[**http://www.foxwho.com/article/156**](http://www.foxwho.com/article/156)

**https://blog.csdn.net/fenglailea/article/details/78711293**

**elasticsearch6.x.x logstash6.x.x kibana6.x.x Filebeat6.x 配置及安装 CentOS7.3或更高**

2017年12月04日 16:47:02

阅读数：1069

环境说明   
系统:Centos7.3   
最低内存：4G   
JAVA: 8.XXXX

默认已root用户执行   
风.foxiswho

**centos 防火墙设置**

要开启端口，否则，除本机外，其他任何机器不能访问

**方式一 开放9200 端口(其他端口按照本案例添加)**

firewall-cmd --zone=public --add-port=9200/tcp --permanent

**方式二 关闭防火墙**

systemctl stop firewalld

**JAVA**

先安装java   
查看java 是否已安装过了

java -version

如果没有那么下载，找最新版本的下载   
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

wget --no-check-certificate --no-cookies --header "Cookie: oraclelicense=accept-securebackup-cookie" http://download.oracle.com/otn-pub/java/jdk/8u151-b12/e758a0de34e24606bca991d704f6dcbf/jdk-8u151-linux-x64.tar.gz

解压缩及复制到目录

tar zxvf jdk-\*

#新建目录

mkdir -p /usr/java

#复制解压缩后的文件到该目录

mv jdk1.8.0\_151 /usr/java/

* 1

注意：jdk解压缩后的目录根据你选择版本显示的，不一定和本案例一样

**设置JAVA 环境变量**

echo "export JAVA\_HOME=/usr/java/jdk1.8.0\_151

export JRE\_HOME=\$JAVA\_HOME/jre #tomcat需要

export PATH=\$JAVA\_HOME/bin:\$PATH

export CLASSPATH=.:\$JAVA\_HOME/lib/dt.jar:\$JAVA\_HOME/lib/tools.jar" > /etc/profile.d/java.sh

应用java 环境变量生效（注意 .后面有一个空格）

source /etc/profile

source /etc/bashrc

注意，这里使用的是source，不是. 因为source 执行后当前进程内就可以使用该环境变量

查看java 版本号

java -version

* 1

**新建用户及用户组**

#新建用户组

groupadd elasticsearch

#建立用户

useradd -g elasticsearch -m elasticsearch

#创建密码

passwd elasticsearch

设置管理员或用户组权限   
设置用户有 visudo权限,即sudo   
执行命令

visudo

在root 那行增加 elasticsearch一行，如下所示

root ALL=(ALL) ALL

elasticsearch ALL=(ALL) ALL

保存退出

**设置内核配置**

/etc/security/limits.conf

echo "elasticsearch hard nofile 65536

elasticsearch soft nofile 65536 ">> /etc/security/limits.conf

/etc/sysctl.conf

echo "vm.max\_map\_count=655360">> /etc/sysctl.conf

应用并生效

sysctl -p

**设置 elasticsearch 环境变量**

echo "export ES\_HOME=/home/elasticsearch/elasticsearch-5.6.1

export PATH=\$ES\_HOME/bin:\$PATH" > /etc/profile.d/elasticsearch.sh

**设置 elasticsearch 环境变量**

echo "export ES\_HOME=/home/elasticsearch/elasticsearch-6.0.0

export PATH=\$ES\_HOME/bin:\$PATH" > /etc/profile.d/elasticsearch.sh

**以下操作都是以 elasticsearch 用户操作**

**以下操作都是以 elasticsearch 用户操作**

**以下操作都是以 elasticsearch 用户操作**

如果使用root用户，那么elasticsearch 是无法启动的

**切换用户**

在 root 用户下切换,或者你也可以直接用 elasticsearch 用户登录

su elasticsearch

#切换到elasticsearch 用户名目录下

cd ~

**elasticsearch 配置安装**

下载 <https://www.elastic.co/downloads/elasticsearch>

wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.0.0.tar.gz

解压缩

tar -zxvf elasticsearch-6.0.0.tar.gz

设置 局域网或者其他机器可以访问，如果不设置，那么直接装有elasticsearch这台机器才可以访问.   
编辑 config/elasticsearch.yml 文件

vim elasticsearch-6.0.0/config/elasticsearch.yml

找到类似#network.host: 192.168.0.1地方，修改为:

network.host: 0.0.0.0

注意： host:与IP地址 0中间有个空格不能删除，否则报错

**elasticsearch 启动**

cd elasticsearch-6.0.0

./bin/elasticsearch #前台运行

或者

./bin/elasticsearch -d #后台运行

**elasticsearch 设置后台启动**

使用nohup 配合

nohup bin/elasticsearch -d &

**elasticsearch 关闭**

ps -ef |grep /elasticsearch|awk '{print $2}'|xargs kill -9

**测试**

浏览器访问

http://10.1.5.66:9200/

如果出现以下内容表示安装成功

{

"name": "yhwzDyT",

"cluster\_name": "elasticsearch",

"cluster\_uuid": "rnivNLavQqOrdFdrUrxmlw",

"version": {

"number": "6.0.0",

"build\_hash": "8f0685b",

"build\_date": "2017-11-10T18:41:22.859Z",

"build\_snapshot": false,

"lucene\_version": "7.0.1",

"minimum\_wire\_compatibility\_version": "5.6.0",

"minimum\_index\_compatibility\_version": "5.0.0"

},

"tagline": "You Know, for Search"

}

**elasticsearch 配置文件详解**

<http://www.cnblogs.com/xiaochina/p/6855591.html>

### 集群

./bin/elasticsearch -Ecluster.name=my\_cluster -E node.name=node1 –d

./bin/ elasticsearch -Ecluster.name=my\_cluster -Epath.data=my\_cluster\_node1 -Enode.name=node1 -Ehttp.port=5200 -d

         配置文件中给出了三种配置高性能集群拓扑结构的模式,如下：   
            # 1. 如果你想让节点从不选举为主节点,只用来存储数据,可作为数据节点   
            # node.master: true   
            # node.data: false  
            # node.ingest: true  
  
            # 2. 如果想让节点成为主节点,且不存储任何数据,并保有空闲资源,可作为协调器   
            # node.master：true   
            # node.data：false   
            # node.ingest：false   
  
            # 3. 如果想让节点既不称为主节点,又不成为数据节点,那么可将他作为摄取节点,从节点中获取数据,生成搜索结果等   
            # node.master: false   
            # node.data: false   
            # node.ingest: true  
  
            # 4. 仅作为协调器   
            # node.master: false   
            # node.data: false

            # node.ingest: false

**logstash 配置及安装**

下载 <https://www.elastic.co/downloads/logstash>

wget https://artifacts.elastic.co/downloads/logstash/logstash-6.0.0.tar.gz

* 1

解压缩

tar -zxvf logstash-6.0.0.tar.gz

* 1

**测试**

测试是否安装成功

~/logstash-6.0.0/bin/logstash -e 'input { stdin { } } output { stdout {}}'

* 1

如果输出类似如下表示安装成功

The stdin plugin is now waiting for input:

[2017-05-16T21:48:15,233][INFO ][logstash.agent ] Successfully started Logstash API endpoint {:port=>9600}

* 1
* 2

**插件**

**中文分词插件analysis-ik**

下载地址 <https://github.com/medcl/elasticsearch-analysis-ik/releases>

#到 elasticsearch-6.0.0/plugins 目录下

cd ~/elasticsearch-6.0.0/plugins

wget https://github.com/medcl/elasticsearch-analysis-ik/releases/download/v6.0.0/elasticsearch-analysis-ik-6.0.0.zip

解压缩

unzip elasticsearch-analysis-ik-6.0.0.zip

#解压缩后目录名修改为analysis-ik

mv elasticsearch analysis-ik

删除压缩文件

rm -rf elasticsearch-analysis-ik-6.0.0.zip

按正常启动就可以了，如果要立即生效，那么 elasticsearch 必须要重启   
设置词库

cd ~/elasticsearch-6.0.0/

vim plugins/analysis-ik/config/IKAnalyzer.cfg.xml

* 1
* 2

如果没有词库下面这个不需要设置。   
如果你有词库，   
那么ext\_dict 这一行修改为如下:

<entry key="ext\_dict">main.dic;extra\_main.dic</entry>

* 1

注意 ext\_dict中的字库根据你的需要自行添加字库

这个时候需要重启 elasticsearch 插件才能生效

**热更新 IK 分词使用方法 来自官方**

<https://github.com/medcl/elasticsearch-analysis-ik>

目前该插件支持热更新 IK 分词，通过上文在 IK 配置文件中提到的如下配置

<!--用户可以在这里配置远程扩展字典 -->

<entry key="remote\_ext\_dict">location</entry>

<!--用户可以在这里配置远程扩展停止词字典-->

<entry key="remote\_ext\_stopwords">location</entry>

其中 location 是指一个 url，比如 <http://yoursite.com/getCustomDict>，该请求只需满足以下两点即可完成分词热更新。

