

# Linux搭建ELK日志收集系统：

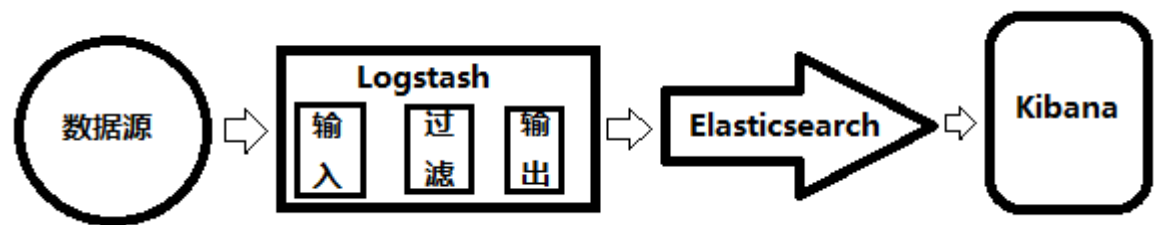
## Centos7部署ELK日志收集系统

### 一、ELK概述：

ELK是一组开源软件的简称，其包括Elasticsearch、Logstash 和 Kibana。ELK最近几年发展迅速，已经成为目前最流行的集中式日志解决方案。

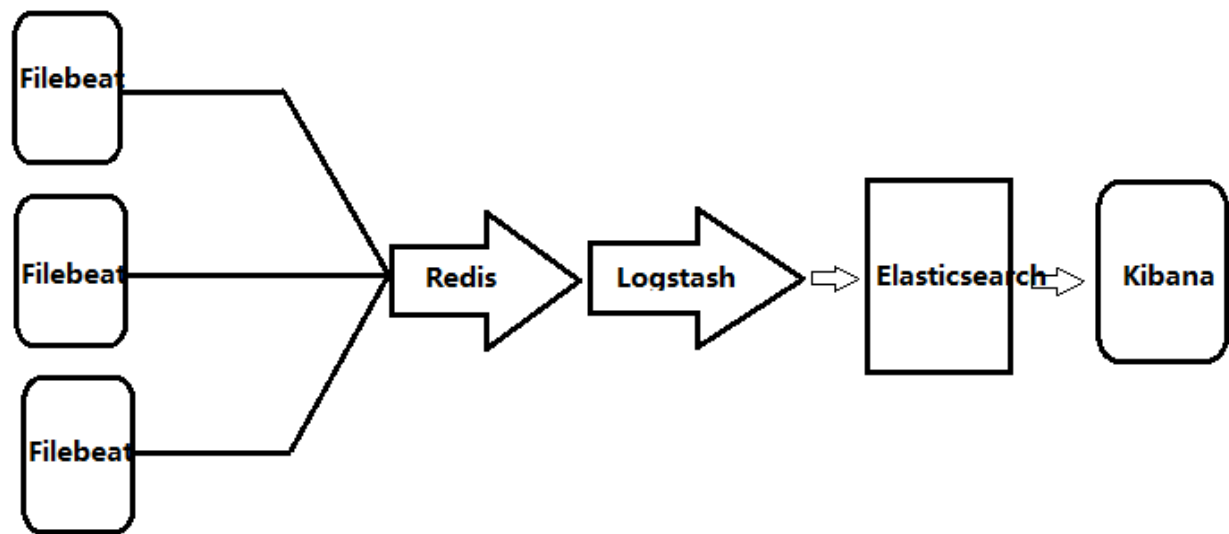
- Elasticsearch: 能对大容量的数据进行接近实时的存储，搜索和分析操作。本项目中主要通过Elasticsearch存储所有获取的日志。
- Logstash: 数据收集引擎，它支持动态的从各种数据源获取数据，并对数据进行过滤，分析，丰富，统一格式等操作，然后存储到用户指定的位置。
- Kibana: 数据分析与可视化平台，对Elasticsearch存储的数据进行可视化分析，通过表格的形式展现出来。
- Filebeat: 轻量级的开源日志文件数据搜集器。通常在需要采集数据的客户端安装Filebeat,并指定目录与日志格式,Filebeat就能快速收集数据，并发送给logstash进行解析，或是直接发给Elasticsearch存储。
- Redis: NoSQL数据库(key-value)，也数据轻型消息队列，不仅可以对高并发日志进行削峰还可以对整个架构进行解耦

