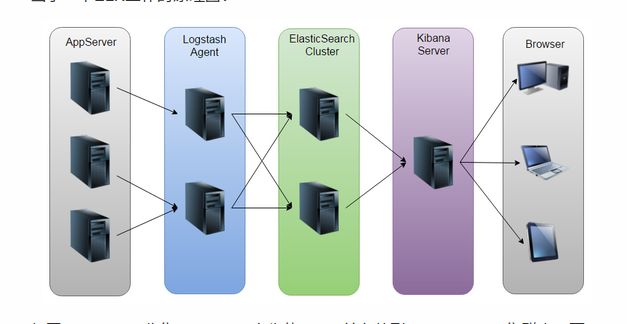
开源实时日志分析平台ELK：ELK由ElasticSearch, Logstash, Kiabana三个开源工具组成。

* Elasticsearch是个开源分布式搜索引擎，它的特点有：分布式，零配置，自动发现，索引自动分片，索引副本机制，restful风格接口，多数据源，自动搜索负载等。
* Logstash是一个完全开源的工具，他可以对你的日志进行收集、过滤，并将其存储供以后使用（如，搜索）。
* Kibana 也是一个开源和免费的工具，它Kibana可以为 Logstash 和 ElasticSearch 提供的日志分析友好的 Web 界面，可以帮助您汇总、分析和搜索重要数据日志。

ELK工作原理如下：

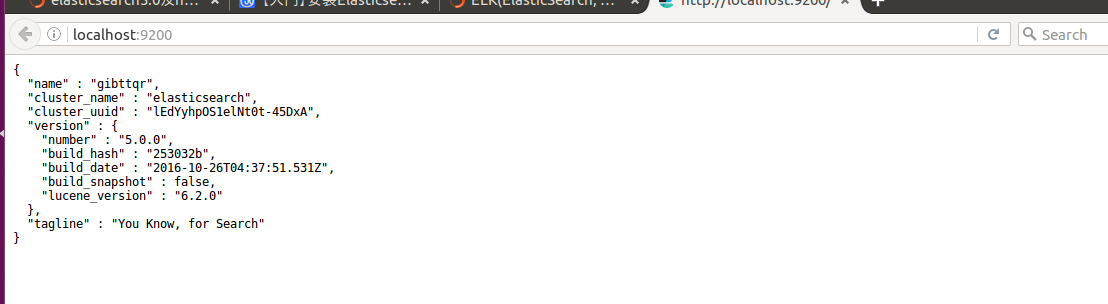


# ElasticSearch配置：

ELK下载： [***https://www.elastic.co/downloads/***](https://www.elastic.co/downloads/)

下载版本为5.0.0

下载完成后，解压缩，默认配置，可通过./bin/elasticsearch来启动ES。然后打开页面localhost:9200，可以看到以下内容：



接下来可以选择安装head插件，它是一个用浏览器跟ES集群交互的插件，可以查看集群状态、集群的doc内容、执行搜索和普通的Rest请求等。在5.0版本中不支持直接安装head插件，需要安装一个服务。

## 1. 环境安装：

a. 安装git：

sudo apt install git

b. 安装node

由于head插件本质上还是一个nodejs的工程，因此需要安装node，使用npm来安装依赖的包。（npm可以理解为maven）去官网下载nodejs，https://nodejs.org/en/download/

