// +----------------------------------------------------------------------------------------------------------------

// | ElasticSearch+Logstash+Kibana+Filebeat+Kafka+ZooKeeper海量日志分析

// +----------------------------------------------------------------------------------------------------------------

// | Version: 1.0

// +----------------------------------------------------------------------------------------------------------------

// | Author: JettJia <JettJia@qq.com>

// +----------------------------------------------------------------------------------------------------------------

// | Date: 2018.5

// +----------------------------------------------------------------------------------------------------------------

灰色底**黑色加粗**：输入的系统命令

灰色底： 输入的系统命令返回的信息或者配置文件文本信息

黄色底： 技巧或需要注意的注释信息

橙色底： 需要特别注意的地方

蓝色字体： 内容注释

# 1 介绍

## 1.1 解决的问题

1）开发人员不能登录线上服务器查看详细日志，节点多了错误日志查看麻烦

2）各个系统都有日志，日至数据分散难以查找

3）日志数据量大，查询速度慢，或者数据不够实时

4）日志收集后，可以用Hadoop、Storm等进行分析

## 1.2 组件说明

1）Filebeat

日志收集，也可以用logstash等

1. Logstash

日志解析，统一格式后输出给ES集群

1. Kafka

统一、高吞吐、低延迟,分布式事务日志架构的大规模发布/订阅消息队列

1. ZooKeeper

分布式配置服务、同步服务和命名注册，状态管理，监控进程等服务

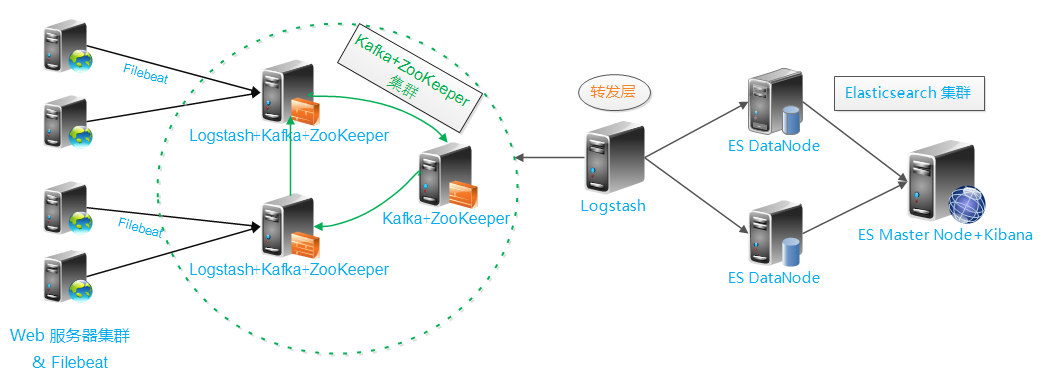
1. Elasticsearch

分布式多全文搜索引擎，基于RESTful web接口，实时日志分析服务的核心技术，强大的搜索功能

1. Kibana

基于Elasticsearch的数据可视化组件，超强的数据可视化能力

## 1.3 架构解读



**解读 : 整个架构从左到右，总共分为5层**

**第一层、数据采集层**

最左边的是业务服务器集群，上面安装了filebeat做日志采集，同时把采集的日志分别发送给两个logstash服务。

**第二层、数据处理层，数据缓存层**

logstash服务把接受到的日志经过格式处理，转存到本地的kafka broker+zookeeper 集群中。

**第三层、数据转发层**

这个单独的Logstash节点会实时去kafka broker集群拉数据，转发至ES DataNode。

**第四层、数据持久化存储**

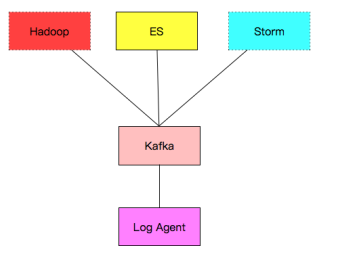
ES DataNode 会把收到的数据，写磁盘，建索引库。

**第五层、数据检索，数据展示**

ES Master + Kibana 主要协调ES集群，处理数据检索请求，数据展示。

**扩展说明**

kafka中的数据是持久化存储的，不像在ELK+redis架构中，redis中收集到的数据会被真正的消耗掉。所以kafka中的数据也可以被Hadoop、Storm等去分析



