# 开源实时日志分析ELK平台部署

**参考文档：**

|  |
| --- |
| <http://bbotte.com/logs-service/use-elk-processing-logs-basic-installation-and-config/>  <http://www.tuicool.com/articles/QFvARfr>  <http://www.open-open.com/lib/view/open1473661753307.html>  http://shaonian.blog.51cto.com/2975261/1795389 |

**启动：**

|  |
| --- |
| /usr/local/elasticsearch-2.4.1/bin/elasticsearch >nohup &  /usr/local/logstash-2.4.0/bin/logstash -f logstash-indexer.conf > nohup &  /usr/local/kibana-4.6.1-linux-x86\_64/bin/kibana > nohup & |

日志主要包括系统日志、应用程序日志和安全日志。系统运维和开发人员可以通过日志了解服务器软硬件信息、检查配置过程中的错误及错误发生的原因。经常分析日志可以了解服务器的负荷，性能安全性，从而及时采取措施纠正错误。

**1 ）安装 Logstash 依赖包 JDK**

|  |
| --- |
| # mkdir /usr/local/java  # tar -zxf jdk-8u45-linux-x64.tar.gz -C /usr/local/java/ |

**设置 JDK 的环境变量，如下：**

|  |
| --- |
| # tail -3 ~/.bash\_profile  export JAVA\_HOME=/usr/local/java/jdk1.8.0\_45  export PATH=$PATH:$JAVA\_HOME/bin  exportCLASSPATH=.:$JAVA\_HOME/lib/tools.jar:$JAVA\_HOME/lib/dt.jar:$CLASSPATH |

**另一种设置方法：**

|  |
| --- |
| # vim ~/.bashrc  export JAVA\_HOME=/usr/java/jdk1.7.0\_79  export JRE\_HOME=${JAVA\_HOME}/jre  export CLASSPATH=.:${JAVA\_HOME}/lib:${JRE\_HOME}/lib  export PATH=${JAVA\_HOME}/bin:$PATH  # . ~/.bashrc  # java -version |

**java -version**

|  |
| --- |
| # java -version  java version "1.8.0\_45"  Java(TM) SE Runtime Environment (build 1.8.0\_45-b14)  Java HotSpot(TM) 64-Bit Server VM (build 25.45-b02,mixed mode) |

**2 ）安装 Logstash**

|  |
| --- |
| # https://download.elastic.co/logstash/logstash/logstash-1.5.2.tar.gz  # tar zxf logstash-1.5.2.tar.gz -C /usr/local/ |

**安装完成后运行如下命令：**

|  |
| --- |
| # /usr/local/logstash-1.5.2/bin/logstash -e 'input { stdin { } } output { stdout {} }'  Logstash startup completed  Hello World!  2015-07-15T03:28:56.938Z noc.vfast.com Hello World! |

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。

使用-e参数在命令行中指定配置是很常用的方式，不过如果需要配置更多设置则需要很长的内容。这种情况，我们首先创建一个简单的配置文件，并且指定logstash使用这个配置文件。 例如：在 logstash 安装目录下创建一个“基本配置”测试文件 logstash-test.conf， 文件内容如下：

|  |
| --- |
| # cat logstash-simple.conf  input { stdin { } }  output {  stdout { codec=> rubydebug }  } |

Logstash 使用 input 和 output 定义收集日志时的输入和输出的相关配置，本例中 input 定义了一个叫 "stdin" 的 input ， output 定义一个叫 "stdout" 的 output 。无论我们输入什么字符， Logstash 都会按照某种格式来返回我们输入的字符，其中 output 被定义为 "stdout" 并使用了 codec 参数来指定 logstash 输出格式。

使用logstash的-f参数来读取配置文件，执行如下开始进行测试：

|  |
| --- |
| # echo "`date` hello World"  Thu Jul 16 04:06:48 CST 2015 hello World  # /usr/local/logstash-1.5.2/bin/logstash agent -f logstash-simple.conf  Logstash startup completed  Tue Jul 14 18:07:07 EDT 2015 hello World #该行是执行echo “`date`hello World” 后输出的结果，直接粘贴到该位置  {  "message" => "Tue Jul 14 18:07:07 EDT 2015 helloWorld",  "@version" => "1",  "@timestamp" => "2015-07-14T22:07:28.284Z",  "host" => "noc.vfast.com"  } |