下载下来的jar包是xz格式的，一般的linux可能不识别，还需要安装xz：

sudo apt install xz

. 解压nodejs的安装包：

xz -d node\*.tar.xz

tar -xvf node\*.tar

. 解压完node的安装文件后，需要配置下环境变量,编辑/etc/profile，添加

# set node environment

export NODE\_HOME=/usr/elk/node-v6.9.1-linux-x64

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

要立即执行以下

source /etc/profile

重启后，全局生效

. 可以测试一下node是否生效：

[root@localnode1 node-v6.9.1-linux-x64]# echo $NODE\_HOME

/usr/elk/node-v6.9.1-linux-x64

[root@localnode1 node-v6.9.1-linux-x64]# node -v

v6.9.1

[root@localnode1 node-v6.9.1-linux-x64]# npm -v

3.10.8

c. 安装grunt

grunt是一个很方便的构建工具，可以进行打包压缩、测试、执行等等的工作，5.0里的head插件就是通过grunt启动的。因此需要安装一下grunt：

sudo npm -g install grunt-cli

安装完成后检查一下：

[root@localnode1 elasticsearch-head]# grunt -version

## 2.Head插件安装

a. 插件安装

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

cd elasticsearch-head

npm install

b. 在elasticsearch-head目录下node\_modules/grunt下如果没有grunt二进制程序，需要执行：

cd elasticsearch-head

npm install grunt --save

c. 修改配置

修改elasticsearch-head下Gruntfile.js文件，默认监听在127.0.0.1下9200端口



修改连接地址：

目录：elasticsearch-head/\_site/app.js

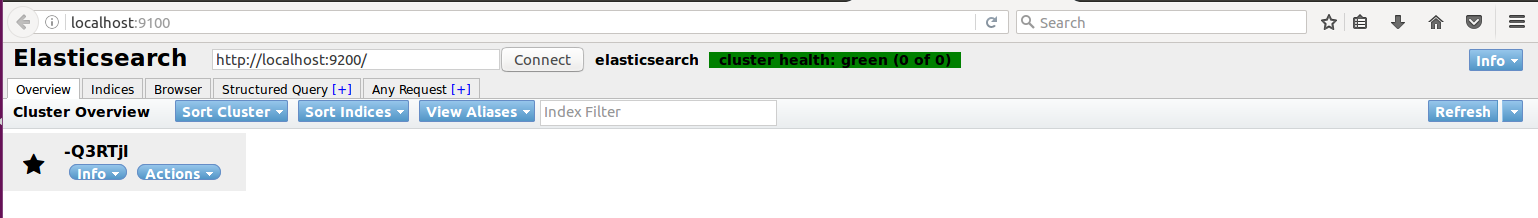
修改head的连接地址:

this.base\_uri = this.config.base\_uri || this.prefs.get("app-base\_uri") || "http://localhost:9200";

d. 启动服务：

/usr/local/elasticsearch-5.0.0/elasticsearch-head/node\_modules/grunt/bin/grunt server

启动成功后，在浏览器输入localhost:9100，可看到如下内容：



# Logstash配置

## 配置Logstash：

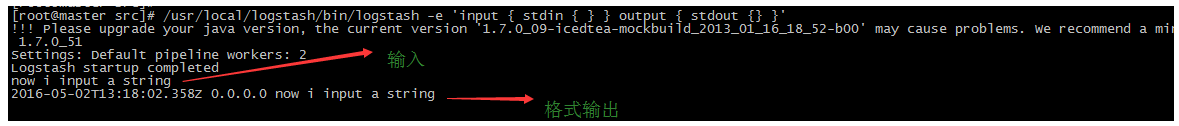
tar -zxvf logstash-5.0.0.tar.gz

cd logstash-5.0.0

测试logstash

(1) 屏幕输入输出方式测试

/usr/local/logstash/bin/logstash -e 'input { stdin { } } output { stdout {} }'



我们可以看到，我们输入什么内容logstash按照某种格式输出，其中-e参数参数允许Logstash直接通过命令行接受设置。这点尤其快速的帮助我们反复的测试配置是否正确而不用写配置文件。使用CTRL-C命令可以退出之前运行的Logstash。

## 编写配置文件，放在config目录下：

配置logstash

创建配置文件目录：

gedit /usr/local/logstash/config/hello\_search.conf

输入下面：

input {

  stdin {

    type => "human"