## 1.3 版本说明

openjdk\_1.8.0\_171

filebeat-1.2.3-1.x86\_64

logstash-2.1.3-1.noarch

kafka\_2.11-0.10.0.1.tgz

zookeeper-3.4.10.tar.gz

elasticsearch-2.4.6-1.noarch

kibana-4.3.1-linux-x64.tar.gz

## 1.4 规范化

需求分析：

访问日志：apache访问日志、nginx访问日志、tomcat; 处理方式file -filter

错误日志：error log、java日志；处理方式直接获取，Java异常需要用multiline

系统日志：/var/logs/\*、syslog; 处理方式rsyslog

运行日志：程序写的（要求json）；处理方式file

网络日志：防火墙、交换机、路由器的日志 处理方式syslog

标准化：

日志路径：/data/logs/

命名规则：/data/logs/access\_log, /data/logs/error\_log, /data/logs/runtime\_log

日志切割：access、 error用脚本crontab定时进行切分；runtime\_log开发来指定;

access按小时切分，error按天切分（根据业务具体而定）

日志备份：所有的原始文本，rsync到NAS，然后删除最近三天前的日志。

# 2 环境准备

## 2.1 服务器介绍

|  |  |  |  |
| --- | --- | --- | --- |
| IP | 名称 | 角色 | 所属集群 |
| 10.0.0.20 | nginx1 | 业务服务器+filebeat | 业务服务器集群 [比如负载均衡的nginx] |
| 10.0.0.30 | kafka1 | Logstash+Kafka+ZooKeeper | Kafka Broker 集群 |
| 10.0.0.31 | kafka2 | Logstash+Kafka+ZooKeeper |
| 10.0.0.32 | kafka3 | Kafka+ZooKeeper |
| 10.0.0.50 | logstash | logstash | 数据转发 |
| 10.0.0.60 | es1 | ES DataNode | es集群，每个node上都安装了kibana（补充可以用nginx实现负载均衡和访问权限） |
| 10.0.0.61 | es2 | ES DataNode |
| 10.0.0.62 | es3 | ES Master+Kibana |

## 2.2 系统环境

cat /etc/redhat-release

CentOS release 6.7 (Final)

uname -a

Linux linux-node1 2.6.32-573.el6.x86\_64 #1 SMP Thu Jul 23 15:44:03 UTC 2015 x86\_64 x86\_64 x86\_64 GNU/Linux

cat /etc/hosts

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

10.0.0.20 nginx1

10.0.0.30 kafka1

10.0.0.31 kafka2

10.0.0.32 kafka3

10.0.0.50 logstash

10.0.0.60 es1

10.0.0.61 es2

10.0.0.62 es3

# 3 部署Elasticsearch集群

## 3.1 部署ES Master节点

### 1 安装jdk1.8 elasticsearch2.4.6

安装JDK

[root@es3 ~]# yum install -y java

安装es

[root@es3 ~]# rpm --import <https://packages.elastic.co/GPG-KEY-elasticsearch>

[root@es3 ~]# vim /etc/yum.repos.d/elasticsearch.repo

[elasticsearch-2.x]

name=Elasticsearch repository for 2.x packages

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

gpgcheck=1

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

enabled=1

[root@es3 ~]# yum install -y elasticsearch

# es 默认安装在/usr/share/elasticsearch/

### 2 系统调优，JVM调优

1. 配置系统最大打开文件描述符数

[root@es3 ~]# vim /etc/sysctl.conf

# file\_max open from jet

fs.file-max=65535

1. 配置进程最大打开文件描述符

[root@es3 ~]# vim /etc/security/limits.conf

# end of file from jet

\* soft nofile 65535

\* hard nofile 65535

1. 配置JVM内存

[root@es3 ~]# vim /etc/sysconfig/elasticsearch

ES\_HEAP\_SIZE=4g #机器内存的一半，这台机器的内存是8g

### 3 配置启动ES Master节点

[root@es3 ~]# vim /etc/elasticsearch/elasticsearch.yml

[root@es3 ~]# grep '^[a-Z]' /etc/elasticsearch/elasticsearch.yml

cluster.name: bigdata

node.name: es3

path.data: /data/es-data

path.logs: /var/log/elasticsearch/

bootstrap.memory\_lock: true

network.host: 0.0.0.0

http.port: 9200

discovery.zen.ping.unicast.hosts: ["10.0.0.62", "10.0.0.61","10.0.0.60"]

创建数据存放目录，并授权

[root@es3 ~]# mkdir -p /data/es-data

[root@es3 ~]# chown elasticsearch.elasticsearch /data/es-data/

启动和检查

[root@es3 ~]# /etc/init.d/elasticsearch start

正在启动 elasticsearch： [确定]