**3 ）安装 Elasticsearch**

下载 Elasticsearch 后，解压到对应的目录就完成 Elasticsearch 的安装。

|  |
| --- |
| # tar -zxf elasticsearch-1.6.0.tar.gz -C /usr/local/ |

**配置文件：**

在/usr/local/elasticsearch-1.6.0/config/elasticsearch.yml中，以下几个属性是建议修改的：特别是标红的字体设置项

|  |
| --- |
| cluster.name: ElasticSearch  node.name: node-1  path.data: /home/hadoop/install/deploy/elasticsearch/data  path.logs: /home/hadoop/install/deploy/elasticsearch/logs  bootstrap.mlockall: true  network.host: 192.168.253.13  http.port: 9200 |

由于防止了root启动，需添加普通用户启动：

|  |
| --- |
| groupadd ela  useradd -g ela ela  chown -R ela:ela /usr/local/elasticsearch-1.6.0 |

此时在普通用户下也要配置java环境：

|  |
| --- |
| # tail -3 ~/.bash\_profile  export JAVA\_HOME=/usr/local/java/jdk1.8.0\_45  export PATH=$PATH:$JAVA\_HOME/bin  exportCLASSPATH=.:$JAVA\_HOME/lib/tools.jar:$JAVA\_HOME/lib/dt.jar:$CLASSPATH |

启动 Elasticsearch

|  |
| --- |
| su - ela  # /usr/local/elasticsearch-1.6.0/bin/elasticsearch |

如果使用远程连接的 Linux 的方式并想后台运行 elasticsearch 执行如下命令：

|  |
| --- |
| # nohup /usr/local/elasticsearch-1.6.0/bin/elasticsearch >nohup & |

确认 elasticsearch 的 9200 端口已监听，说明 elasticsearch 已成功运行

|  |
| --- |
| # netstat -anp |grep :9200  tcp 0 0 :::9200 :::\* LISTEN 3362/java |

接下来我们在 logstash 安装目录下创建一个用于测试 logstash 使用 elasticsearch 作为 logstash 的后端的测试文件 logstash-es-simple.conf，该文件中定义了stdout和elasticsearch作为output，这样的“多重输出”即保证输出结果显示到屏幕上，同时也输出到elastisearch中。

|  |
| --- |
| # cat logstash-es-simple.conf  input { stdin { } }  output {  elasticsearch {hosts=> "localhost" }  stdout { codec=> rubydebug }  } |

注意：上面配置文件output中，elasticsearch在2.\*新版本中host更改为了hosts，下面配置也一样，需要留意，推荐大家使用新版本的ELK

|  |
| --- |
| output { elasticsearch { hosts => localhost } } |

**否则报错信息如下：**

|  |
| --- |
| Pipeline aborted due to error {:exception=>#<LogStash::ConfigurationError: The setting `host` in plugin `elasticsearch` is obsolete and is no longer available. Please use the 'hosts' setting instead. You can specify multiple entries separated by comma in 'host:port' format. |

**执行如下命令**

|  |
| --- |
| # /usr/local/logstash-1.5.2/bin/logstash agent -f logstash-es-simple.conf  Logstash startup completed  hello logstash  {  "message" => "hello logstash",  "@version" => "1",  "@timestamp" => "2015-07-15T18:12:00.450Z",  "host" => "noc.vfast.com"  } |

