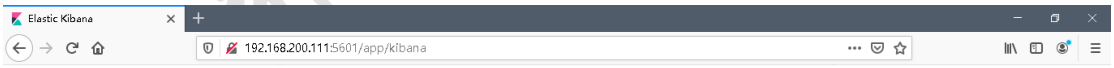
kibana 软件安装时默认添加系统用户 kibana ，不自动创建日志文件
创建日志文件

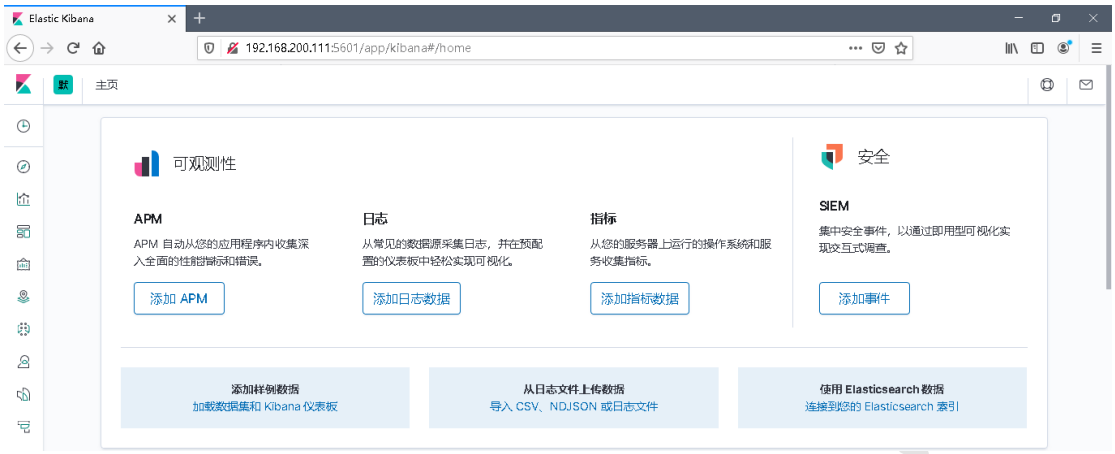
```
[root@elk1 ~]# id kibana
uid=986(kibana) gid=980(kibana)  组=980(kibana)
[root@elk1 ~]# touch /var/log/kibana.log
[root@elk1 ~]# chown kibana.kibana /var/log/kibana.log
[root@elk1 ~]# ll /var/log/kibana.log
-rw-r--r-- 1 kibana kibana 2318963 6 月    26 00:55 /var/log/kibana.log
```

6.3、启动服务

```
[root@elk1 ~]# systemctl start kibana
[root@elk1 ~]# netstat -lnpt | grep 5601
tcp        0      0 0.0.0.0:5601            0.0.0.0:*               LISTEN
65699/node
```

6.4、web 访问测试
通过浏览器访问测试： http://192.168.200.111:5601





6.5、通过 es-head 查看 es 集群与 kibana 的连接性

浏览器访问 es-head 节点服务器 IP 及对应端口

测试成功，等待数据接入

七、安装部署 redis

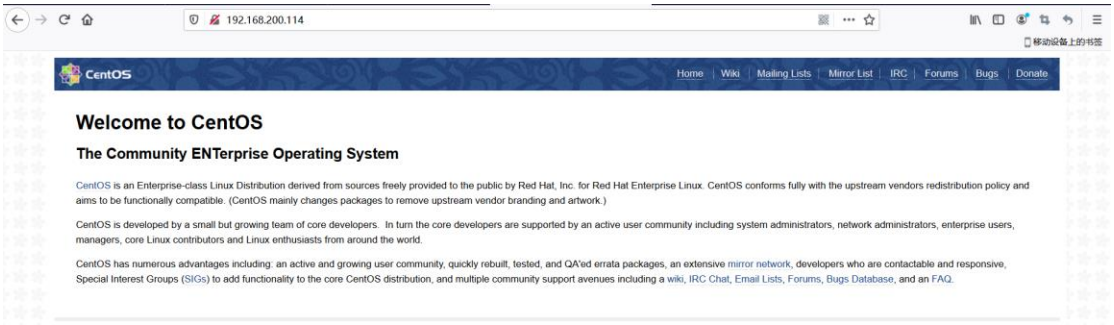
```
[root@redis ~]# rpm -ivh epel-release-latest-7.noarch.rpm
安装 redis
[root@redis ~]# yum -y install redis
[root@redis ~]# vim /etc/redis.conf
   61 bind 0.0.0.0
128 daemonize yes
480 requirepass 123
[root@redis ~]# systemctl start redis
[root@redis ~]# netstat -lnpt | grep redis
tcp        0      0 0.0.0.0:6379            0.0.0.0:*               LISTEN
63655/redis-server
```

八、安装部署 Nginx

```
[root@web ~]# rpm -ivh epel-release-latest-7.noarch.rpm
[root@web ~]# yum -y install nginx
```

```
[root@web ~]# systemctl start nginx
[root@web ~]# netstat -lnpt | grep :80
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
63626/nginx: master
tcp6       0      0 :::80                   :::*                    LISTEN
63626/nginx: master
```

浏览器访问 nginx 服务器 IP 地址



九、安装部署 filebeat

```
[root@web ~]# wget https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.4.0-
x86_64.rpm

[root@web ~]# rpm -ivh filebeat-7.4.0-x86_64.rpm
[root@web ~]# systemctl restart filebeat
[root@web ~]# ps -ef |grep filebeat
root       63691       1    0 16:21 ?        00:00:00 /usr/share/filebeat/bin/filebeat -
environment systemd -c /etc/filebeat/filebeat.yml
-path.home /usr/share/filebeat -path.config /etc/filebeat -path.data /var/lib/filebeat -
path.logs /var/log/filebeatroot       63712    1724    0 16:22 pts/0    00:00:00 grep --
color=auto filebeat
```

Filebeat 是没有指定的端口的，filebeat 通过 post 方式将日志推送至指定的 output 对象。但是当 filebeat 与传输对象建立连接时，其推送端口与接收日志的服务器端口保持一致。