[root@es3 ~]# netstat -lntup|grep 9200

tcp 0 0 :::9200 :::\* LISTEN 26805/java

### 4 安装head、kopf插件

head插件显示索引和分片情况

[root@es3 ~]# /usr/share/elasticsearch/bin/plugin install mobz/elasticsearch-head

kopf插件监控elasticsearch

[root@es3 ~]# /usr/share/elasticsearch/bin/plugin install lmenezes/elasticsearch-kopf

说明：

1）插件安装目录 /usr/share/elasticsearch/plugins/

2）插件访问

http://10.0.0.62:9200/\_plugin/head/

http://10.0.0.62:9200/\_plugin/kopf/

## 3.2 部署ES DataNode节点

安装和系统调优同master节点，不用安装插件，配置文件不同，配置文件如下

### 3.2.1 部署ES DataNode 10.0.0.60

1）vim /etc/elasticsearch/elasticsearch.yml

cluster.name: bigdata

node.name: es1

path.data: /data/es-data

path.logs: /var/log/elasticsearch/

bootstrap.memory\_lock: true

network.host: 0.0.0.0

http.port: 9200

discovery.zen.ping.unicast.hosts: ["10.0.0.62", "10.0.0.61","10.0.0.60"]

2）

mkdir -p /data/es-data

chown elasticsearch.elasticsearch /data/es-data/

3）

/etc/init.d/elasticsearch start

### 3.2.2 部署ES DataNode 10.0.0.61

1）vim /etc/elasticsearch/elasticsearch.yml

cluster.name: bigdata

node.name: es2

path.data: /data/es-data

path.logs: /var/log/elasticsearch/

bootstrap.memory\_lock: true

network.host: 0.0.0.0

http.port: 9200

discovery.zen.ping.unicast.hosts: ["10.0.0.62", "10.0.0.61","10.0.0.60"]