  }

}

output {

  stdout {

    codec => rubydebug

  }

  elasticsearch {

        hosts => "192.168.100.10:9200"

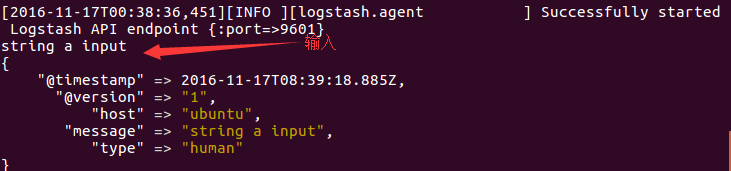
  }

}

启动：

/usr/local/logstash-5.0.0/bin/logstash -f /usr/local/logstash-5.0.0/config/hello\_search.conf

看到如下页面，代表配置成功



## **测试logstash日志是否传输到了elasticsearch**

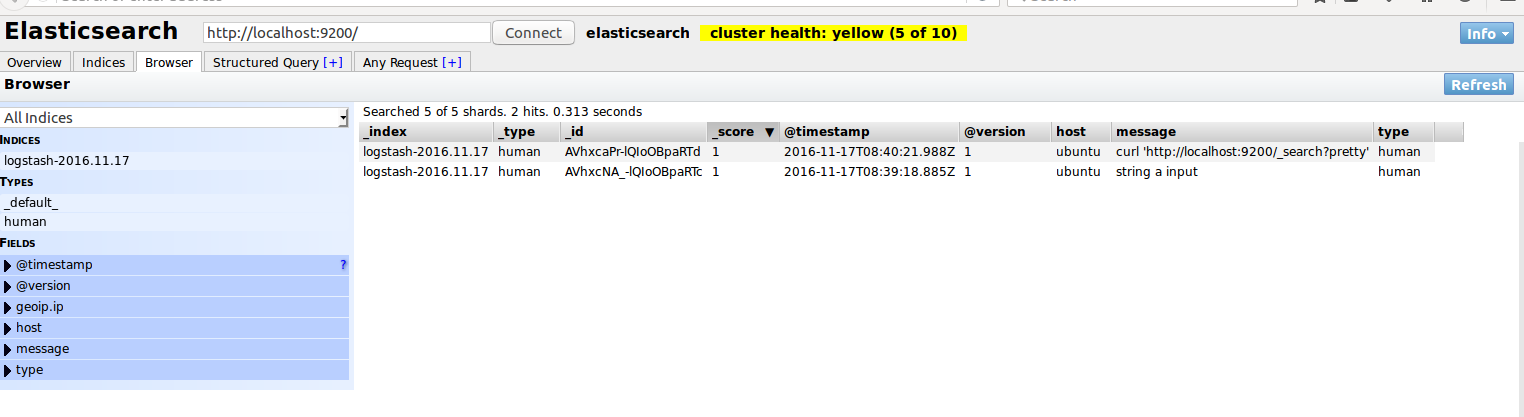
打开http://localhost:9100/，可看到以下页面：