### 传统ELK的经典框架



单一的架构，logstash作为日志搜集器，从数据源采集数据，并对数据进行过滤，格式化处理，然后交由Elasticsearch存储，kibana对日志进行可视化处理。

### 新型ELK框架



Filebeats是一种轻量级的日志搜集器，其不占用系统资源，自出现之后，迅速更新了原有的elk架构。Filebeats将收集到的数据发送给Logstash解析过滤，在Filebeats与Logstash传输数据的过程中，为了安全性，可以通过ssl认证来加强安全性。之后将其发送到Elasticsearch存储，并由kibana可视化分析。

## 二、新型ELK搭建详细过程

主机	IP	部署程序
1	192.169.203.143	Filebeat
2	192.169.203.144	Redis
3	192.169.203.135	Log
4	192.169.203.142	Elasticsearch
5	192.169.203.134	Kibana

下面是搭建过程中所需程序安装包

链接: <https://pan.baidu.com/s/1xQP-MW2XTywLgIhOI7kLA> 提取码: 8p3z 复制这段内容后打开百度网盘手机App，操作更方便哦

所有机器关闭 firewalld 和 selinux

### 1、客户端部署filebeat:

#### (1) 下载包

```
[root@filebeat ~]# wget https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-6.5.4-linux-x86_64.tar.gz
```

#### (2) 解压

```
[root@filebeat ~]# tar xzvf filebeat-6.5.4-linux-x86_64.tar.gz -C /usr/local/
[root@filebeat ~]# cd /usr/local/
[root@filebeat ~]# mv filebeat-6.5.4-linux-x86_64 filebeat
[root@elasticsearch local]# cd filebeat/
```

#### (3) 修改配置

修改 Filebeat 配置，支持收集本地目录日志 传到reids中

```
[root@filebeat ~]# cat /usr/local/filebeat
filebeat.inputs:
- type: log    #指定输入的类型

enabled: true
```

```

paths:
  - /var/log/nginx/*.log    #日志的路径
filebeat.config.modules:
  path: ${path.config}/modules.d/*.yaml
  reload.enabled: false
setup.template.settings:
  index.number_of_shards: 3
setup.kibana:
output.redis:
  hosts: ["192.168.203.144:6379"] #输出到redis的ip + 端口
  key: "eureka-log"    #指定输出的key

```

[root@filebeat ~]#

#### (4) 启动

```

[root@filebeat filebeat]# nohup ./filebeat -e -c filebeat.yml &
[root@filebeat filebeat]# [root@filebeat filebeat]# tail -f nohup.out
2019-12-11T18:17:00.000+0800    INFO    [monitoring]    log/log.go:144    Non-zero metrics in the
last 30s    {"monitoring": {"metrics": {"beat":{"cpu":{"system":{"ticks":1450,"time":
{"ms":3}}, "total":{"ticks":4410,"time":{"ms":9}, "value":4410}, "user":{"ticks":2960,"time":
{"ms":6}}}, "handles":{"limit":{"hard":4096, "soft":1024}, "open":6}, "info":
{"ephemeral_id":"5d064a8b-991e-4902-be1d-c037a75ba914", "uptime":{"ms":9600031}}, "memstats":
{"gc_next":4194304, "memory_alloc":2837256, "memory_total":116083920}}, "filebeat":{"harvester":
{"open_files":0, "running":0}}, "libbeat":{"config":{"module":{"running":0}}, "pipeline":
{"clients":5, "events":{"active":0}}, "registrar":{"states":{"current":2}}, "system":{"load":
{"1":0.01, "15":0.05, "5":0.02, "norm":{"1":0.01, "15":0.05, "5":0.02}}}}}}
. . . . .