2）

mkdir -p /data/es-data

chown elasticsearch.elasticsearch /data/es-data/

3）

/etc/init.d/elasticsearch start

## 3.3 查看集群状态

<http://10.0.0.62:9200/_plugin/head/>



# 4 部署kafka集群

## 4.1 zookeeper集群

安装JDK（3个节点都要安装）

yum install -y java

zk下载地址https://www.apache.org/dyn/closer.cgi/zookeeper/

### 1 安装配置

# 10.0.0.30

mkdir -p /home/tools

cd /home/tools/

tar zxf zookeeper-3.4.10.tar.gz

mv zookeeper-3.4.10 /usr/local/zookeeper

cd /usr/local/zookeeper

mkdir data

cp conf/zoo\_sample.cfg conf/zoo.cfg

vim conf/zoo.cfg

# The number of milliseconds of each tick

tickTime=2000

# The number of ticks that the initial

# synchronization phase can take

initLimit=10

# The number of ticks that can pass between

# sending a request and getting an acknowledgement

syncLimit=5

# the directory where the snapshot is stored.

# do not use /tmp for storage, /tmp here is just

# example sakes.

dataDir=/usr/local/zookeeper/data

# the port at which the clients will connect

clientPort=2181

# the maximum number of client connections.

# increase this if you need to handle more clients

#maxClientCnxns=60

server.0=10.0.0.30:2888:3888

server.1=10.0.0.31:2888:3888

server.2=10.0.0.32:2888:3888

#

# Be sure to read the maintenance section of the

# administrator guide before turning on autopurge.

#

# http://zookeeper.apache.org/doc/current/zookeeperAdmin.html#sc\_maintenance

#

# The number of snapshots to retain in dataDir

#autopurge.snapRetainCount=3

# Purge task interval in hours

# Set to "0" to disable auto purge feature

#autopurge.purgeInterval=1

### 2 同步配置到另外两节点

# 10.0.0.30

rsync -avzP /usr/local/zookeeper/ root@10.0.0.31:/usr/local/zookeeper/

rsync -avzP /usr/local/zookeeper/ root@10.0.0.32:/usr/local/zookeeper/

### 3 创建myid文件

# 10.0.0.30

echo 0 >/usr/local/zookeeper/data/myid

# 10.0.0.31

echo 1 >/usr/local/zookeeper/data/myid

# 10.0.0.32

echo 2 >/usr/local/zookeeper/data/myid

### 4 启动服务 & 查看节点状态

# 10.0.0.30

/usr/local/zookeeper/bin/zkServer.sh start

netstat -lntup|grep 2181

# 10.0.0.31

/usr/local/zookeeper/bin/zkServer.sh start

netstat -lntup|grep 2181

# 10.0.0.32

/usr/local/zookeeper/bin/zkServer.sh start

netstat -lntup|grep 2181

查看状态

/usr/local/zookeeper/bin/zkServer.sh status

## 4.2 部署kafka集群

### 1 安装配置

# 10.0.0.30

cd /home/tools/

tar zxf kafka\_2.11-0.10.0.1.tgz

mv kafka\_2.11-0.10.0.1 /usr/local/

vim /usr/local/kafka\_2.11-0.10.0.1/config/server.properties

# Licensed to the Apache Software Foundation (ASF) under one or more

# contributor license agreements. See the NOTICE file distributed with

# this work for additional information regarding copyright ownership.

# The ASF licenses this file to You under the Apache License, Version 2.0

# (the "License"); you may not use this file except in compliance with

# the License. You may obtain a copy of the License at

#

# http://www.apache.org/licenses/LICENSE-2.0

#

# Unless required by applicable law or agreed to in writing, software

# distributed under the License is distributed on an "AS IS" BASIS,

# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

# See the License for the specific language governing permissions and

# limitations under the License.

# see kafka.server.KafkaConfig for additional details and defaults

############################# Server Basics #############################

# The id of the broker. This must be set to a unique integer for each broker.

broker.id=0 #唯一，3个node可以按0,1,2来指定

############################# Socket Server Settings #############################

# The address the socket server listens on. It will get the value returned from

# java.net.InetAddress.getCanonicalHostName() if not configured.

# FORMAT:

# listeners = security\_protocol://host\_name:port

# EXAMPLE:

# listeners = PLAINTEXT://your.host.name:9092

#listeners=PLAINTEXT://:9092

# Hostname and port the broker will advertise to producers and consumers. If not set,

# it uses the value for "listeners" if configured. Otherwise, it will use the value

# returned from java.net.InetAddress.getCanonicalHostName().

#advertised.listeners=PLAINTEXT://your.host.name:9092

# The number of threads handling network requests

num.network.threads=3

# The number of threads doing disk I/O

num.io.threads=8

# The send buffer (SO\_SNDBUF) used by the socket server

socket.send.buffer.bytes=102400

# The receive buffer (SO\_RCVBUF) used by the socket server

socket.receive.buffer.bytes=102400

# The maximum size of a request that the socket server will accept (protection against OOM)

socket.request.max.bytes=104857600

############################# Log Basics #############################

# A comma seperated list of directories under which to store log files

log.dirs=/usr/local/kafka/kafka\_2.11-0.10.0.1/data

# The default number of log partitions per topic. More partitions allow greater

# parallelism for consumption, but this will also result in more files across

# the brokers.

num.partitions=1

# The number of threads per data directory to be used for log recovery at startup and flushing at shutdown.

# This value is recommended to be increased for installations with data dirs located in RAID array.

num.recovery.threads.per.data.dir=1

############################# Log Flush Policy #############################

# Messages are immediately written to the filesystem but by default we only fsync() to sync

# the OS cache lazily. The following configurations control the flush of data to disk.

# There are a few important trade-offs here:

# 1. Durability: Unflushed data may be lost if you are not using replication.

# 2. Latency: Very large flush intervals may lead to latency spikes when the flush does occur as there will be a lot of data to flush.

# 3. Throughput: The flush is generally the most expensive operation, and a small flush interval may lead to exceessive seeks.