该 http 请求需要返回两个头部(header)，一个是 Last-Modified，一个是 ETag，这两者都是字符串类型，只要有一个发生变化，该插件就会去抓取新的分词进而更新词库。   
- 该 http 请求返回的内容格式是一行一个分词，换行符用 \n 即可。   
- 满足上面两点要求就可以实现热更新分词了，不需要重启 ES 实例。

可以将需自动更新的热词放在一个 UTF-8 编码的 .txt 文件里，放在 nginx 或其他简易 http server 下，当 .txt 文件修改时，http server 会在客户端请求该文件时自动返回相应的 Last-Modified 和 ETag。可以另外做一个工具来从业务系统提取相关词汇，并更新这个 .txt 文件。

**分词测试**

curl -XPUT "http://localhost:9200/index"

* 1

**创建 mapping**

**创建测试信息**

curl -H "Content-Type: application/json;charset=UTF-8" -XPOST http://localhost:9200/index/fulltext/1 -d' {"content":"美国留给伊拉克的是个烂摊子吗"} '

curl -H "Content-Type: application/json;charset=UTF-8" -XPOST http://localhost:9200/index/fulltext/2 -d' {"content":"公安部：各地校车将享最高路权"} '

curl -H "Content-Type: application/json;charset=UTF-8" -XPOST http://localhost:9200/index/fulltext/3 -d' {"content":"中韩渔警冲突调查：韩警平均每天扣1艘中国渔船"} '

curl -H "Content-Type: application/json;charset=UTF-8" -XPOST 'http://localhost:9200/index/fulltext/4' -d' {"content":"中国驻洛杉矶领事馆遭亚裔男子枪击 嫌犯已自首"}'

**查询**

curl -XPOST http://localhost:9200/index/fulltext/\_search -d'

{

"query" : { "match" : { "content" : "中国" }},

"highlight" : {

"pre\_tags" : ["<tag1>", "<tag2>"],

"post\_tags" : ["</tag1>", "</tag2>"],

"fields" : {

"content" : {}

}

}

}

'

结果如下

{

"took": 418,

"timed\_out": false,

"\_shards": {

"total": 5,

"successful": 5,

"skipped": 0,

"failed": 0

},

"hits": {

"total": 2,

"max\_score": 0.2876821,

"hits": [

{

"\_index": "index",

"\_type": "fulltext",

"\_id": "5",

"\_score": 0.2876821,

"\_source": {

"content": "中国驻洛杉矶领事馆遭亚裔男子枪击 嫌犯已自首"

},

"highlight": {

"content": [

"<tag1>中国</tag1>驻洛杉矶领事馆遭亚裔男子枪击 嫌犯已自首"

]

}

},

{

"\_index": "index",

"\_type": "fulltext",

"\_id": "4",

"\_score": 0.2876821,

"\_source": {

"content": "中国驻洛杉矶领事馆遭亚裔男子枪击 嫌犯已自首"

},

"highlight": {

"content": [

"<tag1>中国</tag1>驻洛杉矶领事馆遭亚裔男子枪击 嫌犯已自首"

]

}

}

]

}

}

**elasticsearch head 根据你的需要是否要安装，我的没有安装**

**安装 notejs npm 需要使用 root用户安装,安装完成后 切换回 elasticsearch用户**

先要安装 npm 也就是notejs   
<http://blog.csdn.net/fenglailea/article/details/56484144>   
<https://github.com/nodesource/distributions#debinstall> 推荐此文章   
先安装支持

sudo yum install -y gcc-c++ make

* 1

Centos

sudo curl -sL https://rpm.nodesource.com/setup\_8.x | bash -

sudo yum install -y nodejs

* 1
* 2

UBUNTU

sudo curl -sL https://deb.nodesource.com/setup\_8.x | bash -

sudo apt-get install -y nodejs

* 1
* 2

**安装 elasticsearch head**

来自：<http://blog.csdn.net/fenglailea/article/details/52934263>   
此插件已独立运行。新的安装方式

cd ~/

#方式一

#以下用git 拉取，如果没有请换方式二

git clone git://github.com/mobz/elasticsearch-head.git

#方式二

wget https://github.com/mobz/elasticsearch-head/archive/master.zip -O elasticsearch-head.zip

unzip elasticsearch-head.zip

mv elasticsearch-head-master elasticsearch-head

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8

**配置 elasticsearch.yml 和Gruntfile.js**

修改elasticsearch.yml

vim ~/elasticsearch-6.0.0/config/elasticsearch.yml

* 1

加入以下内容：

http.cors.enabled: true

http.cors.allow-origin: "\*"