通过 filebeat 将 nginx 的访问日志传输给 elasticsearch 集群进行处理

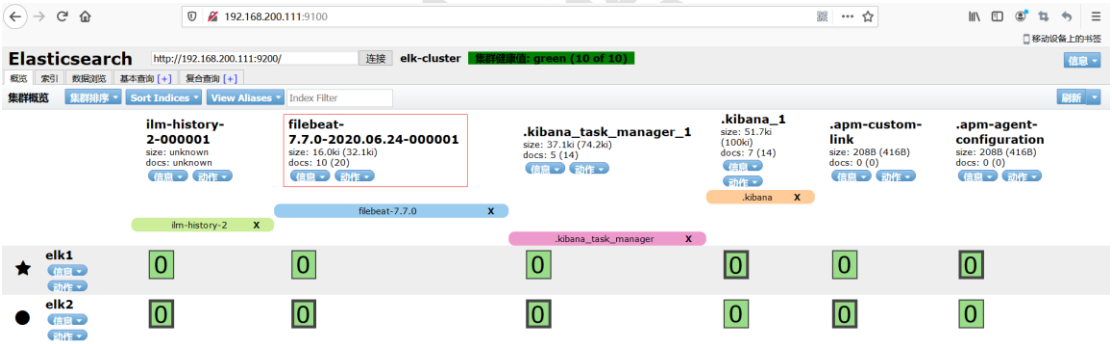
9.1、修改 filebeat 配置文件，将 nginx 访问日志输出到 es 集群

```
[root@web ~]# vim /etc/filebeat/filebeat.yml
[root@web ~]# grep -Ev "#|^$" /etc/filebeat/filebeat.yml
filebeat.inputs:      #数据输入源
- type: log           #数据类型，关于数据类型信息可以查看/etc/filebeat/field.yml 文件
  enabled: true       #将默认的 false 修改为 true
  paths:
    - /var/log/nginx/access.log   #数据源输入的文件
```

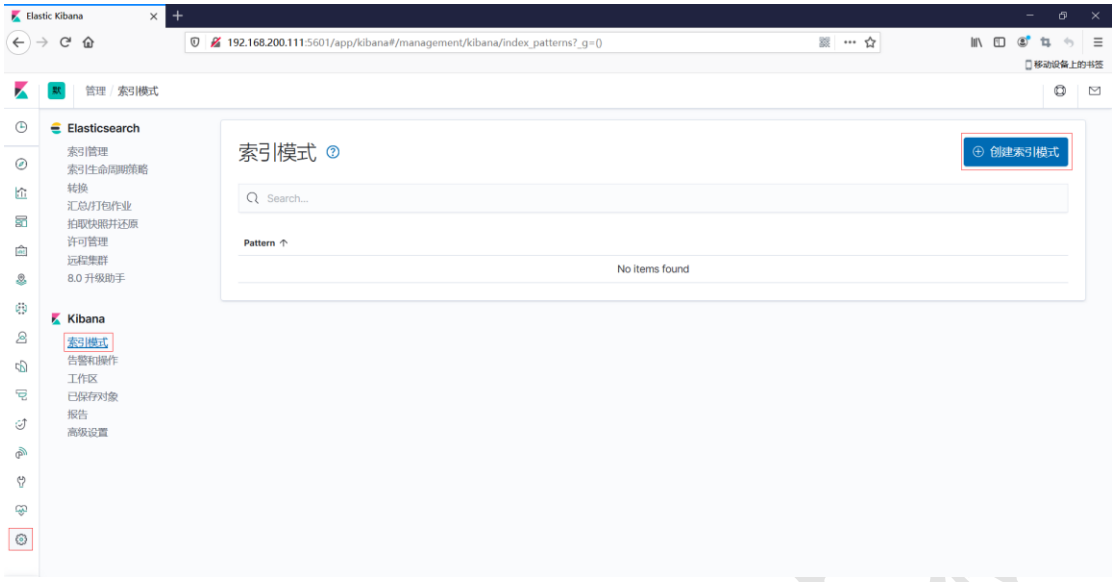
```
filebeat.config.modules:                    #默认的配置模块
  path: ${path.config}/modules.d/*.yml
  reload.enabled: false
setup.template.settings:
  index.number_of_shards: 1      #7.x 版本以后，默认分片从原来的 5 改为 1，可以在此修改
setup.kibana:                               #filebeat 也可以在配置文件中 kibana 做相应设置，具体自行探索
output.elasticsearch:                       #数据输出
  hosts: ["192.168.200.111:9200"]     #数据输出服务器及端口
processors:                                 #默认添加的元数据
  - add_host_metadata: ~
  - add_cloud_metadata: ~

[root@web ~]# systemctl restart filebeat
[root@web ~]# netstat -anpt | grep :9200
tcp        0      0 192.168.200.114:43462   192.168.200.111:9200    ESTABLISHED 63964/filebeat
```