# The settings below allow one to configure the flush policy to flush data after a period of time or

# every N messages (or both). This can be done globally and overridden on a per-topic basis.

# The number of messages to accept before forcing a flush of data to disk

#log.flush.interval.messages=10000

# The maximum amount of time a message can sit in a log before we force a flush

#log.flush.interval.ms=1000

############################# Log Retention Policy #############################

# The following configurations control the disposal of log segments. The policy can

# be set to delete segments after a period of time, or after a given size has accumulated.

# A segment will be deleted whenever \*either\* of these criteria are met. Deletion always happens

# from the end of the log.

# The minimum age of a log file to be eligible for deletion

log.retention.hours=168

# A size-based retention policy for logs. Segments are pruned from the log as long as the remaining

# segments don't drop below log.retention.bytes.

#log.retention.bytes=1073741824

# The maximum size of a log segment file. When this size is reached a new log segment will be created.

log.segment.bytes=1073741824

# The interval at which log segments are checked to see if they can be deleted according

# to the retention policies

log.retention.check.interval.ms=300000

############################# Zookeeper #############################

# Zookeeper connection string (see zookeeper docs for details).

# This is a comma separated host:port pairs, each corresponding to a zk

# server. e.g. "127.0.0.1:3000,127.0.0.1:3001,127.0.0.1:3002".

# You can also append an optional chroot string to the urls to specify the

# root directory for all kafka znodes.

zookeeper.connect=10.0.0.30:2181,10.0.0.31:2181,10.0.0.32:2181 #zk集群

# Timeout in ms for connecting to zookeeper

zookeeper.connection.timeout.ms=6000

### 2 同步到另外两节点

# 10.0.0.30

scp -rp /usr/local/kafka\_2.11-0.10.0.1 root@10.0.0.31:/usr/local/

scp -rp /usr/local/kafka\_2.11-0.10.0.1 root@10.0.0.32:/usr/local/

### 3 修改节点的borker.id

# 10.0.0.30

borker.id=0

# 10.0.0.31

borker.id=1

# 10.0.0.32

borker.id=2

### 4 启动服务 & 查看节点状态

# 10.0.0.{30,31,32}

/usr/local/kafka\_2.11-0.10.0.1/bin/kafka-server-start.sh /usr/local/kafka\_2.11-0.10.0.1/config/server.properties

后台运行

nohup /usr/local/kafka\_2.11-0.10.0.1/bin/kafka-server-start.sh /usr/local/kafka\_2.11-0.10.0.1/config/server.properties > /dev/null 2>&1 &

查看状态

ps -ef|grep kafka

## 4.3 部署kafka集群--logstash服务

配置 GeoLiteCity ， 用于地图显示IP访问的城市

官网地址: http://dev.maxmind.com/geoip/legacy/geolite/

下载地址: http://geolite.maxmind.com/download/geoip/database/GeoLiteCity.dat.gz

### 1 安装配置logstash

# 10.0.0.30

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

vim /etc/yum.repos.d/logstash.repo

[logstash-2.1]

name=Logstash repository for 2.1.x packages

baseurl=http://packages.elastic.co/logstash/2.1/centos

gpgcheck=1

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

enabled=1

yum install -y logstash

解压gunzip GeoLiteCity.dat.gz

gunzip GeoLiteCity.dat.gz

mv GeoLiteCity.dat /etc/logstash/

配置logstash\_in\_kafka.conf文件

vim /etc/logstash/conf.d/logstash\_in\_kafka.conf

input {

beats {

port => 5044

codec => "json"

}

}

filter {

if [type] == "nginxacclog" {

geoip {

source => "clientip" # 与日志中访问地址的key要对应

target => "geoip"

database => "/etc/logstash/GeoLiteCity.dat"

add\_field => [ "[geoip][coordinates]","%{[geoip][longitude]}" ]

add\_field => [ "[geoip][coordinates]","%{[geoip][latitude]}" ]

}

mutate {

convert => [ "[geoip][coordinates]","float" ]

}

}

}

output {

kafka {

workers => 2

bootstrap\_servers => "10.0.0.30:9092,10.0.0.31:9092,10.0.0.32:9092"

topic\_id => "peiyinlog"

}

}

启动logstash

/opt/logstash/bin/logstash -f /etc/logstash/conf.d/logstash\_in\_kafka.conf &

# 10.0.0.31 同上

# 5 部署业务层，数据采集+filebeat

## 5.1安装配置nginx

这里直接yum安装，用于采集数据测试

yum install nginx -y

定制Nginx日志格式为json

log\_format json '{"@timestamp":"$time\_iso8601",'

'"slbip":"$remote\_addr",'

'"clientip":"$http\_x\_forwarded\_for",'

'"serverip":"$server\_addr",'

'"size":$body\_bytes\_sent,'

'"responsetime":$request\_time,'

'"domain":"$host",'

'"method":"$request\_method",'

'"requesturi":"$request\_uri",'

'"url":"$uri",'

'"appversion":"$HTTP\_APP\_VERSION",'

'"referer":"$http\_referer",'

'"agent":"$http\_user\_agent",'

'"status":"$status",'