```

## 2、源码安装redis

### 1、安装单机版redis

```

[root@redis ~]# mkdir -p /data/application    ---创建工作目录
[root@redis ~]# wget http://download.redis.io/releases/redis-4.0.9.tar.gz    ---下载redis
[root@redis ~]# tar xzf redis-4.0.9.tar.gz -C /data/application/    ---解压
[root@redis ~]# cd /data/application/
[root@redis application]# mv redis-4.0.9/ redis
[root@redis application]# cd redis/
[root@redis redis]# yum install -y gcc make    #安装编译工具
[root@redis redis]# make

```

注：如果报错请将刚才解压的安装包删除掉，再次重新解压并进行make安装即可。

```
[root@redis redis]# mv redis.conf redis.conf.bak
[root@redis redis]# vim redis.conf      ---修改如下
bind 192.168.203.144      #只监听内网IP
daemonize yes             #开启后台模式将on改为yes
port 6379                 #端口号
dir /data/application/redis/data      #本地数据库存放持久化数据的目录该目录-----需要存在
创建存放数据的目录
[root@redisredis]# mkdir /data/application/redis/data
```

## 配置redis为systemctl启动

```
[root@redis redis]# cd /lib/systemd/system
[root@redis system]# vim redis.service
[Unit]
Description=Redis
After=network.target

[Service]
ExecStart=/data/application/redis/src/redis-server /data/application/redis/redis.conf --
daemonize no
ExecStop=/data/application/redis/src/redis-cli -h 127.0.0.1 -p 6379 shutdown

[Install]
WantedBy=multi-user.target
```

=====

### 参数详解:

- [Unit] 表示这是基础信息
- Description 是描述
- After 是在那个服务后面启动，一般是网络服务启动后启动
- [Service] 表示这里是服务信息
- ExecStart 是启动服务的命令
- ExecStop 是停止服务的指令
- [Install] 表示这是是安装相关信息
- WantedBy 是以哪种方式启动：multi-user.target表明当系统以多用户方式（默认的运行级别）启动时，这个服务需要被自动运行。

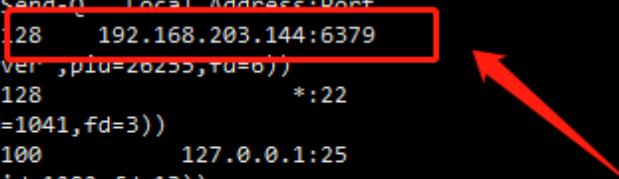
=====

==

## 3、启动服务

```
[root@redis system]# systemctl daemon-reload #重新加载
[root@redis system]# systemctl start redis.service
```

```
[root@localhost ~]# ss -lntp
State      Recv-Q    Send-Q    Local Address:Port      Peer Address:Port
LISTEN     0         128       192.168.203.144:6379    *:*
users:((("redis-server",pid=26255,fd=0))
LISTEN     0         128       *:22                    *:*
users:((("sshd",pid=1041,fd=3))
LISTEN     0         100      127.0.0.1:25           *:*
users:((("master",pid=1202,fd=13))
LISTEN     0         128       :::22                   :::*
users:((("sshd",pid=1041,fd=4))
LISTEN     0         100      :::1:25                 :::*
users:((("master",pid=1202,fd=14))
```



#### 4、测试数据

```
[root@localhost ~]# pwd
/data/application/redis/src
[root@localhost src]# ./redis-cli -h 192.168.203.144 -p 6379
192.168.203.144:6379> ping
PONG
192.168.203.144:6379> set name1 baidu
OK
192.168.203.144:6379> get name1
"baidu"
192.168.203.144:6379>
```