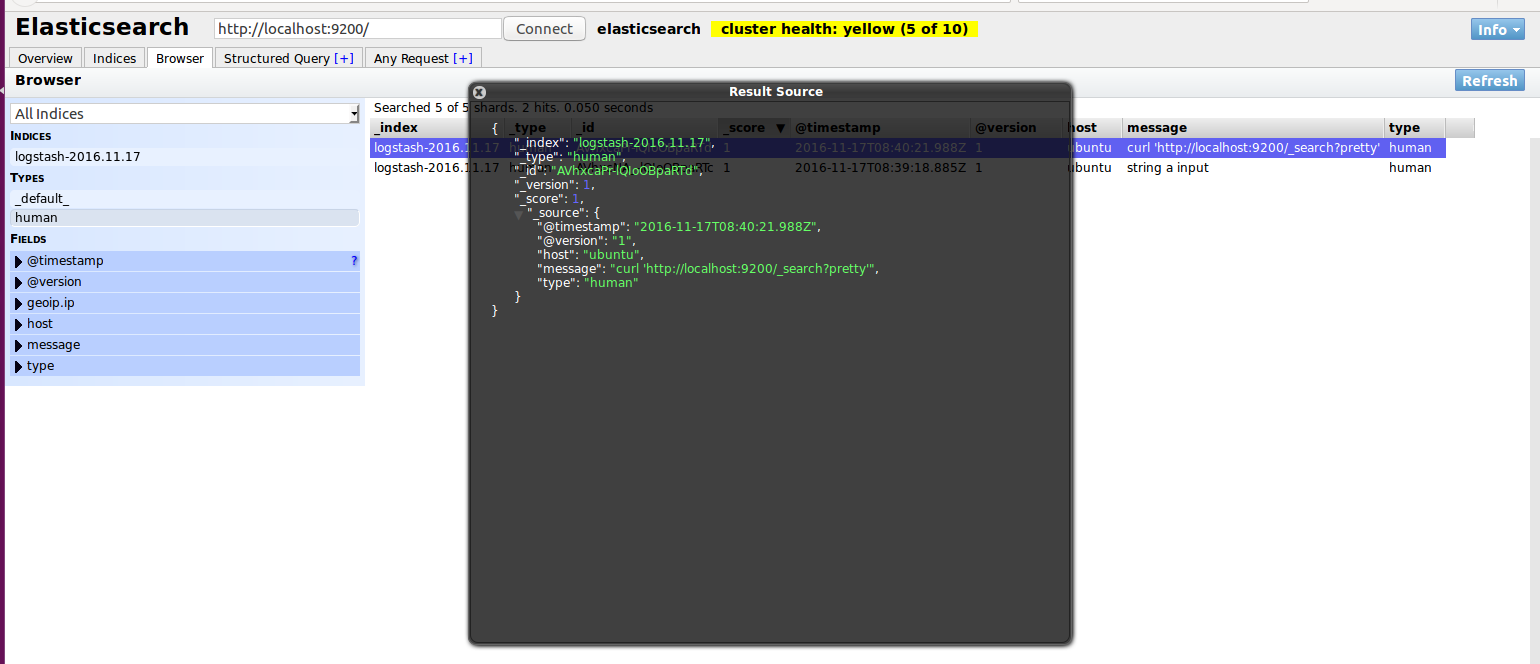
表明已经接收到了日志

至此，你已经成功利用Elasticsearch和Logstash来收集日志数据了。

切换到browser标签：



单击一个log,可以看到其详细信息



# 配置Kibana

tar -zxvf kibana-5.0.0-linux-x86.tar.gz

cp -r kibana-5.0.0-linux-x86.tar.gz /usr/local/kibana

cd /usr/local/kibana

gedit config/kibana.yml

修改以下几项：

server.port: 5601

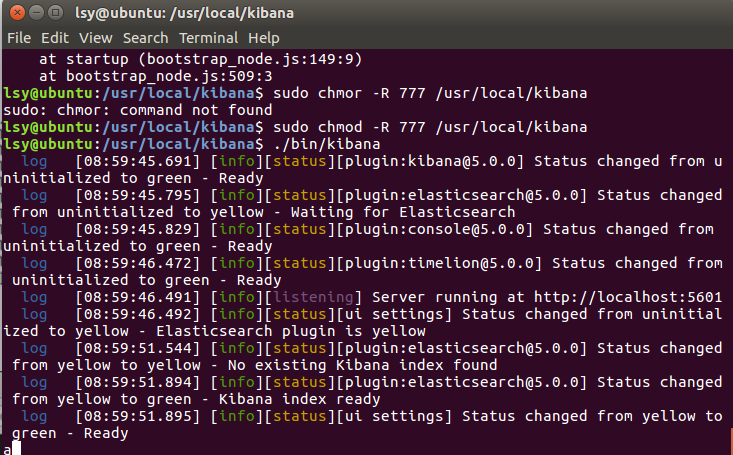
server.host: “localhost”