* 1
* 2

修改Gruntfile.js

vim ~/elasticsearch-head/Gruntfile.js

* 1

找到下面配置修改为：

connect: {

server: {

options: {

hostname: '0.0.0.0',

port: 9100,

base: '.',

keepalive: true

}

}

}

注意： 设置 hostname: 主要为了其他IP可以访问，否则只有 本机可以访问

**启动 elasticsearch head**

如果 npm 没有安装安装 <http://blog.csdn.net/fenglailea/article/details/52934263> 教程安装

cd ~/elasticsearch-head

#使用国内镜像

sudo npm install -g cnpm --registry=https://registry.npm.taobao.org

sudo npm install

grunt server

* 1
* 2
* 3
* 4
* 5

如果使用sudo npm install命令安装时报错 请看后面的 FAQ   
访问地址

http://localhost:9100/

* 1

**kibana 配置及安装**

下载

wget https://artifacts.elastic.co/downloads/kibana/kibana-6.0.0-linux-x86\_64.tar.gz

* 1

解压缩

tar -zxvf kibana\*.tar.gz

mv kibana-6.0.0-linux-x86\_64 kibana-6.0.0

* 1
* 2

设置 局域网或者其他机器可以访问，如果不设置，那么直接装有elasticsearch这台机器才可以访问.   
编辑 config/elasticsearch.yml 文件

vim kibana-6.0.0/config/kibana.yml

* 1

找到类似#server.host: "localhost"地方，修改为:

server.host: "0.0.0.0"

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

* 1
* 2

注意： host:与IP地址 0中间有个空格不能删除，否则报错

**kibana 启动**

cd kibana-6.0.0

bin/kibana

* 1
* 2

**kibana 设置后台启动**

使用nohup 配合

nohup bin/kibana -d &

* 1

**kibana 关闭**

ps -ef |grep /kibana |awk '{print $2}'|xargs kill -9

* 1

**x-pack 插件**

**elasticsearch 安装 x-pack 插件**

cd elasticsearch-6.0.0

bin/elasticsearch-plugin install x-pack

如果提示Continue with installation? [y/N]输入y回车

**设置登陆用户及密码**

cd elasticsearch-6.0.0

bin/x-pack/setup-passwords auto

手动设置各类密码（包括elastic、kibana、logstash 都是密码：changme）

./setup-passwords interactive

Please confirm that you would like to continue [y/N] 输入y回车   
系统会自动创建3个用户及其密码,密码是系统随机创建的，每个人的都不一样   
例如:

Changed password for user kibana

PASSWORD kibana = g2MAq\_KTK!t~qmkX0-Ke

Changed password for user logstash\_system

PASSWORD logstash\_system = ssDX7\*y6tt8Od^tw0#wg

Changed password for user elastic

PASSWORD elastic = fGw6z\_xn%%xp\_-jp3bd?

**kibana 安装 x-pack 插件**

cd kibana-6.0.0

bin/kibana-plugin install x-pack

在 kibana 配置文件中加入

elasticsearch.username: "kibana"

elasticsearch.password: "你上面得到的密码"

最后重启kibana

**x-pack访问**

<http://localhost:5601/>   
或   
<http://10.1.5.66:5601/>

登陆时就用 elastic用户及密码登陆

**x-pack 插件 登陆不验证**

不需要登陆验证，则在es和kibana的配置里分别加入

xpack.security.enabled: false

**FAQ**

**Failed at the phantomjs-prebuilt@2.1.15 install script ‘node install.js’.**

npm ERR! phantomjs-prebuilt@2.1.15 install: `node install.js`

npm ERR! Exit status 1

npm ERR!

npm ERR! Failed at the phantomjs-prebuilt@2.1.15 install script 'node install.js'.

npm ERR! Make sure you have the latest version of node.js and npm installed.

npm ERR! If you do, this is most likely a problem with the phantomjs-prebuilt package,

解决方法

sudo npm install phantomjs-prebuilt@2.1.15 --ignore-scripts

参考：<https://stackoverflow.com/questions/40992231/failed-at-the-phantomjs-prebuilt2-1-13-install-script-node-install-js>

**grunt-cli: The grunt command line interface (v1.2.0)**

grunt-cli: The grunt command line interface (v1.2.0)

Fatal error: Unable to find local grunt.