9.2、测试 filebeat 与 es 集群的连接

查看 es-head 的 web 界面 http://192.168.200.111:9100，可以发现以 filebeat 命名的索引页

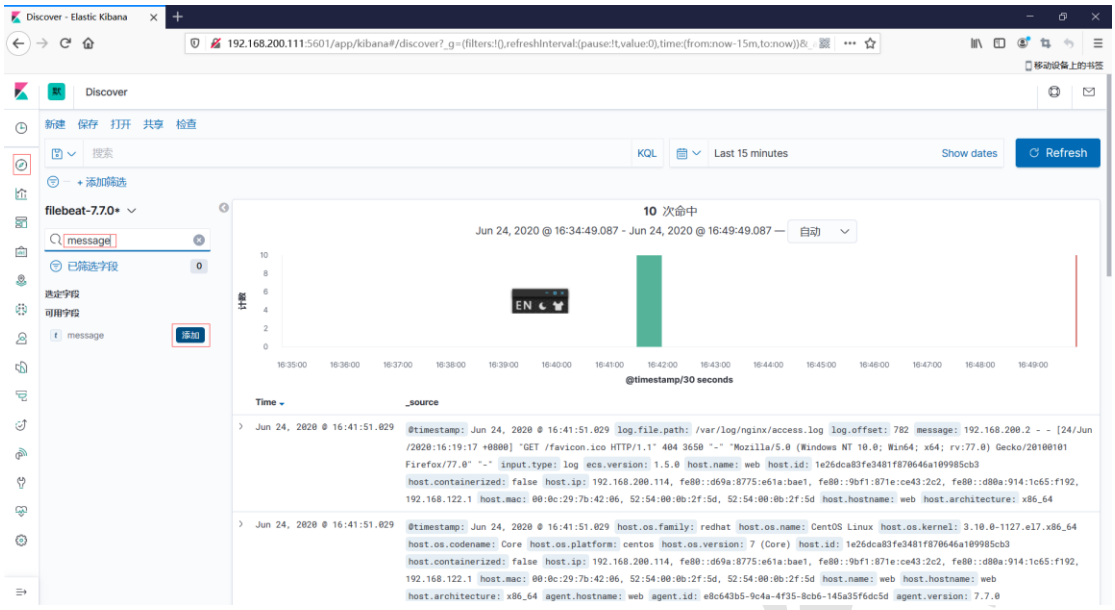


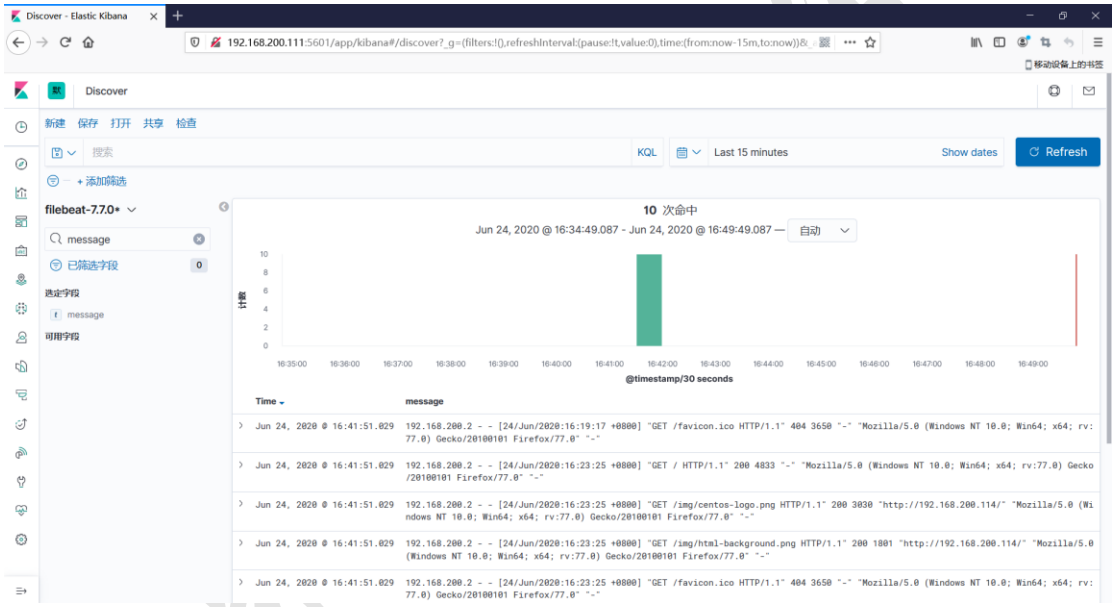
9.3、通过 kibana 建立 kibana 索引


Elastic Kibana
192.168.200.111:5601/app/kibana#/management/kibana/index_patterns?_g=()
管理 / 索引模式
Elasticsearch
索引管理
索引生命周期策略
转换
汇总/打包作业
拍取快照并还原
许可管理
远程集群
8.0 升级助手
Kibana
索引模式
告警和操作
工作区
已保存对象
报告
高级设置
索引模式
创建索引模式
Search...
Pattern
No items found


Elastic Kibana
192.168.200.111:5601/app/kibana#/management/kibana/index_pattern?_g=()
管理 / 索引模式 / 创建索引模式
Elasticsearch
索引管理
索引生命周期策略
转换
汇总/打包作业
拍取快照并还原
许可管理
远程集群
8.0 升级助手
Kibana
索引模式
告警和操作
工作区
已保存对象
报告
高级设置
创建 索引模式
Kibana 使用索引模式从 Elasticsearch 索引中检索数据，以实现诸如可视化等功能。
包括系统索引
第 1 步（共 2 步）：定义索引模式
索引模式
filebeat-7.7.0*
可以在索引模式中将 * 用作通配符。
不能使用空格或字符 \, /, ?, ", <, >, |。
成功！ 您的索引模式匹配 1 个索引。
filebeat-7.7.0-2020.06.24-000001
每页行数: 10
下一步


Elastic Kibana
192.168.200.111:5601/app/kibana#/management/kibana/index_pattern?_g=()
管理 / 索引模式 / 创建索引模式
Elasticsearch
索引管理
索引生命周期策略
转换
汇总/打包作业
拍取快照并还原
许可管理
远程集群
8.0 升级助手
Kibana
索引模式
告警和操作
工作区
已保存对象
报告
高级设置
创建 索引模式
Kibana 使用索引模式从 Elasticsearch 索引中检索数据，以实现诸如可视化等功能。
包括系统索引
第 2 步（共 2 步）：配置设置
您已将 "filebeat-7.7.0*" 定义为 "索引模式"。现在，在我们创建之前，您可以指定一些设置。
时间筛选字段名称
@timestamp
显示高级选项
上一步
创建索引模式





9.4、测试 filebeat 通过 logstash 传输日志给 es 集群（json 插件的应用）

修改 filebeat 配置文件

```
[root@web ~]# vim /etc/filebeat/filebeat.yml
148 output.logstash:     #修改日志输出对象
150     hosts: ["192.168.200.111:5044"]     #日志输出的对象服务器 IP 及端口
```

定义 logstash 文件

```
[root@elk1 ~]# vim /etc/logstash/conf.d/nginx_grok.conf
input {
      beats {    #信息源为 filebeat
            port => 5044   #数据传输端口
            }
      }
```

```
filter {
      json {      #json 模块
            source => "message"     #模块作用于 massage 字段
            }
      }
output {
      elasticsearch {       #输出对象
            hosts => ["192.168.200.111:9200"]      #输出对象 IP 及端口
            index => "nginx_access.log"                #输出时的索引名称
      }
}
```

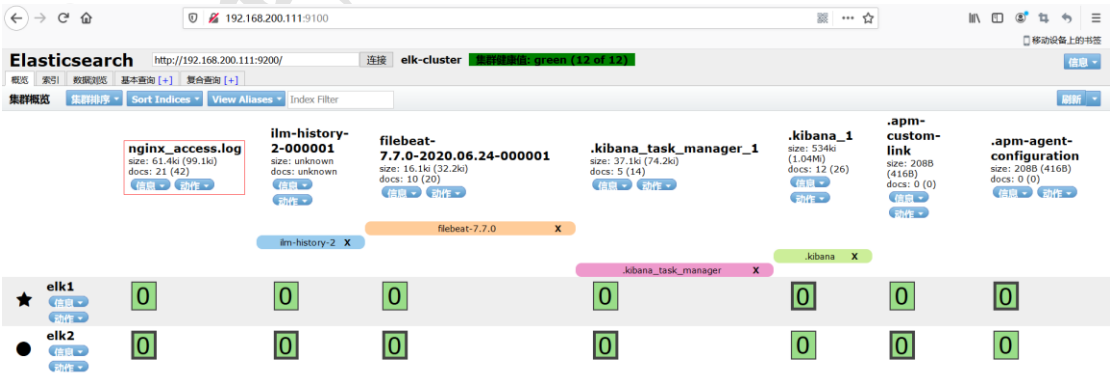
```
[root@web ~]# vim /etc/nginx/nginx.conf
      log_format    main    '$remote_addr - $remote_user [$time_local] "$request" '
                              '$status $body_bytes_sent "$http_referer" '
                              '"$http_user_agent" "$http_x_forwarded_for"';

      log_format json '{ "@timestamp": "$time_iso8601", '
            '"remote_addr": "$remote_addr", '
            '"remote_user": "$remote_user", '
            '"body_bytes_sent": "$body_bytes_sent", '
            '"request_time": "$request_time", '
            '"status": "$status", '
            '"request_uri": "$request_uri", '
            '"request_methed": "$request_uri",'
            '"http_referer": "$http_referer", '
            '"http_x_forwarded_for": "$http_x_forwarded_for", '
            '"http_user_agent": "$http_user_agent"}';

      access_log    /var/log/nginx/access.log    json;
```