我们可以使用 curl 命令发送请求来查看 ES 是否接收到了数据：

|  |
| --- |
| # curl 'http://localhost:9200/\_search?pretty'  返回结果  {  "took": 58,  "timed\_out" : false,  "\_shards" : {  "total" : 5,  "successful" : 5,  "failed" : 0  },  "hits": {  "total" : 1,  "max\_score" : 1.0,  "hits" : [ {  "\_index" : "logstash-2015.07.15",  "\_type" : "logs",  "\_id" : "AU6TWiixxDXYhySMyTkP",  "\_score" : 1.0,  "\_source":{"message":"hellologstash","@version":"1","@timestamp":"2015-07-15T20:13:55.199Z","host":"noc.vfast.com"}  } ]  }  } |

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

4 ）安装 elasticsearch 插件

Elasticsearch-kopf 插件可以查询 Elasticsearch 中的数据，安装 elasticsearch-kopf，只要在你安装 Elasticsearch 的目录中执行以下命令即可：

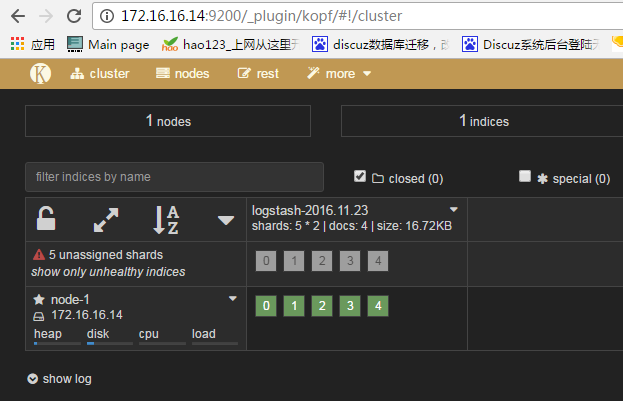
|  |
| --- |
| # cd /opt/elasticsearch  # ./bin/plugin install lmenezes/elasticsearch-kopf  -> Installing lmenezes/elasticsearch-kopf...  Trying https://github.com/lmenezes/elasticsearch-kopf/archive/master.zip...  Downloading .............................................DONE  Installed lmenezes/elasticsearch-kopf into /opt/elasticsearch/plugins/kopf |

可见插件的安装是下载zip包，再解压到plugins文件夹

下面测试这个kopf插件

|  |
| --- |
| # ps aux|egrep "elastic|logstash"|grep -v grep|awk '{print $2}'|xargs kill -9  # ./bin/elasticsearch &  # /opt/logstash/bin/logstash -e 'input { stdin { } } output { elasticsearch { host => localhost } stdout { } }'  hello world #输入测试日志  2016-02-23T07:12:25.770Z localhost.localdomain hello world    # curl 'http://localhost:9200/\_search?pretty' #会看到刚才输出一定格式log文件  {  "took" : 37,  "timed\_out" : false,  "\_shards" : {  "total" : 5,  "successful" : 5,  "failed" : 0  },  "hits" : {  "total" : 3,  "max\_score" : 1.0,  "hits" : [ {  "\_index" : "logstash-2016.02.23",  "\_type" : "logs",  "\_id" : "AVMM-M6sQ5RLDkTmkGOV",  "\_score" : 1.0,  "\_source":{"message":"hello world","@version":"1","@timestamp":"2016-02-23T07:12:25.770Z","host":"localhost.localdomain"}  } ]  }  } |

在浏览器访问 <http://172.16.16.14:9200/_plugin/kopf> 浏览保存在 Elasticsearch 中的数据，如下所示：



***http://172.16.16.14:9200/\_plugin/head***



另，还有好多很不错的插件，都可以安装上去：  
**es\_head:** 这个主要提供的是健康状态查询，当然标签页里也提供了简单的form给你提交API请求。es\_head现在可以直接通过

|  |
| --- |
| elasticsearch/bin/plugin install mobz/elasticsearch-head |

安装，然后浏览器里直接输入http://$eshost:9200/\_plugin/head/ 就可以看到cluster/node/index/shards的状态了  
**bigdesk:** 这个主要提供的是节点的实时状态监控，包括jvm的情况，linux的情况，elasticsearch的情况。排查性能问题的时候很有用，现在也可以过

|  |
| --- |
| elasticsearch/bin/plugin install lukas-vlcek/bigdesk |

直接安装了。然后浏览器里直接输入 http://$eshost:9200/\_plugin/bigdesk/ 就可以看到了。注意如果使用的 bulk\_index 的话，如果选择的刷新间隔太长，indexing per second数据是不准的