#### 4、测试filebeat传到redis中的数据

```
127.0.0.1:6379> keys *
1) "eureka-log"
127.0.0.1:6379>
127.0.0.1:6379> lrange eureka-log 0 -1
1) "{\"@timestamp\":\"2018-08-07T07:02:55.083Z\",\"@metadata\":{\"beat\":\"\",\"type\":\"doc\",\"version\":\"6.3.2\"},\"source\":\"/opt/work/hcb-eureka/logs/10100/spring.log\",\"prospector\":{\"type\":\"log\"},\"input\":{\"type\":\"log\"},\"beat\":{\"name\":\"hcb\",\"hostname\":\"hcb\",\"version\":\"6.3.2\"},\"host\":{\"name\":\"hcb\"},\"offset\":9944407,\"message\":\"2018-08-07 15:02:50.030 INFO 23157 --- [Eureka-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 0ms\"}"
2) "{\"@timestamp\":\"2018-08-07T07:03:02.084Z\",\"@metadata\":{\"beat\":\"\",\"type\":\"doc\",\"version\":\"6.3.2\"},\"offset\":9944562,\"prospector\":{\"type\":\"log\"},\"input\":{\"type\":\"log\"},\"host\":{\"name\":\"hcb\"},\"beat\":{\"name\":\"hcb\",\"hostname\":\"hcb\",\"version\":\"6.3.2\"},\"message\":\"2018-08-07 15:03:00.030 INFO 23157 --- [Eureka-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 0ms\"},\"source\":\"/opt/work/hcb-eureka/logs/10100/spring.log\"}"
3) "{\"@timestamp\":\"2018-08-07T07:03:17.085Z\",\"@metadata\":{\"beat\":\"\",\"type\":\"doc\",\"version\":\"6.3.2\"},\"beat\":{\"version\":\"6.3.2\",\"name\":\"hcb\",\"hostname\":\"hcb\"},\"host\":{\"name\":\"hcb\"},\"offset\":994717,\"message\":\"2018-08-07 15:03:10.030 INFO 23157 --- [Eureka-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 0ms\"},\"source\":\"/opt/work/hcb-eureka/logs/10100/spring.log\"},\"prospector\":{\"type\":\"log\"},\"input\":{\"type\":\"log\"}"
4) "{\"@timestamp\":\"2018-08-07T07:03:20.086Z\",\"@metadata\":{\"beat\":\"\",\"type\":\"doc\",\"version\":\"6.3.2\"},\"message\":\"2018-08-07 15:03:20.030 INFO 23157 --- [Eureka-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 0ms\"},\"prospector\":{\"type\":\"log\"},\"input\":{\"type\":\"log\"}"
```

### 3、安装Elasticsearch

#### (1)、ES运行依赖jdk8

上传 解压

```
[root@elasticsearch ~]# tar xzf /usr/local/src/jdk-8u131-linux-x64.tar.gz -C /usr/local/
[root@elasticsearch ~]# cd /usr/local
[root@elasticsearch local]# mv jdk-8u131-linux-x64 java
[root@elasticsearch local]# vim /etc/profile
JAVA_HOME=/usr/local/java
export JRE_HOME=/usr/local/java/jre
export CLASSPATH=.:$JAVA_HOME/lib:$JRE_HOME/lib:$CLASSPATH
```

```
export PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin
[root@elasticsearch local]# source /etc/profile
[root@elasticsearch local]# java -version
java version "1.8.0_131"
Java(TM) SE Runtime Environment (build 1.8.0_131-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.131-b11, mixed mode)
[root@elasticsearch local]#
```

## (2)、安装ES

Elasticsearch: 6.5.4 #<https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.5.4.tar.gz>

## 2、安装配置ES----只在第一台操作操作下面的部分

### (1) 创建运行ES的普通用户

```
[root@elasticsearch~]# useradd elsearch
[root@elasticsearch ~]# echo "123456" | passwd --stdin "elsearch"
```