nginx 变量说明

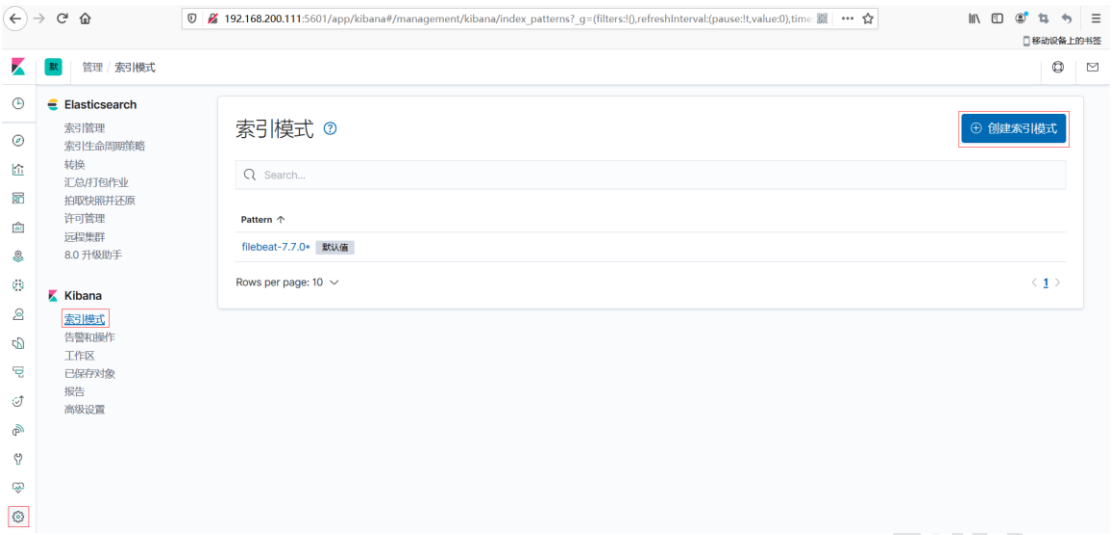
日志变量	含义
$request_method	表示 HTTP 请求方法，通常为"GET"或"POST"。
$request_uri	表示客户端请求参数的原始 URI，它无法修改。
$status	表示请求状态（常见状态码有：200 表示成功，404 表示页面不存在，301 表示永久重定向等）。
$http_referer	表示来源页面，即从哪个页面请求过来的，这个专业名称叫做“referer”，直接访问的话值为空。
$body_bytes_sent	表示发送给客户端的字节数，不包括响应头的大小。
$request_time	表示请求处理时间，单位为秒，精度毫秒； 从读入客户端的第一个字节开始，直到把最后一个字符发送给客户端后到日志写入为止。
$http_user_agent	表示用户浏览器信息，例如浏览器版本、浏览器类型等。
$bytes_sent	表示传输给客户端的字节数。
$server_addr	表示服务器端地址。
$server_name	表示请求到达的服务器对应的服务器名。
$server_port	表示请求到达的服务器对应的服务器端口。
$http_host	表示请求地址，即浏览器中输入的地址（IP 或域名）。
$request_filename	表示当前请求的文件的路径名。
$args	表示请求 URI 地址中的参数值。
$uri	表示请求中的当前 URI(不带请求参数，请求参数位于$args 中)。

重启 nginx、filebeat、logstash 进行验证

```
[root@web ~]# systemctl restart nginx
[root@web ~]# systemctl restart filebeat

[root@elk1 ~]# /usr/share/logstash/bin/logstash --path.settings /etc/logstash/ -f
/etc/logstash/conf.d/nginx_grok.conf &
[root@elk1 ~]# netstat -lnpt | grep 5044
tcp6       0      0 :::5044                 :::*                    LISTEN
65970/java
```

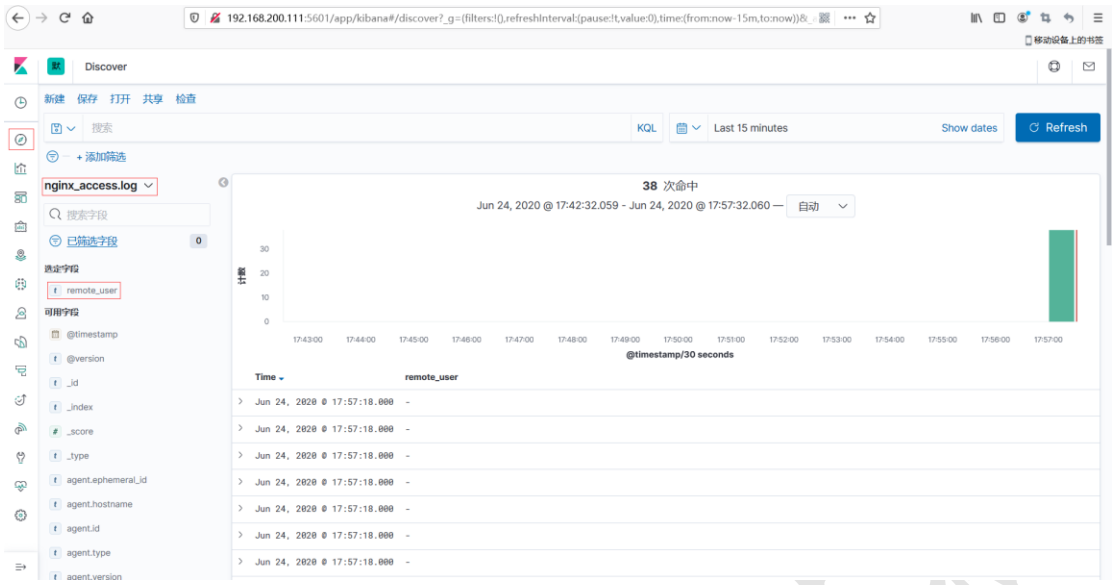


在 Kibana 中添加索引


192.168.200.111:5601/app/kibana#/management/kibana/index_patterns?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time
管理 / 索引模式
Elasticsearch
索引管理
索引生命周期策略
转换
汇总/打包作业
拍取快照并还原
许可管理
远程集群
8.0 升级助手
Kibana
索引模式
告警和操作
工作区
已保存对象
报告
高级设置
索引模式
创建索引模式
Search...
Pattern
filebeat-7.7.0*
默认值
Rows per page: 10
1