If you're seeing this message, grunt hasn't been installed locally to

your project. For more information about installing and configuring grunt,

please see the Getting Started guide:

http://gruntjs.com/getting-started

安装 grunt

npm install -g grunt

**npm WARN elasticsearch-head@0.0.0 license should be a valid SPDX license expression**

一个开源软件声明, 将 elasticsearch-head/package.json 中的 license后面Apache2修改为Apache-2.0

**curl -XPOST “type” : “security\_exception”,”reason” : “missing authentication token for REST request [/**

报类似这样错误时，只要在curl 后面加入-u elastic:密码 即可解决

curl -u elastic:fGw6z\_xn%%xp\_-jp3bd? -H "Content-Type: application/json;charset=UTF-8" -XPOST 'http://localhost:9200/index/fulltext/5' -d' {"content":"中国驻洛杉矶领事馆遭亚裔男子枪击 嫌犯已自首"}'

* 1

**配置案例 ELK方式**

测试案例之前 先要启动 elasticsearch   
以 www.lanmps.com 站点日志为案例

**创建配置**

cd ~/logstash-6.0.0

mkdir -p etc

vim etc/nginx-lanmps.conf

nginx-lanmps.conf 文件内容如下

input {

file {

type => "nginx\_lanmps"

#监听的文件

path => [

"/www/wwwLogs/www.lanmps.com.\*.log"

]

#排除不想监听的文件

exclude => ["\*.gz", "access.log"]

#设置新事件的标志

#delimiter => "\n"

#添加自定义的字段

#add\_field => {"test"=>"test"}

#增加标签

#tags => "tag1"

#设置多长时间扫描目录，发现新文件

discover\_interval => 15

#设置多长时间检测文件是否修改

stat\_interval => 1

#监听文件的起始位置，默认是end

start\_position => "beginning"

#监听文件读取信息记录的位置

#sincedb\_path => "/home/elasticsearch/elk/sincedb\_trade.txt"

#设置多长时间会写入读取的位置信息

#sincedb\_write\_interval => 15

}

}

filter {

grok {

match=>[

"message",

"%{IPORHOST:client\_ip} %{USER:ident} %{USER:auth} \[%{HTTPDATE:timestamp}\] \"(?:%{WORD:verb} %{NOTSPACE:request}(?: HTTP/%{NUMBER:http\_version})?|-)\" %{NUMBER:response} (?:%{NUMBER:bytes}|-) (%{QS:referrer}|-) (%{QS:agent}|-) \"(%{WORD:x\_forword}|-)\" (%{HOSTNAME:domain}|-) (%{WORD:request\_method}|-) (%{QS:uri}|-) (%{QS:query\_string}) (%{NUMBER:upstream\_response}|-) (%{WORD:upstream\_cache\_status}|-) (%{URIHOST:upstream\_host}|-) (%{USERNAME:upstream\_response\_time}) > (%{USERNAME:response\_time}) %{QS:upstream\_content\_type} (?:%{QS:request\_body}|-)"

]

}

mutate {

gsub => [

# 将filed\_name\_2字段中所有"\","?","#","-"转换为"."

"agent", "\"", "",

"upstream\_content\_type", "\"", "",

"query\_string", "\"", "",

"uri", "\"", "",

"request\_body", "\"", "",

"referrer", "\"", ""

]

}

#匹配模式 message是每段读进来的日志，IP、HTTPDATE、WORD、NOTSPACE、NUMBER都是patterns/grok-patterns中定义好的正则格式名称，对照上面的日志进行编写,冒号，(?:%{USER:ident}|-)这种形式是条件判断，相当于程序里面的二目运算。如果有双引号""或者[]号，需要在前面加\进行转义。

#kv {

# source => "request"

# field\_split => "&?"

# value\_split => "="

#}

#再单独将取得的URL、request字段取出来进行key-value值匹配，需要kv插件。提供字段分隔符"&?"，值键分隔符"="，则会自动将字段和值采集出来。