### (2) 安装配置ES

```
[root@elasticsearch ~]# tar xzf elasticsearch-6.5.4.tar.gz -C /usr/local/
[root@elasticsearch ~]# cd /usr/local/elasticsearch-6.5.4/config/
[root@elasticsearch config]# ls
elasticsearch.yml  log4j2.properties  roles.yml  users_roles
jvm.options        role_mapping.yml  users
[root@elasticsearch config]# cp elasticsearch.yml elasticsearch.yml.bak
[root@elasticsearch config]# vim elasticsearch.yml ----找个地方添加如下内容
[root@elasticsearch config]# cat elasticsearch.yml
cluster.name: elk
node.name: elk01
node.master: true
node.data: true
path.data: /data/elasticsearch/data
path.logs: /data/elasticsearch/logs
bootstrap.memory_lock: false
bootstrap.system_call_filter: false
network.host: 0.0.0.0
http.port: 9200
discovery.zen.ping.unicast.hosts: ["192.168.203.142"]
http.cors.enabled: true
http.cors.allow-origin: "*"

[root@elasticsearch config]#
```

cluster.name	集群名称，各节点配成相同的集群名称。
node.name	节点名称，各节点配置不同。
node.master	指示某个节点是否符合成为主节点的条件。

<code>node.data</code>	指示节点是否为数据节点。数据节点包含并管理索引的一部分。
<code>path.data</code>	数据存储目录。
<code>path.logs</code>	日志存储目录。
<code>bootstrap.memory_lock</code>	内存锁定，是否禁用交换。
<code>bootstrap.system_call_filter</code>	系统调用过滤器。
<code>network.host</code>	绑定节点IP。
<code>http.port</code>	端口。
<code>discovery.zen.ping.unicast.hosts</code>	提供其他 Elasticsearch 服务节点的单点广播发现功能。
<code>discovery.zen.minimum_master_nodes</code>	集群中可工作的具有Master节点资格的最小数量，官方的推荐值是 $(N/2)+1$ ，其中N是具有master资格的节点的数量。
<code>discovery.zen.ping_timeout</code>	节点在发现过程中的等待时间。
<code>discovery.zen.fd.ping_retries</code>	节点发现重试次数。
<code>http.cors.enabled</code>	是否允许跨源 REST 请求，用于允许head插件访问ES。
<code>http.cors.allow-origin</code>	允许的源地址。

### (3) 设置JVM堆大小

```
[root@elasticsearch config]# vim jvm.options      ----将
-Xms1g      ----修改成 -Xms2g
-Xmx1g      ----修改成 -Xms2g

或者：
推荐设置为4G，请注意下面的说明：
sed -i 's/-Xms1g/-Xms4g/' /usr/local/elasticsearch-6.5.4/config/jvm.options
sed -i 's/-Xmx1g/-Xmx4g/' /usr/local/elasticsearch-6.5.4/config/jvm.options
```

注意：确保堆内存最小值（Xms）与最大值（Xmx）的大小相同，防止程序在运行时改变堆内存大小。堆内存大小不要超过系统内存的50%

### (4) 创建ES数据及日志存储目录

```
[root@elasticsearch ~]# mkdir -p /data/elasticsearch/data      (/data/elasticsearch)
[root@elasticsearch ~]# mkdir -p /data/elasticsearch/logs      (/log/elasticsearch)

[root@elasticsearch ~]# chown -R elasticsearch:elasticsearch /data/elasticsearch
[root@elasticsearch ~]# chown -R elasticsearch:elasticsearch /usr/local/elasticsearch-6.5.4
```

## 3、系统优化

### (1) 增加最大文件打开数

永久生效方法：

```
echo "* - nofile 65536" >> /etc/security/limits.conf
```

### (2) 增加最大进程数

```
[root@elasticsearch ~]# vim /etc/security/limits.conf    ---在文件最后面添加如下内容
* soft nofile 65536
* hard nofile 131072
* soft nproc 2048
* hard nproc 4096
更多的参数调整可以直接用这个
```