'"devicecode":"$HTTP\_HA"}';

access\_log /data/logs/access\_log/access.log json;

error\_log /data/logs/error\_log/error.log error;

mkdir -p /data/logs/access\_log

mkdir -p /data/logs/error\_log

mkdir -p /data/logs/runtime\_log

重启

nginx -t

/etc/init.d/nginx restart

## 5.2 安装配置filebeat

官方文档

https://www.elastic.co/guide/en/beats/filebeat/1.2/filebeat-installation.html

# 10.0.0.20

安装

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

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

filebeat配置文件

vim /etc/filebeat/filebeat.yml

filebeat:

prospectors:

-

paths:

- /var/log/messages

input\_type: log

document\_type: messages

-

paths:

- /data/logs/access\_log/access.log

input\_type: log

document\_type: nginxacclog

-

paths:

- /data/logs/error\_log/error.log

input\_type: log

document\_type: peiyinlar\_error

-

paths:

- /data/logs/runtime\_log/\*.log

input\_type: log

document\_type: runtime\_log

output:

logstash:

hosts: ["10.0.0.30:5044","10.0.0.31:5044"]

启动filebeat

/etc/init.d/filebeat start

# 6 部署logstash数据转发

安装Java

yum install java -y

## 1 安装logstash

10.0.0.50

安装logstash同上

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

vim /etc/yum.repos.d/logstash.repo

[logstash-2.1]

name=Logstash repository for 2.1.x packages

baseurl=http://packages.elastic.co/logstash/2.1/centos

gpgcheck=1

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

enabled=1

yum install -y logstash

## 2 配置logstash 写入es

vim /etc/logstash/conf.d/kafka\_to\_es.conf

input{

kafka {

zk\_connect => "10.0.0.30:2181,10.0.0.31:2181,10.0.0.32:2181"

group\_id => "logstash"

topic\_id => "peiyinlog"

reset\_beginning => false

consumer\_threads => 50

decorate\_events => true

}

}

# 删除一些不需要的字段

filter {

if [type] == "nginxacclog" {

mutate {

remove\_field => ["slbip","kafka","domain","serverip","url","@version","offset","input\_type","count","source","fields","beat.hostname","host","tags"]

}

}

}

output {

if [type] == "nginxacclog" {

# stdout {codec => rubydebug }

elasticsearch {

hosts => ["10.0.0.60:9200","10.0.0.61:9200","10.0.0.62:9200"]

index => "logstash-nginxacclog-%{+YYYY.MM.dd}"

manage\_template => true

flush\_size => 50000

idle\_flush\_time => 10

workers => 2

}

}

if [type] == "messages" {

elasticsearch {

hosts => ["10.0.0.60:9200","10.0.0.61:9200","10.0.0.62:9200"]

index => "logstash-messages-%{+YYYY.MM.dd}"

manage\_template => true

flush\_size => 50000

idle\_flush\_time => 30

workers => 1

}

}

if [type] == "peiyinlar\_error" {

elasticsearch {

hosts => ["10.0.0.60:9200","10.0.0.61:9200","10.0.0.62:9200"]

index => "logstash-nginxerror-%{+YYYY.MM.dd}"

manage\_template => true

flush\_size => 2000

idle\_flush\_time => 10

}

}

if [type] == "runtime\_log" {

elasticsearch {

hosts => ["10.0.0.60:9200","10.0.0.61:9200","10.0.0.62:9200"]

index => "logstash-runtime\_log-%{+YYYY.MM.dd}"

manage\_template => true

flush\_size => 2000

idle\_flush\_time => 10

}

}

}

启动logstash

/opt/logstash/bin/logstash -f /etc/logstash/conf.d/kafka\_to\_es.conf &

登陆到任意一台kafka主机，查看数据的缓存和消费情况

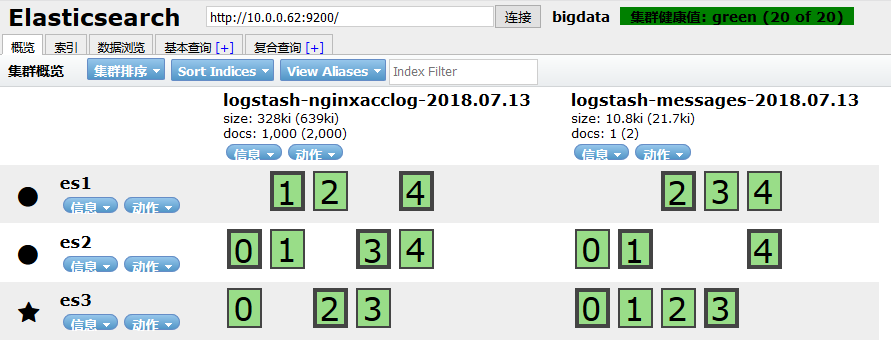
bin/kafka-consumer-groups.sh --group logstash -describe --zookeeper 127.0.0.1:2181