#urldecode {

# all\_fields => true

#}

#把所有字段进行urldecode（显示中文）

#定义时间戳的格式

date {

match => [ "timestamp", "yyyy-MM-dd-HH:mm:ss" ]

locale => "cn"

}

}

output {

elasticsearch {

hosts => "localhost:9200"

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

template => "/home/elasticsearch/logstash-6.0.0/template/nginx\_log.json"

template\_name => "nginx\_log-%{type}"

template\_overwrite => true

#安装了 x-pack后会用到 用户名及密码

#user => elastic

#password => changeme

}

stdout { codec => rubydebug }

}

建立模版文件

cd ~/logstash-6.0.0/

#目录

mkdir -p template

vim template/nginx\_log.json

nginx\_log.json 内容如下

{

"template": "nginx\_log-\*",

"settings": {

"index.number\_of\_shards": 5,

"number\_of\_replicas": 1,

"index.refresh\_interval": "60s"

},

"mappings": {

"\_default\_": {

"\_all": {

"enabled": true

},

"dynamic\_templates": [

{

"string\_fields": {

"match": "\*",

"match\_mapping\_type": "string",

"mapping": {

"type": "string",

"index": "not\_analyzed",

"omit\_norms": true,

"doc\_values": true,

"fields": {

"raw": {

"type": "string",

"index": "not\_analyzed",

"ignore\_above": 256,

"doc\_values": true

}

}

}

}

}

],

"properties": {

"@version": {

"type": "string",

"index": "not\_analyzed"

},

"@timestamp": {

"type": "date",

"format": "strict\_date\_optional\_time||epoch\_millis",

"doc\_values": true

},

"client\_ip": {

"type": "string",

"index": "not\_analyzed"

},

"ident": {

"type": "string",

"index": "not\_analyzed"

},

"auth": {

"type": "string",

"index": "not\_analyzed"

},

"verb": {

"type": "string",

"index": "not\_analyzed"

},

"request": {

"type": "string",

"index": "not\_analyzed"

},

"http\_version": {

"type": "string",

"index": "not\_analyzed"

},

"response": {

"type": "string",

"index": "not\_analyzed"

},

"bytes": {

"type": "string",

"index": "not\_analyzed"

},

"referrer": {

"type": "string",

"index": "not\_analyzed"

},

"agent": {

"type": "string",

"index": "not\_analyzed"

},

"x\_forword": {

"type": "string",

"index": "not\_analyzed"

},

"domain": {

"type": "string",

"index": "not\_analyzed"

},

"request\_method": {

"type": "string",

"index": "not\_analyzed"

},

"uri": {

"type": "string",

"index": "not\_analyzed"

},

"query\_string": {

"type": "string",

"index": "not\_analyzed"

},

"request\_body": {

"type": "string",

"index": "not\_analyzed"

},

"upstream\_response": {

"type": "string",

"index": "not\_analyzed"

},

"upstream\_cache\_status": {

"type": "string",

"index": "not\_analyzed"

},

"upstream\_host": {

"type": "string",

"index": "not\_analyzed"

},

"upstream\_response\_time": {

"type": "string",

"index": "not\_analyzed"

},

"response\_time": {

"type": "string",

"index": "not\_analyzed"

},

"upstream\_content\_type": {

"type": "string",

"index": "not\_analyzed"

}

}

}

}

}

建立启动项文件(未实际验证)

cd ~/logstash-6.0.0/

#目录

mkdir -p sbin

vim sbin/nginx\_lanmps.sh

* 1
* 2
* 3
* 4

nginx\_lanmps.sh 内容如下

#! /bin/sh

# Startup script for the logstash

# chkconfig: - 85 15

# description: logstash

# processname: logstash

#PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

NAME=logstash

NAME\_CONF=nginx-lanmps

DESC="logstash-$NAME\_CONF daemon"

IN\_DIR="/home/elasticsearch"