192.168.200.111:5601/app/kibana#/management/kibana/index_pattern?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time
管理 / 索引模式 / 创建索引模式
Elasticsearch
索引管理
索引生命周期策略
转换
汇总/打包作业
拍取快照并还原
许可管理
远程集群
8.0 升级助手
Kibana
索引模式
告警和操作
工作区
已保存对象
报告
高级设置
创建 索引模式
Kibana 使用索引模式从 Elasticsearch 索引中检索数据，以实现诸如可视化等功能。
包括系统索引
第 1 步（共 2 步）：定义索引模式
索引模式
nginx_access.log
可以在索引模式中将 * 用作通配符。
不能使用空格或字符 \, /, ?, ", <, >, |。
成功！ 您的索引模式匹配 1 个索引。
nginx_access.log
每页行数：10
下一步


192.168.200.111:5601/app/kibana#/management/kibana/index_pattern?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time
管理 / 索引模式 / 创建索引模式
Elasticsearch
索引管理
索引生命周期策略
转换
汇总/打包作业
拍取快照并还原
许可管理
远程集群
8.0 升级助手
Kibana
索引模式
告警和操作
工作区
已保存对象
报告
高级设置
创建 索引模式
Kibana 使用索引模式从 Elasticsearch 索引中检索数据，以实现诸如可视化等功能。
包括系统索引
第 2 步（共 2 步）：配置设置
您已将 "nginx_access.log" 定义为 "索引模式"。现在，在我们创建之前，您可以指定一些设置。
时间筛选字段名称 刷新
@timestamp
显示高级选项
上一步
创建索引模式



十、结合 Redis 可视化 Nginx 访客地理位置

配置 geoip 与 grok 插件的应用

10.1、配置思路：

- Filebeat 搜集 nginx 访问日志传输给 Redis；
- 创建 logstash 子配置文件定义信息输入源、过滤及输出对象；
- es-head 验证全局连通性；
- kibana 建立可视化图像；

10.2、修改 nginx 配置文件

将之前测试的 json 形式的日志格式修改为原来的 main

```
[root@web ~]# vim /etc/nginx/nginx.conf
http {
    log_format   main   '$remote_addr - $remote_user [$time_local] "$request" '
                        '$status $body_bytes_sent "$http_referer" '
                        '"$http_user_agent" "$http_x_forwarded_for"';


    log_format json '{ "@timestamp": "$time_iso8601", '
        '"remote_addr": "$remote_addr", '
        '"remote_user": "$remote_user", '
        '"body_bytes_sent": "$body_bytes_sent", '
        '"request_time": "$request_time", '
        '"status": "$status", '
        '"request_uri": "$request_uri", '
        '"request_methed": "$request_uri",'
        '"http_referer": "$http_referer", '
        '"http_x_forwarded_for": "$http_x_forwarded_for", '
```

```
        '"http_user_agent": "$http_user_agent"}';

    access_log  /var/log/nginx/access.log  main;

[root@web ~]# systemctl restart nginx
[root@web ~]# netstat -lnpt | grep :80
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
68170/nginx: master
tcp6       0      0 :::80                   :::*                    LISTEN
68170/nginx: master
```

10.3、修改 filebeat 配置文件

```
[root@web ~]# vim /etc/filebeat/filebeat.yml
148 output.redis:   #输出对象
150     hosts: ["192.168.200.113:6379"]     #redis 服务器 IP 及端口
151     password: 123          #redis 安全验证密码
152     key: "nginx_redis"     #redis 数据库的 key 名
153     data_type: "list"      #数据类型
154     db: 0                  #使用的第一个数据库
[root@web ~]# systemctl restart filebeat

[root@redis ~]# systemctl restart redis
```

10.4、基于 IP 地址库过滤访客地理位置

安装 IP 地址库

```
[root@elk1 ~]# tar xf GeoLite2-City_20200513-latest.tar.gz
[root@elk1 ~]# cd GeoLite2-City_20200512/
[root@elk1 GeoLite2-City_20200512]# ls
COPYRIGHT.txt    GeoLite2-City.mmdb    LICENSE.txt    README.txt
[root@elk1 GeoLite2-City_20200512]# cp GeoLite2-City.mmdb /opt/
```

10.5、创建 logstash 子配置文件

```
[root@elk1 ~]# vim /etc/logstash/conf.d/nginx_redis.conf
input {
    redis {
        host => "192.168.200.113"
        port => "6379"
        password => "123"
        key => "nginx_redis"
        data_type => "list"
        db => 0
    }
}
```

```
filter {
      grok {
            match => {
                  "message" => "%{IPV4:remote_addr} - (%{USERNAME:remote_user}|-)
\[%{HTTPDATE:time_local}\] \"%{WORD:request_me
thod} %{URIPATHPARAM:request_uri}
HTTP/%{NUMBER:http_protocol}\" %{NUMBER:http_status} %{NUMBER:body_bytes_sent}
\"%{GREEDYDATA:http_referer}\" \"%{GREEDYDATA:http_user_agent}\"
\"(%{IPV4:http_x_forwarded_for}|-)\""            }
      }

      geoip {
            source => "remote_addr"
            target => "geoip"
            database => "/opt/GeoLite2-City.mmdb"
            add_field => ["[geoip][coordinates]", "%{[geoip][longitude]}"]
            add_field => ["[geoip][coordinates]", "%{[geoip][latitude]}"]
      }

      date {
            locale => "en"
            match => ["time_local", "dd/MM/yyyy:HH:mm:ss Z"]
      }

      mutate {
            convert => ["[geoip][coordinates]", "float"]
      }
}

output {
      elasticsearch {
            hosts => ["http://192.168.200.111:9200","http://192.168.200.112:9200"]
            index => "logstash-filebeat-redis-logs-escluster-kibana-%{+YYYY.MM.dd}"
      }
}
```