解释:

soft xxx : 代表警告的设定, 可以超过这个设定值, 但是超过后会有警告。

hard xxx : 代表严格的设定, 不允许超过这个设定的值。

nofile : 是每个进程可以打开的文件数的限制

nproc : 是操作系统级别对每个用户创建的进程数的限制

### (3) 增加最大内存映射数

```
[root@elasticsearch ~]# vim /etc/sysctl.conf    ---添加如下
vm.max_map_count=262144
vm.swappiness=60
[root@elasticsearch ~]# sysctl -p
解释: 在内存不足的情况下, 使用交换空间。
```

```
[root@elasticsearch ~]# sysctl -w vm.max_map_count=262144
增大用户使用内存的空间(临时)
```

## 4、启动ES

```
[root@elasticsearch ~]# su - elsearch
Last login: Sat Aug  3 19:48:59 CST 2019 on pts/0
[root@elasticsearch ~]$ cd /usr/local/elasticsearch-6.5.4/
[root@elasticsearch elasticsearch-6.5.4]$ ./bin/elasticsearch  #先启动看看报错不, 需要多等一会
终止之后
[root@elasticsearch elasticsearch-6.5.4]$ nohup ./bin/elasticsearch &  #放后台启动
[1] 11462
nohup: ignoring input and appending output to 'nohup.out'
[root@elasticsearch elasticsearch-6.5.4]$ tail -f nohup.out  #看一下是否启动
或者:
su - elsearch -c "cd /usr/local/elasticsearch-6.5.4 && nohup bin/elasticsearch &"
```

测试: 浏览器访问<http://172.16.203.144:9200>

## 5.安装配置head监控插件 (Web前端)

### (1) 安装node



```
[root@elasticsearch ~]# wget https://npm.taobao.org/mirrors/node/latest-v4.x/node-v4.4.7-linux-x64.tar.gz
[root@elasticsearch ~]# tar -xzf node-v4.4.7-linux-x64.tar.gz -C /usr/local
[root@elasticsearch ~]# vim /etc/profile #添加如下变量
NODE_HOME=/usr/local/node-v4.4.7-linux-x64
PATH=$NODE_HOME/bin:$PATH
export NODE_HOME PATH
[root@elasticsearch ~]# source /etc/profile
[root@elasticsearch ~]# node --version #检查node版本号
v4.4.7
```

## (2) 下载head插件

```
[root@elasticsearch ~]# wget https://github.com/mobz/elasticsearch-head/archive/master.zip
[root@elasticsearch ~]# cp master.zip /usr/local/
[root@elasticsearch ~]# yum -y install unzip
[root@elasticsearch ~]# cd /usr/local
[root@elasticsearch ~]# unzip master.zip
```

## (3) 安装grunt

```
[root@elasticsearch ~]# cd elasticsearch-head-master/
[root@elasticsearch elasticsearch-head-master]# npm install -g grunt-cli #时间会很长
[root@elasticsearch elasticsearch-head-master]# grunt --version #检查grunt版本号
grunt-cli v1.3.2
```

## (4) 修改head源码

```
[root@elasticsearch elasticsearch-head-master]# vim /usr/local/elasticsearch-head-master/Gruntfile.js (95左右)
```



```

    files: [ 'Gruntfile.js' ]
  },
  connect: {
    server: {
      options: {
        port: 9100,
        base: '.',
        keepalive: true,
        hostname: '*'
      }
    }
  }
});

```

操作。保如果在一台机器上面可以不修改下面的持原来的就可以了

如果是集群需要修改如下信息:

```

4358         this.prefs = services.Preferences.instance();
4359         this.base_uri = this.config.base_uri || this.prefs.get("app
-base_uri") || "http://192.168.246.234:9200";
4360         if( this.base_uri.charAt( this.base_uri.length - 1 ) !== "/"
" ) {
4361             // XHR request fails if the URL is not ending with