DAEMON\_PATH=$IN\_DIR/logstash-5.6.1

DAEMON=$DAEMON\_PATH/bin/$NAME

CONF=$DAEMON\_PATH/etc/$NAME\_CONF.conf

CONF\_DATA=$DAEMON\_PATH/data/$NAME\_CONF

PID\_FILE=$DAEMON\_PATH/data/$NAME\_CONF.pid

SCRIPT\_NAME=$DAEMON\_PATH/sbin/$NAME\_CONF

CONF\_LOG=$DAEMON\_PATH/logs/$NAME\_CONF.log

//

INDEX\_NAME=$NAME\_CONF

HOST\_URL=localhost:9200/$INDEX\_NAME

set -e

[ -x "$DAEMON" ] || exit 0

do\_start() {

cd $DAEMON\_PATH

mkdir -p $CONF\_DATA

nohup $DAEMON -f $CONF --path.data=$CONF\_DATA > $CONF\_LOG 2>&1 &

}

do\_stop() {

ps -ef |grep /$NAME\_CONF|awk '{print $2}'|xargs kill -9

}

do\_reload() {

ps -ef |grep /$NAME\_CONF|awk '{print $2}'|xargs kill -HUP

}

do\_delete(){

echo "DELETE "$NAME\_CONF

#curl -XDELETE 'http://'$HOST\_URL

echo "\n"

}

do\_create(){

echo "Auto Create "$NAME\_CONF

echo "\n"

}

case "$1" in

start)

echo -n "Starting $DESC: $NAME"

do\_start

echo "."

;;

stop)

echo -n "Stopping $DESC: $NAME"

do\_stop

echo "."

;;

reload|graceful)

echo -n "Reloading $DESC configuration..."

do\_reload

echo "."

;;

restart)

echo -n "Restarting $DESC: $NAME"

do\_stop

do\_start

echo "."

;;

status)

if [ -f $PID\_FILE ]; then

echo "$NAME is runing!"

else

echo "$NAME is stop!"

fi

;;

create)

do\_delete

do\_create

;;

test)

$DAEMON -f $CONF -t

;;

\*)

echo "Usage: $SCRIPT\_NAME {start|stop|reload|restart|status|test|create} " >&2

exit 3

;;

esac

exit 0

启动方式

~/logstash-5.6.1/sbin/nginx\_lanmps.sh start 启动

~/logstash-5.6.1/sbin/nginx\_lanmps.sh stop 关闭

其他请自行摸索

* 1
* 2
* 3

**配置案例 ELKF （F是Filebeat） 方式 推荐**

如果直接用Logstash 去读取站点日志，Logstash太占用CPU了。

**Filebeat安装**

下载   
<https://www.elastic.co/downloads/beats/filebeat>

wget https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-6.0.0-linux-x86\_64.tar.gz

* 1

解压缩

tar -zxvf filebeat-6.0.0-linux-x86\_64.tar.gz

mv filebeat-6.0.0-linux-x86\_64 /www/filebeat/

* 1
* 2

**配置Filebeat**

环境说明：   
1）elasticsearch和logstash 在不同/相同的服务器上,只发送数据给logstash/elasticsearch   
2）监控nginx日志   
3）监控站点日志

**配置**

编辑filebeat.yml

cd /www/filebeat/filebeat-6.0.0-linux-x86\_64

vim filebeat.yml

* 1
* 2

修改为

filebeat.prospectors:

- input\_type: log

paths:

- /www/wwwLog/www.foxwho.com/\*.log

input\_type: log

document\_type: nginx-www.foxwho.com

multiline.pattern: '^\['

multiline.negate: true

multiline.match: after

- input\_type: log

paths:

- /www/wwwroot/www.foxwho.com/runtime/log/\*/[0-9]\*[\_\w]?\*.log

input\_type: log

document\_type: web-www.foxwho.com

multiline.pattern: '^\['

multiline.negate: true

multiline.match: after

#以下输出给elasticsearch

#output.elasticsearch:

# hosts: ["localhost:9200"]

# index: "filebeat-www.babymarkt.cn"

# template.name: "filebeat"

# template.path: "filebeat.template.json"

# template.overwrite: false

#以下输出给logstash 进行处理

output.logstash:

hosts: ["10.1.5.65:5044"]

...其他部分没有改动，不需要修改

* 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

**filebeat配置说明**

1. paths：指定要监控的日志，目前按照Go语言的glob函数处理。没有对配置目录做递归处理，比如配置的如果是：

/var/log/\* /\*.log

* 1

则只会去/var/log目录的所有子目录中寻找以”.log”结尾的文件，而不会寻找/var/log目录下以”.log”结尾的文件。   
2. input\_type：指定文件的输入类型log(默认)或者stdin。   
3. document\_type：设定Elasticsearch输出时的document的type字段，也可以用来给日志进行分类。