配置说明

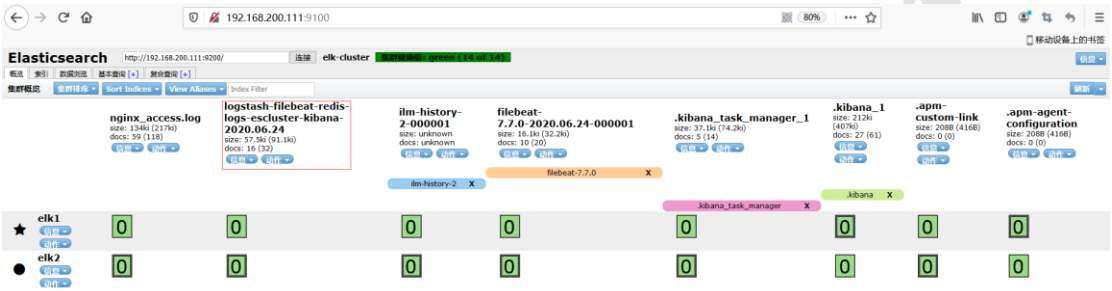
```
[root@elk1 ~]# cat /etc/logstash/conf.d/nginx_redis.conf
input {    #数据源信息，需与 redis 服务器的信息保持一致
      redis {
            host => "192.168.200.113"
            port => "6379"
            password => "123"
            key => "nginx_redis"
```

```
            data_type => "list"
            db => 0
        }
}

filter {
        grok {     #格式化结构的插件
            match => {
                "message" => "%{IPV4:remote_addr} - (%{USERNAME:remote_user}|-)
\[%{HTTPDATE:time_local}\] \"%{WORD:request_me
thod} %{URIPATHPARAM:request_uri}
HTTP/%{NUMBER:http_protocol}\" %{NUMBER:http_status} %{NUMBER:body_bytes_sent}
\"%{GREEDYDATA:http_referer}\" \"%{GREEDYDATA:http_user_agent}\"
\"(%{IPV4:http_x_forwarded_for}|-)\""            }     #前者代表被调用的插件函数，后者被
赋予为函数值对应键名
        }

        geoip {    #分析处理访客 IP 的插件
            source => "remote_addr"
            target => "geoip"
            database => "/opt/GeoLite2-City.mmdb"
            add_field => ["[geoip][coordinates]", "%{[geoip][longitude]}"]
            add_field => ["[geoip][coordinates]", "%{[geoip][latitude]}"]
        }

        date {
            locale => "en"
            match => ["time_local", "dd/MM/yyyy:HH:mm:ss Z"]       #定义时间格式
        }

        mutate {
            convert => ["[geoip][coordinates]", "float"]       #将经纬度转化为浮点数
        }
}

output {
        elasticsearch {
            hosts => ["http://192.168.200.111:9200","http://192.168.200.112:9200"] #输出对象
            index => "logstash-filebeat-redis-logs-escluster-kibana-%{+YYYY.MM.dd}"      #输出信
息的索引
        }     #调用 geoip 插件时为了能使后续地理图制作时系统识别 gro_point 数据类型，将
输出索引页的名称改为”logstash-*”开头就可以，此外添加 API 映射也能够解决。
}
```

10.6、杀死之前的 logstash 进程

```
[root@elk1 ~]# ps aux | grep logstash
[root@elk1 ~]# kill -9 65970
[root@elk1 ~]# ps aux | grep logstash
root       69915  0.0  0.0 112824   980 pts/0    S+   23:43   0:00 grep --color=auto logstash

[root@elk1 ~]# /usr/share/logstash/bin/logstash --path.settings /etc/logstash/ -f /etc/logstash/conf.d/nginx_redis.conf &
```

10.7、通过 es-head 查看验证



10.8、导入外网 IP 的 nginx 访问日志

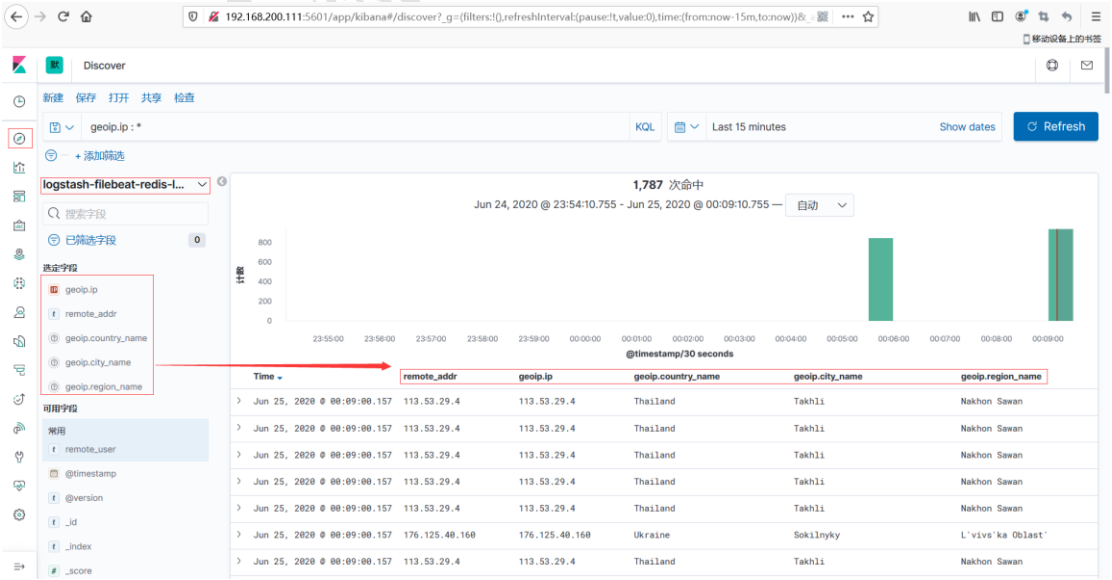
在内网环境下，无法获取外网真实 IP，所以为了模拟真实的生产环境，向 nginx 的访问日志中加入以下数条数据。

```
[root@web ~]# cat access.log-demo >> /var/log/nginx/access.log
```