```

原本是<http://localhost:9200>，如果head和ES不在同一个节点，注意修改成ES的主节点的IP地址

#### (5) 下载head必要的文件

```

[root@elasticsearch ~]# wget
https://github.com/Medium/phantomjs/releases/download/v2.1.1/phantomjs-2.1.1-linux-
x86_64.tar.bz2
[root@elasticsearch ~]# yum -y install bzip2
[root@elasticsearch ~]# tar -jxf phantomjs-2.1.1-linux-x86_64.tar.bz2 -C /tmp/ #解压

```

#### (6) 运行head

```

[root@elasticsearch ~]# cd /usr/local/elasticsearch-head-master/
[root@elasticsearch elasticsearch-head-master]# npm install
...
grunt-contrib-jasmine@1.0.3 node_modules/grunt-contrib-jasmine
├─ sprintf-js@1.0.3
├─ lodash@2.4.2
├─ es5-shim@4.5.13
├─ chalk@1.1.3 (escape-string-regexp@1.0.5, supports-color@2.0.0, ansi-styles@2.2.1, strip-
ansi@3.0.1, has-ansi@2.0.0)
├─ jasmine-core@2.99.1
├─ rimraf@2.6.3 (glob@7.1.4)
└─ grunt-lib-phantomjs@1.1.0 (eventemitter2@0.4.14, semver@5.7.0, temporary@0.0.8, phan
[root@elasticsearch elasticsearch-head-master]# nohup grunt server &
[root@elasticsearch elasticsearch-head-master]# tail -f nohup.out
Running "connect:server" (connect) task
Waiting forever...
Started connect web server on http://localhost:9100

```

#### 测试

访问<http://172.16.203.142:9100>

← → ↻ 不安全 | 192.168.203.142:9100

应用 Gmail YouTube 地图 在线翻译\_有道 Elastic Stack(ELK)... 1: ELK概述和安装...

**Elasticsearch** http://192.168.203.142:9200/ 连接 elk 集群健康值: yellow (6 of 11)

概览 索引 数据浏览 基本查询 [+] 复合查询 [+] 集群概览 集群排序 Sort Indices View Aliases Index Filter

**logstash1-2019.12.11**  
size: 74.9ki (74.9ki)  
docs: 34 (34)  
信息 动作

**.kibana\_1**  
size: 19.5ki (19.5ki)  
docs: 4 (4)  
信息 动作  
.kibana X

**Unassigned** 0 1 2 3 4

★ **elk01** 信息 动作 0 1 2 3 4 0

## 4、安装logstash

Logstash运行同样依赖jdk 安装jdk 安装logstash

### 1、上传 解压缩

```
# tar /usr/local/src/logstash-5.3.1.tar.gz -C /usr/local/
# ln -s /usr/local/logstash-5.3.1 /usr/local/logstash
```

测试logstash是否可用

```
[root@log ~]# /usr/local/logstash/bin/logstash -e 'input { stdin { } } output { stdout { } }'
hello world
Sending Logstash logs to /usr/local/logstash/logs which is now configured via log4j2.properties
[2019-12-11T16:25:17,120][INFO ][logstash.setting.writabledirectory] Creating directory
{:setting=>"path.queue", :path=>"/usr/local/logstash/data/queue"}
[2019-12-11T16:25:17,173][INFO ][logstash.setting.writabledirectory] Creating directory
{:setting=>"path.dead_letter_queue", :path=>"/usr/local/logstash/data/dead_letter_queue"}
[2019-12-11T16:25:18,864][WARN ][logstash.config.source.multilocal] Ignoring the 'pipelines.yml'
file because modules or command line options are specified
[2019-12-11T16:25:18,960][INFO ][logstash.runner] Starting Logstash
{"logstash.version"=>"6.5.4"}
[2019-12-11T16:25:19,106][INFO ][logstash.agent] No persistent UUID file found.
Generating new UUID {:uuid=>"74269a89-f497-4147-8bb8-2f313da11bd8",
:path=>"/usr/local/logstash/data/uuid"}
[2019-12-11T16:25:29,560][INFO ][logstash.pipeline] Starting pipeline
{:pipeline_id=>"main", "pipeline.workers"=>1, "pipeline.batch.size"=>125,
"pipeline.batch.delay"=>50}
[2019-12-11T16:25:31,002][INFO ][logstash.pipeline] Pipeline started successfully
{:pipeline_id=>"main", :thread=>"#<Thread:0xb5716f3 run>"}
The stdin plugin is now waiting for input:
[2019-12-11T16:25:31,354][INFO ][logstash.agent] Pipelines running {:count=>1,
:running_pipelines=>[:main], :non_running_pipelines=>[]}
{
  "host" => "log",
```

```
"@version" => "1",
"message" => "hello world",
"@timestamp" => 2019-12-11T08:25:31.445Z
}
```