**Marvel:** 这个插件是监控elasticsearch的状态和性能。包括集群的状态、实时处理的日志条数等

如果不能在线安装elasticsearch插件，则  
bin/plugin install file:///path/to/plugin.zip  
**或者如下操作：**  
1.下载插件的zip包并解压，比如abc.zip  
2.建立/opt/elasticsearch/plugins/abc文件夹  
3.将解压后的abc文件夹下的文件copy到abc下  
4.运行es  
5.打开http://IP:9200/\_plugin/abc/

5 ）安装 Kibana

下载 kibana 后，解压到对应的目录就完成 kibana 的安装

|  |
| --- |
| # tar -zxf kibana-4.1.1-linux-x64.tar.gz -C /usr/local/ |

启动 kibana

|  |
| --- |
| # /usr/local/kibana-4.1.1-linux-x64/bin/kibana |

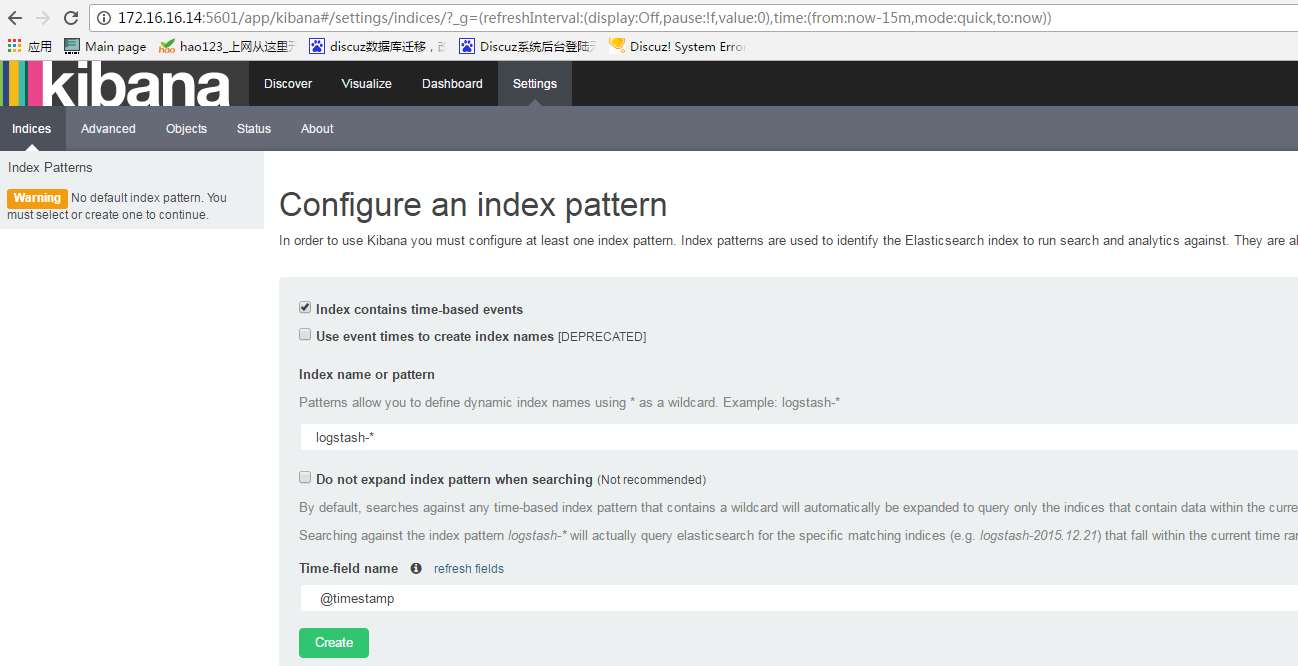
使用 [http://kibanaServerIP: 5601](http://kibanaServerIP: 5601 )访问 Kibana ，出现如下报错时：

|  |
| --- |
| [warning][elasticsearch] Unable to revive connection: http://localhost:9200/ |