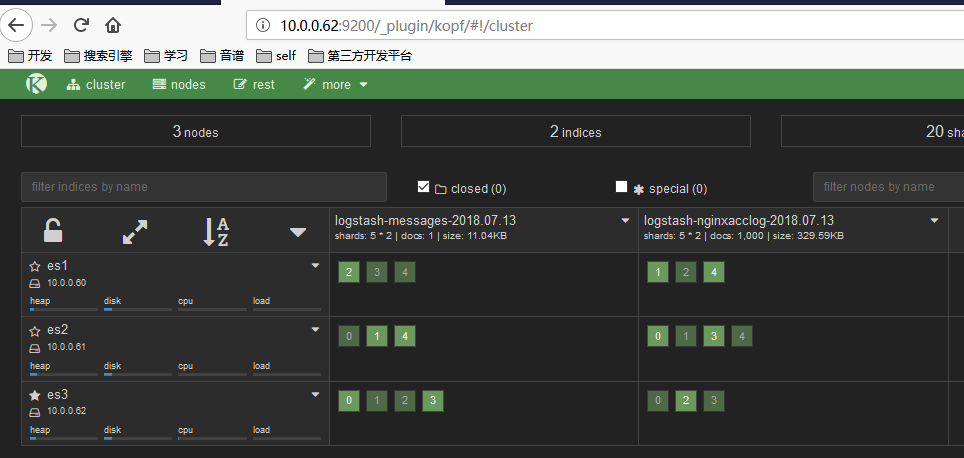
## 3 登录head检查

增加Nginx访问日志

yum install httpd-tools

ab -n 1000 -c 1 http://10.0.0.20/





# 7 修改ES索引模板配置

为什么要做这一步呢？ 因为logstash写入数据到ES时，会自动选用一个索引模版。 我们可以看一下



这个模版其实也挺好，不过有一个参数，我标记出来了。 "refresh\_interval":"5s" 这个参数用于控制，索引的刷新频率。 索引的刷新频率越快，你搜索到的数据就实时。 这里是5秒。 一般我们日志场景不需要这么高的实时性。 可以适当降低该参数，提高ES 索引库的写入速度。

**上传自定义模版**

curl -XPUT http://10.0.0.62:9200/\_template/logstash2 -d '

{

"order":1,

"template":"logstash-\*",

"settings":{

"index":{

"refresh\_interval":"120s"

}

},

"mappings":{

"\_default\_":{

"\_all":{

"enabled":false

}

}

}

}'

由于这个自定义模版，我把优先级 order 定义的比logstash模版高，而模版的匹配规则又一样，所以这个自定义模版的配置会覆盖原logstash模版。

# 8 配置kibana数据展示层

# 10.0.0.62

cd /usr/local/src

wget https://download.elastic.co/kibana/kibana/kibana-4.3.1-linux-x64.tar.gz

tar zxf kibana-4.3.1-linux-x64.tar.gz

mv kibana-4.3.1-linux-x64 /usr/local/

ln -s /usr/local/kibana-4.3.1-linux-x64/ /usr/local/kibana

编辑kibana配置文件

grep '^[a-Z]' /usr/local/kibana/config/kibana.yml

server.port: 5601

server.host: "0.0.0.0"

elasticsearch.url: "http://10.0.0.62:9200"

kibana.index: ".kibana"

启动kibana

/usr/local/kibana/bin/kibana

浏览器访问

http://10.0.0.62:5601/

**定制 Elasticsearch 索引的 Index pattern**

默认情况下，Kibana认为你要访问的是通过Logstash导入Elasticsearch的数据，这时候你可以用默认的 logstash-\* 作为你的 index pattern。 通配符（\*）匹配索引名中任意字符任意个数。