创建主配置文件

```
# vim /usr/local/logstash/config/logstash-simple.conf
input { stdin { } }
output {
  stdout { codec=> rubydebug }
}
```

```
[2019-12-11T16:30:11,297][INFO ][logstash.agent          ] Successfully started Logstash API
endpoint { :port=>9600}
{
  "host" => "log",
  "@timestamp" => 2019-12-11T08:30:09.450Z,
  "@version" => "1",
  "message" => "hello wanghaibing ye"
}
{
  "host" => "log",
  "@timestamp" => 2019-12-11T08:30:09.571Z,
  "@version" => "1",
  "message" => ""
}
{
  "host" => "log",
  "@timestamp" => 2019-12-11T08:30:09.571Z,
  "@version" => "1",
  "message" => ""
}
```

此时说明我们的logstash是完全没有问题了，可以进行日志收集了

1. 创建配置文件获取redis日志的数据：

配置文件如下：

```
# vim /usr/local/logstash/config/redis-spring.conf
input {
  redis {
    port => "6379"
    host => "192.168.203.135"
    data_type => "list"
    type => "log"
    key => "eureka-log"
  }
}
output {
  elasticsearch {
```

通过配置文件启动服务查看效果：

```
# /usr/local/logstash/bin/logstash -f /usr/local/logstash/config/redis-spring.conf
```

此时我们再去查看reids中key：（此时已经没有数据了，数据已经被logstash取完）

使用curl 查看ES是否接受到数据

```
curl http://192.168.203.135:9200/_search?pretty
```

此时说明我们logstash从redis中取数据，在把数据推到ES中是ok的！

## 5、安装kibana

```
[root@kibana ~]# tar -xvf kibana-6.5.4-linux-x86_64.tar.gz -C /usr/local/
[root@kibana ~]# cd /usr/local/
[root@kibana kibana-6.5.4-linux-x86_64]# vim /usr/local/kibana-5.3.1-linux-
x86_64/config/kibana.yml
server.port: 5601 #开启默认端口5601
server.host: "192.168.203.134" #kibana站点IP
elasticsearch.url: http://192.168.203.142:9200 #只想ES服务所在IP Port
kibana.index: ".kibana"
```

后台启动kibana:

```
[root@kibana kibana-6.5.4-linux-x86_64]# nohup ./bin/kibana &
[root@kibana kibana-6.5.4-linux-x86_64]# tail -f nohup.out
[root@kibana kibana-6.5.4-linux-x86_64]# ss -lntp
users:((("master",pid=1093,fd=13))
LISTEN    0      128    192.168.203.134:5601          *:*
```



kibana

Management / Kibana

Index Patterns Saved Objects Advanced Settings

Discover

Visualize

Dashboard

Timeline

Dev Tools

Management

No default index pattern. You must select or create one to continue.

## Configure an index pattern

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

Index name or pattern

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

logstash-\*

2

☒ Index contains time-based events

Time-field name

refresh fields

@timestamp

3

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

By default, searches against any time-based index pattern that contains a wildcard will automatically be expanded to query only the indices that contain data within the currently selected time range. Searching against the index pattern `logstash-*` will actually query elasticsearch for the specific matching indices (e.g. `logstash-2015.12.27`) that fall within the current time range.

☐ Use event times to create index names (DEPRECATED)

4

Create