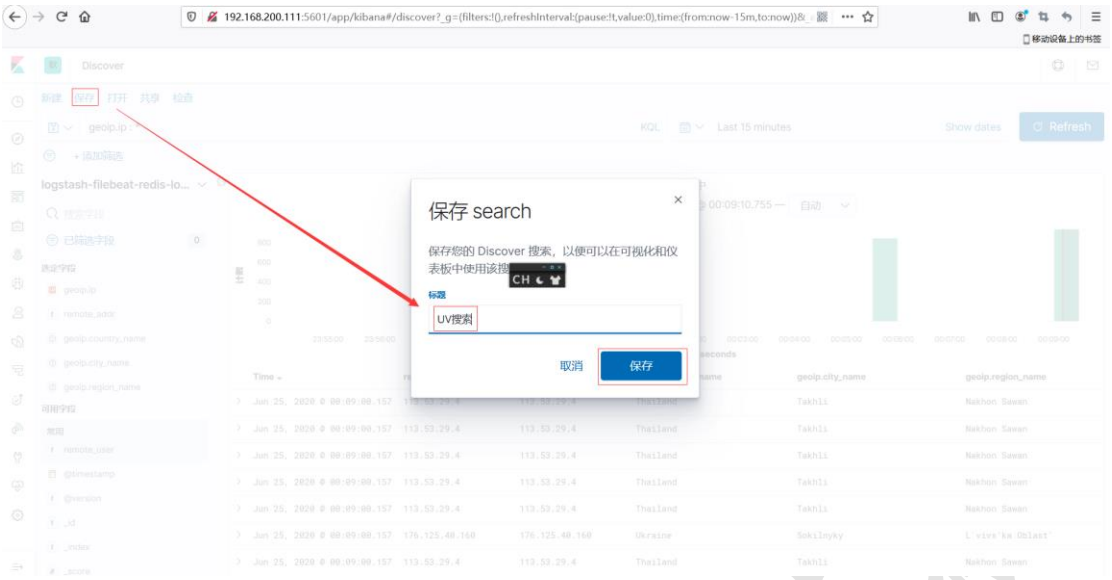
10.9、制作 UV 地理位置图

创建索引模式

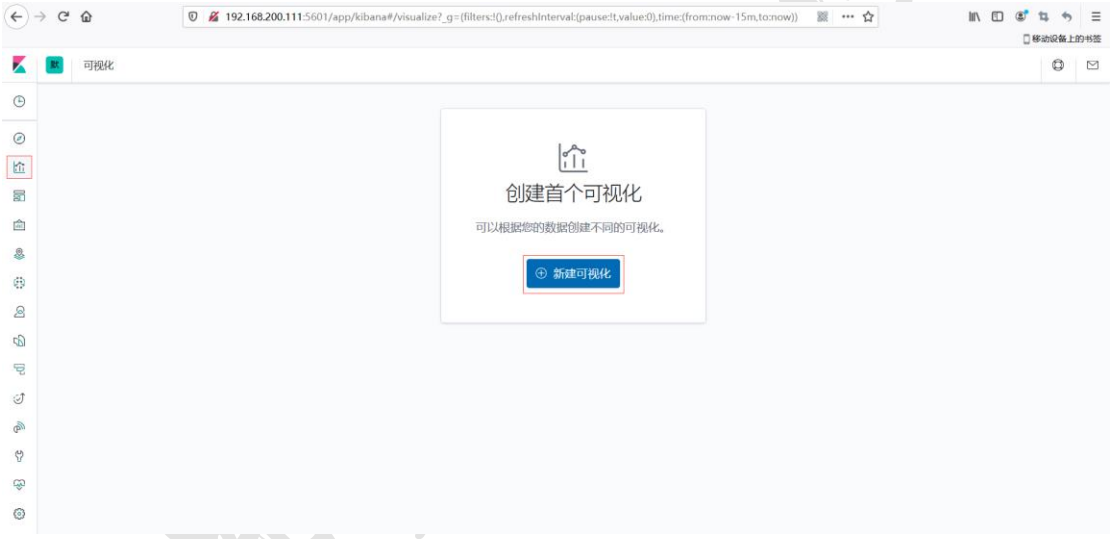
查看检索索引数据

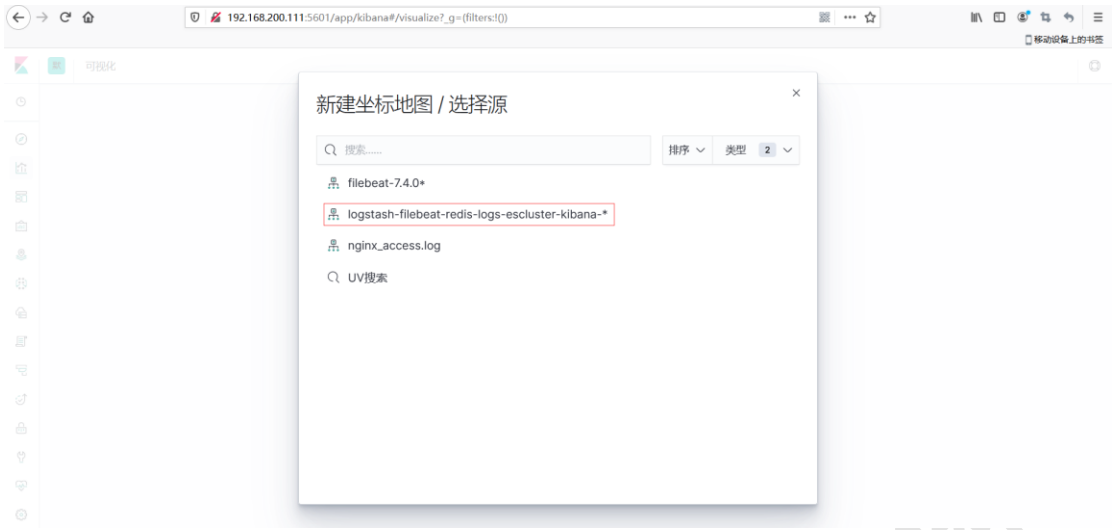


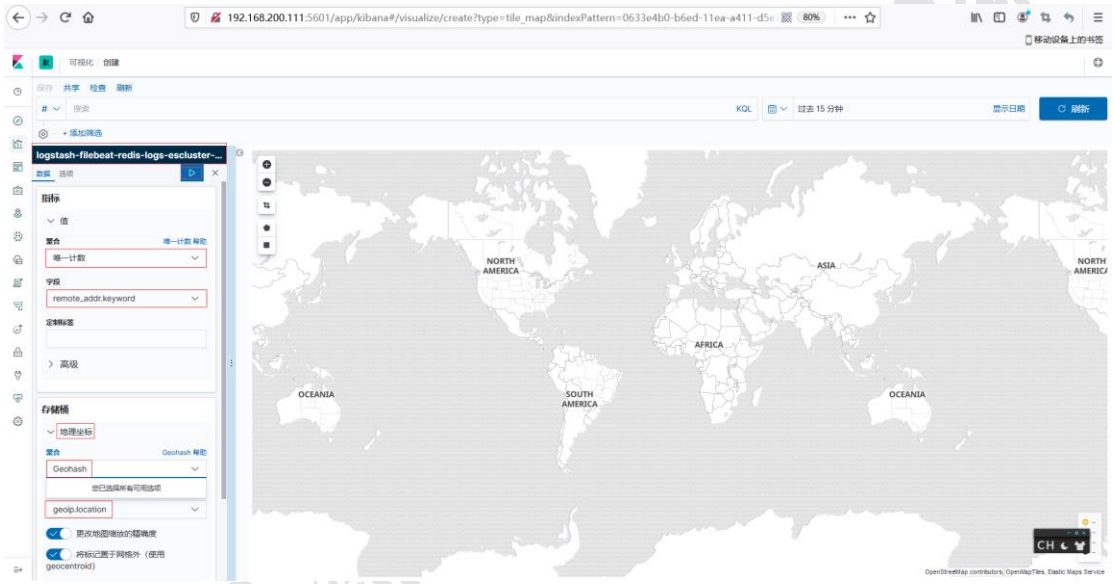
保存索引搜索

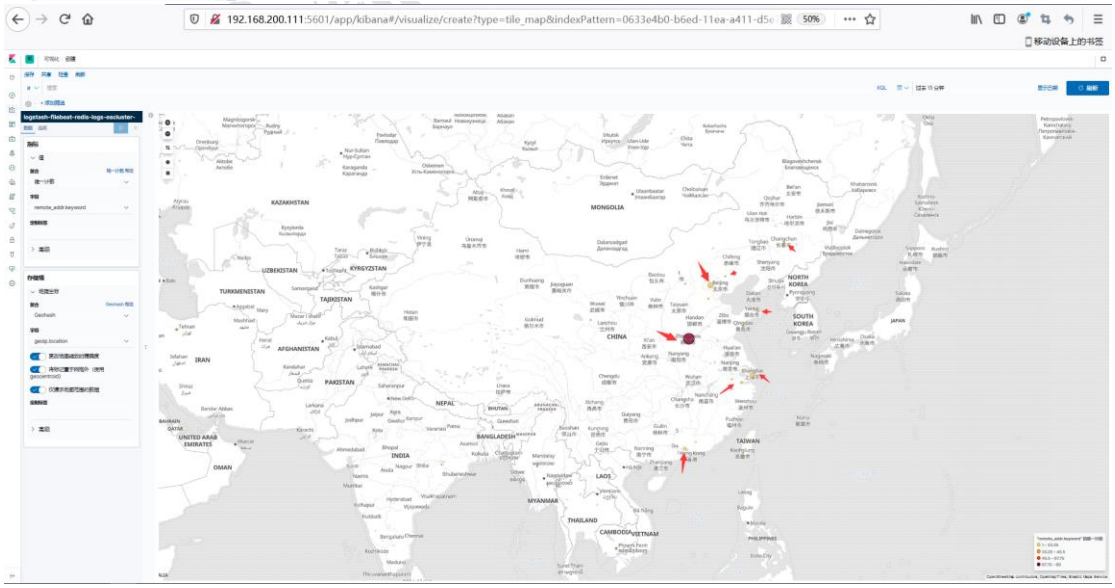


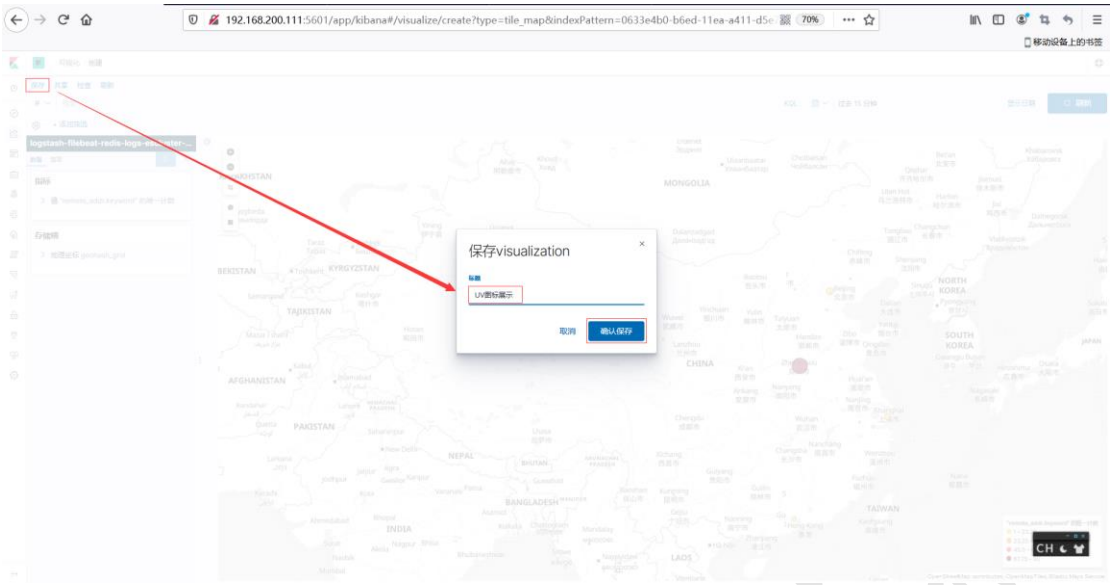
10.10、创建可视化图并保存




新建坐标地图 / 选择源
filebeat-7.4.0+
logstash-filebeat-redis-logs-escluster-kibana-*
nginx_access.log
UV搜索


NORTH AMERICA
ASIA
AFRICA
OCEANIA
SOUTH AMERICA
OCEANIA


KAZAKHSTAN
MONGOLIA
UZBEKISTAN
KYRGYZSTAN
TURKMENISTAN
TAJIKISTAN
AFGHANISTAN
IRAN
PAKISTAN
CHINA
NEPAL
INDIA
BANGLADESH
MYANMAR
THAILAND
LAOS
CAMBODIA
VIETNAM
TAIWAN
NORTH KOREA
SOUTH KOREA
JAPAN



十一、基于 Nginx 的用户控制

```
[root@elk1 ~]# rpm -ivh epel-release-latest-7.noarch.rpm
[root@elk1 ~]# yum -y install nginx
[root@elk1 ~]# htpasswd -c /etc/nginx/conf.d/user crushlinux
[root@elk1 ~]# vim /etc/nginx/nginx.conf
116         include /etc/nginx/conf.d/*.conf;
117 }
[root@elk1 ~]# vim /etc/nginx/conf.d/kibana.conf
upstream kibana_server {
            server    192.168.200.111:5601 weight=1 max_fails=3 fail_timeout=60;
}

server {
            listen 5600;
            server_name 192.168.200.111;
            auth_basic "Kibana Access";
            auth_basic_user_file /etc/nginx/conf.d/user;

            location / {
                        proxy_pass http://kibana_server;
                        proxy_http_version 1.1;
                        proxy_set_header Upgrade $http_upgrade;
                        proxy_set_header Connection 'upgrade';
                        proxy_set_header Host $host;
                        proxy_cache_bypass $http_upgrade;
            }
}
```

```
[root@elk1 ~]# netstat -lnpt | grep nginx
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
68992/nginx: master
tcp        0      0 0.0.0.0:5600            0.0.0.0:*               LISTEN
68992/nginx: master
```

浏览器访问测试: http://192.168.200.111:5600