elasticsearch.url: http://localhost:9200

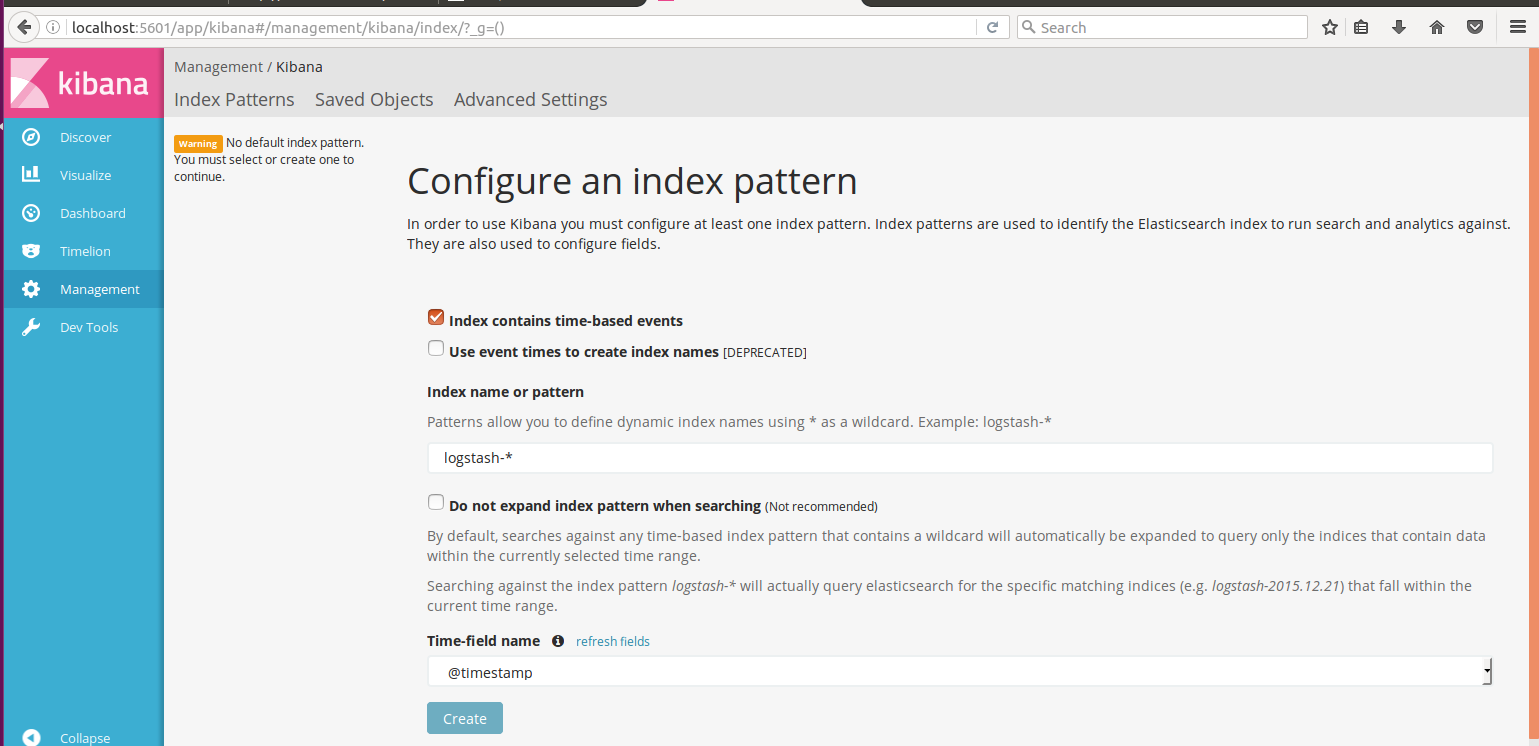
kibana.index: “.kibana

启动kibana：

./bin/kibana



用浏览器打开地址：localhost:5601



登录后，首先，配置一个索引，默认，Kibana的数据被指向Elasticsearch，使用默认的logstash-\*的索引名称，并且是基于时间的，点击“Create”即可。