选择一个包含了时间戳的索引字段（字段类型为 date 的字段），可以用来做基于时间的处理。Kibana 会读取索引的

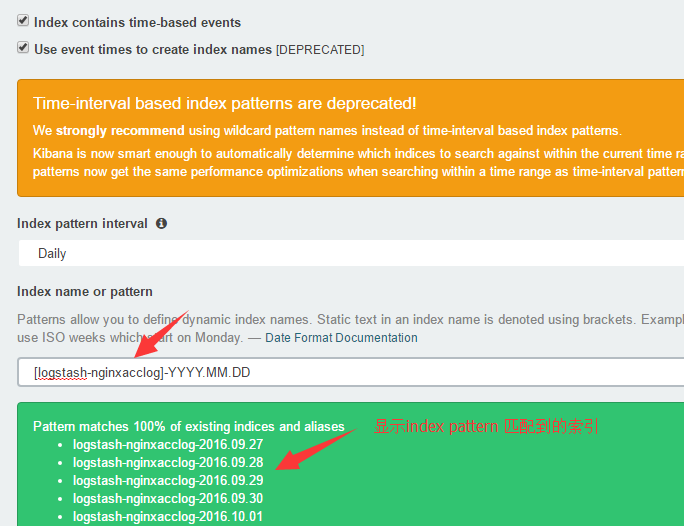
映射，然后列出所有包含了时间戳的字段。如果你的索引没有基于时间的数据.

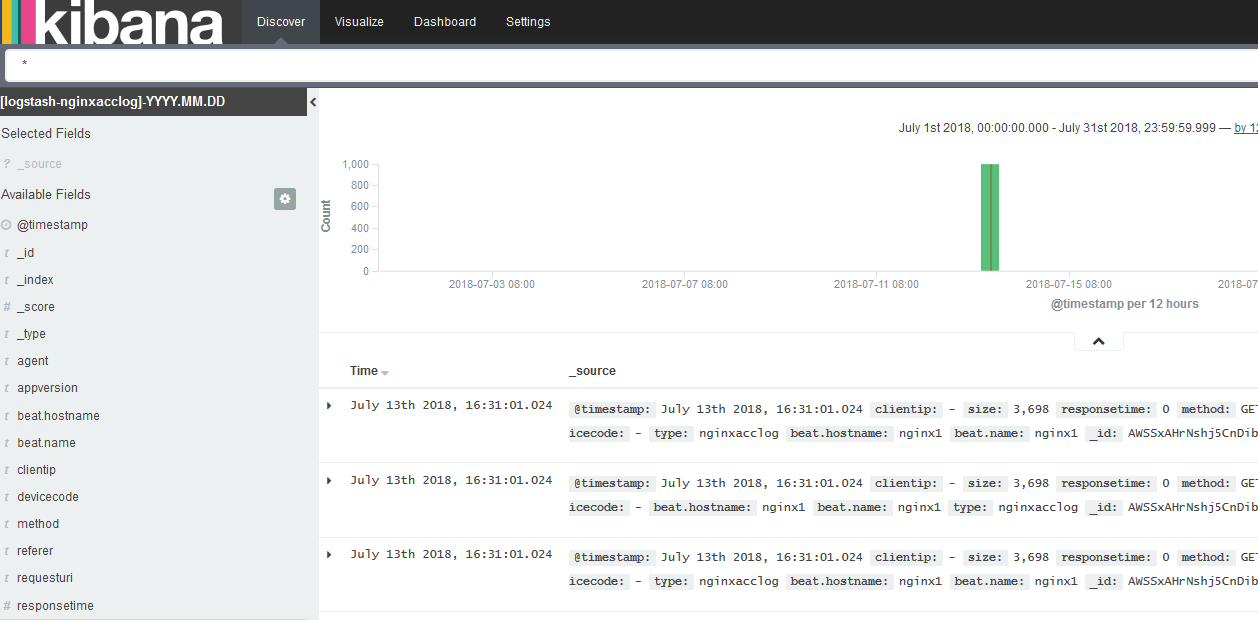
关闭 Index contains time-based events 参数。

如果一个新索引是定期生成，而且索引名中带有时间戳，选择 Use event times to create index names 选项，

然后再选择 Index pattern interval 。这可以提高搜索性能，Kibana 会至搜索你指定的时间范围内的索引。在你用 Logstash 输出数据给Elasticsearch 的情况下尤其有效。

由于我们的索引是用日期命名，按照每天分割的。 index pattern 如下

[](http://s2.51cto.com/wyfs02/M00/88/AC/wKiom1f-_yTCvOe2AADRDyL4MEg193.png)



参考

<http://blog.51cto.com/tchuairen/1861167>

<http://www.voidcn.com/article/p-nciizxus-bpm.html>

<https://www.unixhot.com/article/61>

<https://www.elastic.co>