把 elasticsearch和其下的所有都注释掉（这里Filebeat是新安装的，只注释这2处即可）

#output.elasticsearch:

# hosts: ["localhost:9200"]

* 1
* 2

开启 logstash(删除这两行前的#号)，并把localhost改为logstash服务器地址

output.logstash:

hosts: ["10.1.5.65:5044"]

* 1
* 2

**logstash 配置**

如果开启logstash了，那么Logstash配置中要设置监听端口 5044：   
这个是默认文件位置，如果不存在请自行查找   
建立beats-input.conf配置文件

vim ~/logstash-6.0.0/etc/beats-input-foxwho.com.conf

* 1

增加端口

input {

beats {

port => 5044

}

}

filter {

grok {

match=>[

"message",

"%{IPORHOST:client\_ip} %{USER:ident} %{USER:auth} \[%{HTTPDATE:timestamp}\] \"(?:%{WORD:verb} %{NOTSPACE:request}(?: HTTP/%{NUMBER:http\_version})?|-)\" %{NUMBER:response} (?:%{NUMBER:bytes}|-) (%{QS:referrer}|-) (%{QS:agent}|-) \"(%{WORD:x\_forword}|-)\" (%{HOSTNAME:domain}|-) (%{WORD:request\_method}|-) (%{QS:uri}|-) (%{QS:query\_string}) (%{NUMBER:upstream\_response}|-) (%{WORD:upstream\_cache\_status}|-) (%{URIHOST:upstream\_host}|-) (%{USERNAME:upstream\_response\_time}) > (%{USERNAME:response\_time}) %{QS:upstream\_content\_type} (?:%{QS:request\_body}|-)"

]

}

mutate {

gsub => [

# 将filed\_name\_2字段中所有"\","?","#","-"转换为"."

"agent", "\"", "",

"upstream\_content\_type", "\"", "",

"query\_string", "\"", "",

"uri", "\"", "",

"request\_body", "\"", "",

"referrer", "\"", ""

]

}

#匹配模式 message是每段读进来的日志，IP、HTTPDATE、WORD、NOTSPACE、NUMBER都是patterns/grok-patterns中定义好的正则格式名称，对照上面的日志进行编写,冒号，(?:%{USER:ident}|-)这种形式是条件判断，相当于程序里面的二目运算。如果有双引号""或者[]号，需要在前面加\进行转义。

#kv {

# source => "request"

# field\_split => "&?"

# value\_split => "="

#}

#再单独将取得的URL、request字段取出来进行key-value值匹配，需要kv插件。提供字段分隔符"&?"，值键分隔符"="，则会自动将字段和值采集出来。

#urldecode {

# all\_fields => true

#}

#把所有字段进行urldecode（显示中文）

#定义时间戳的格式

date {

match => [ "timestamp", "yyyy-MM-dd-HH:mm:ss" ]

locale => "cn"

}

}

output {

elasticsearch {

hosts => "localhost:9200"

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

template => "/home/elasticsearch/logstash-6.0.0/template/nginx\_log.json"

template\_name => "nginx\_log-%{type}"

template\_overwrite => true

#安装了 x-pack后会用到 用户名及密码

#user => elastic

#password => changeme

}

stdout { codec => rubydebug }

}

* 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
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54

**启动logstash 该配置**

/home/elasticsearch/logstash-6.0.0/bin/logstash -f /home/elasticsearch/logstash-6.0.0/etc/beats-input-foxwho.com.conf

* 1

启动成功后，那么就可以进行filebeat启动了

**filebeat启动**

**filebeat测试**

cd /www/filebeat/filebeat-6.0.0-linux-x86\_64

./filebeat -e -c filebeat.yml -d "Publish"

* 1
* 2

如果能看到一堆东西输出，表示正在向elasticsearch或logstash发送日志。   
如果是elasticsearch可以浏览：<http://localhost:9200/_search?pretty> 如果有新内容返回，表示ok   
测试正常后，Ctrl+C结束

logstash 控制台中看看是否有接收到日志

**filebeat启动**

nohup ./filebeat -e -c filebeat.yml &

* 1

上面会转入后台运行

**filebeat停止**

查找进程 ID

ps -ef |grep filebeat

* 1

KILL他

kill -9 id

* 1

至此全部完成