点击“Discover”，可以搜索和浏览Elasticsearch中的数据，默认搜索的是最近15分钟的数据。可以自定义选择时间。



至此，ELK平台全部部署完成。

# 使用LOG4J与ELK进行协作

修改logstash启动时的指定配置文件，内容如下：

*input {*

*tcp {*

*host => "0.0.0.0"*

*port => "4560"*

*mode => "server"*

*type => "microwiki"*

*add\_field => {*

*"name" => "Routh"*

*}*

*}*

*stdin {}*

*}*

*filter {*

*}*

*output {*

*stdout {*

*codec => rubydebug*

*}*

*elasticsearch {*

*hosts => ["localhost:9200"]*

*action => "index"*

*codec => rubydebug*

*index => "microwiki-%{+YYYY.MM.dd}"*

*template\_name => "microwiki"*

*}*

*}*

在eclipse中新建maven项目，pom.xml添加依赖：

<dependency>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j-api</artifactId>

<version>2.6.2</version>

</dependency>

<dependency>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j-core</artifactId>

<version>2.6.2</version>

</dependency>

配置log4j2.xml如下：

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<Configuration status=*"WARN"* monitorInterval=*"60"*>

<Properties>

<Property name=*"PATTERN"*>%d %-5p [%t] %C{1} (%F:%L) - %m%n</Property>

</Properties>

<Appenders>

<Socket name=*"Logstash"* host=*"127.0.0.1"* port=*"4560"* protocol=*"TCP"*>

<PatternLayout pattern=*"${PATTERN}"* />

</Socket>

</Appenders>

<Loggers>

<Root level=*"error"*>

<AppenderRef ref=*"Logstash"*/>

</Root>

<Logger name=*"Error"* level=*"error"* additivity=*"false"*>

<AppenderRef ref=*"Logstash"*/>

</Logger>

<Logger name=*"Request"* level=*"info"* additivity=*"false"*>

<AppenderRef ref=*"Logstash"*/>

</Logger>

</Loggers>

</Configuration>

编写java文件如下：

**package** com.giot.elk\_demo;

**import** org.apache.logging.log4j.LogManager;

**import** org.apache.logging.log4j.Logger;

**public** **class** App

{

**final** **static** Logger ***logger*** = LogManager.*getLogger*(App.**class**);

**public** **static** **void** main(String[] args) **throws** InterruptedException {

**for** (**int** i = 0; i < 10; i++) {

***logger***.error("Info log [" + i + "].");

Thread.*sleep*(500);

}

}

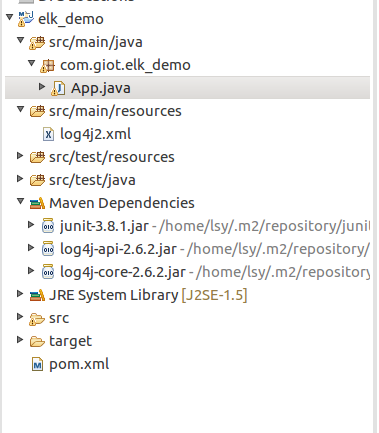
**private** **void** divide(){

**int** i = 10 /0;

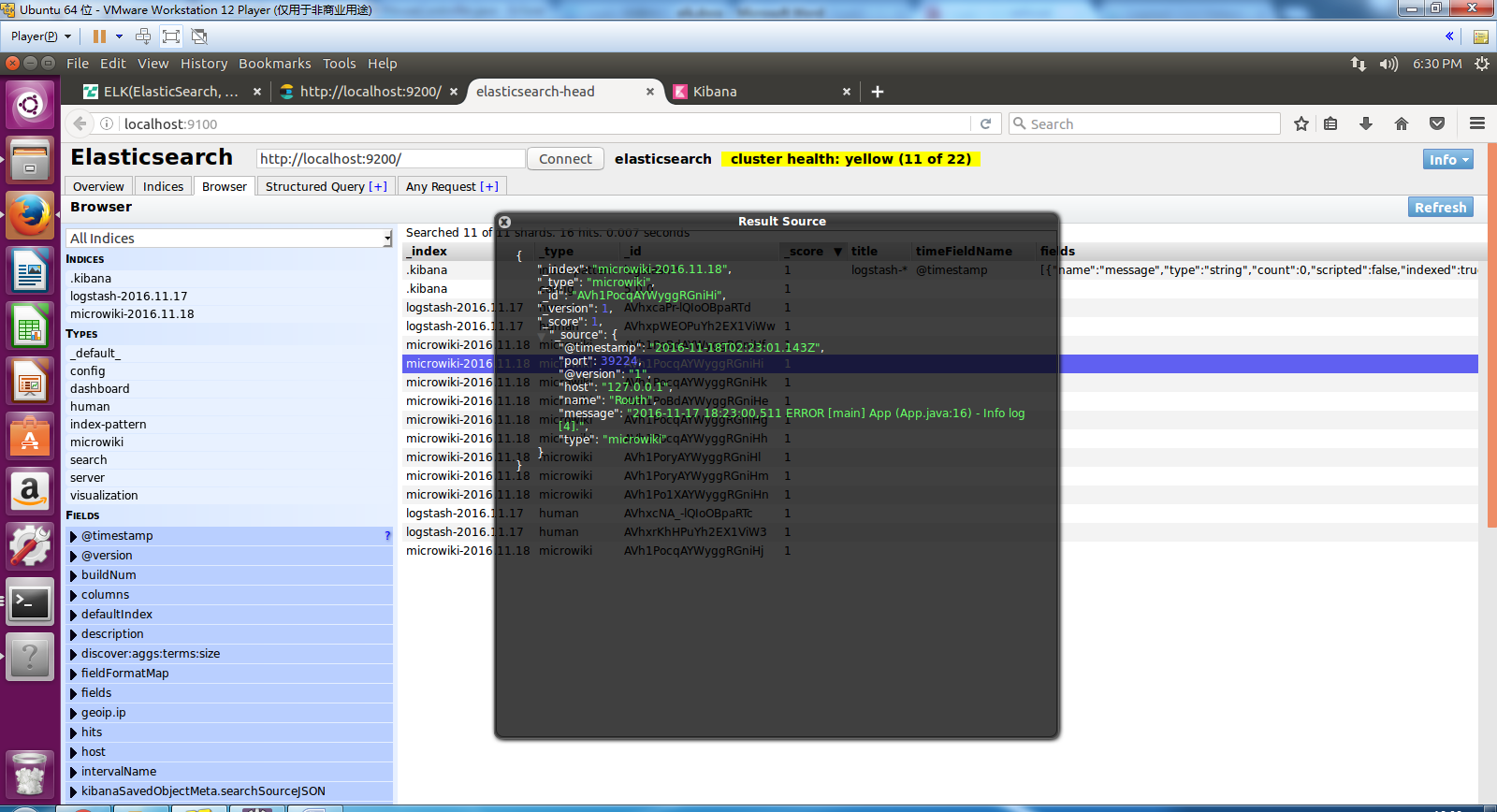
}

}

项目目录结构如下：



运行app.java，打开localhost:9100，可以看到程序中输出的log信息：



打开Kibana(localhost:5601)，可看到新增的log信息

根据我们之前设定的index，在kibana中添加新的索引项：