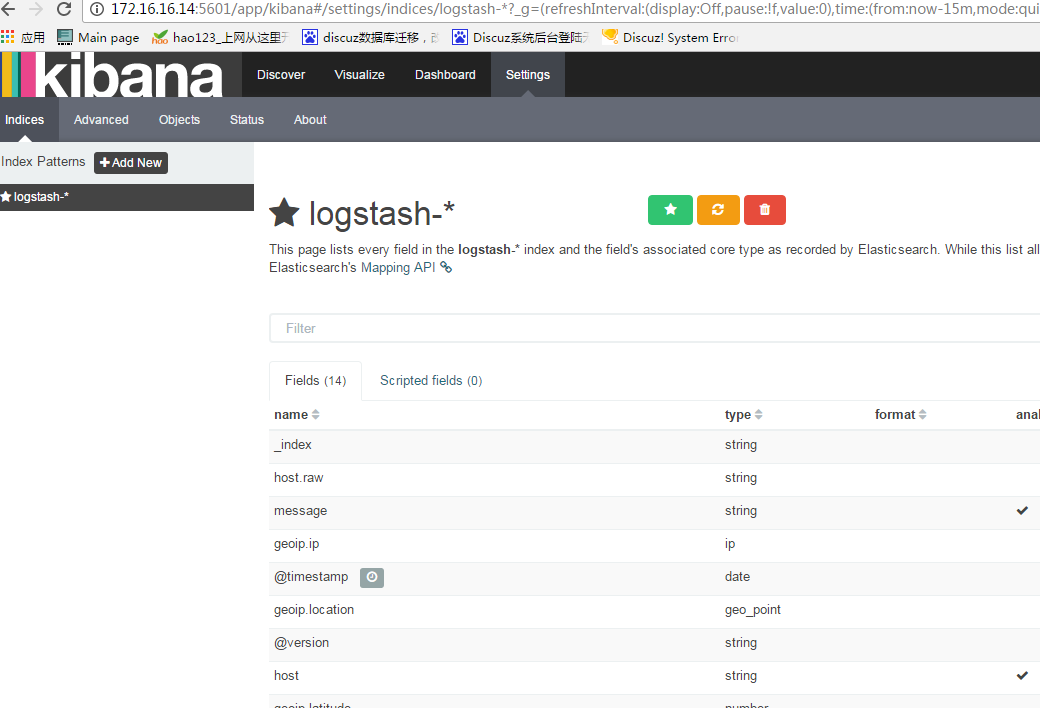
需更改如下配置：

|  |
| --- |
| vim /usr/local/kibana-4.6.1-linux-x86\_64/config/kibana.yml |

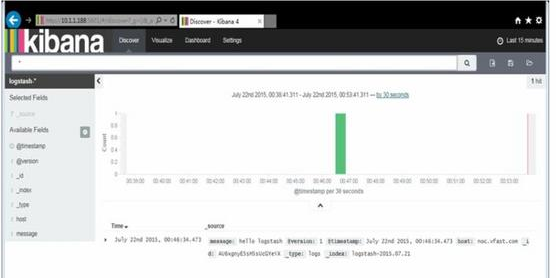
访问：<http://172.16.16.14:5601>登录后，首先，配置一个索引，默认， Kibana 的数据被指向 Elasticsearch ，使用默认的 logstash-\* 的索引名称，并且是基于时间的，点击“ Create ”即可。



看到如下界面说明索引创建完成。



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。



到此， ELK 平台安装部署完成。

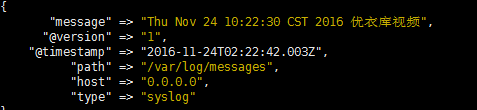
 6 ）配置 logstash 作为 Indexer

将 logstash 配置为索引器，并将 logstash 的日志数据存储到 Elasticsearch ，本范例主要是索引本地系统日志。

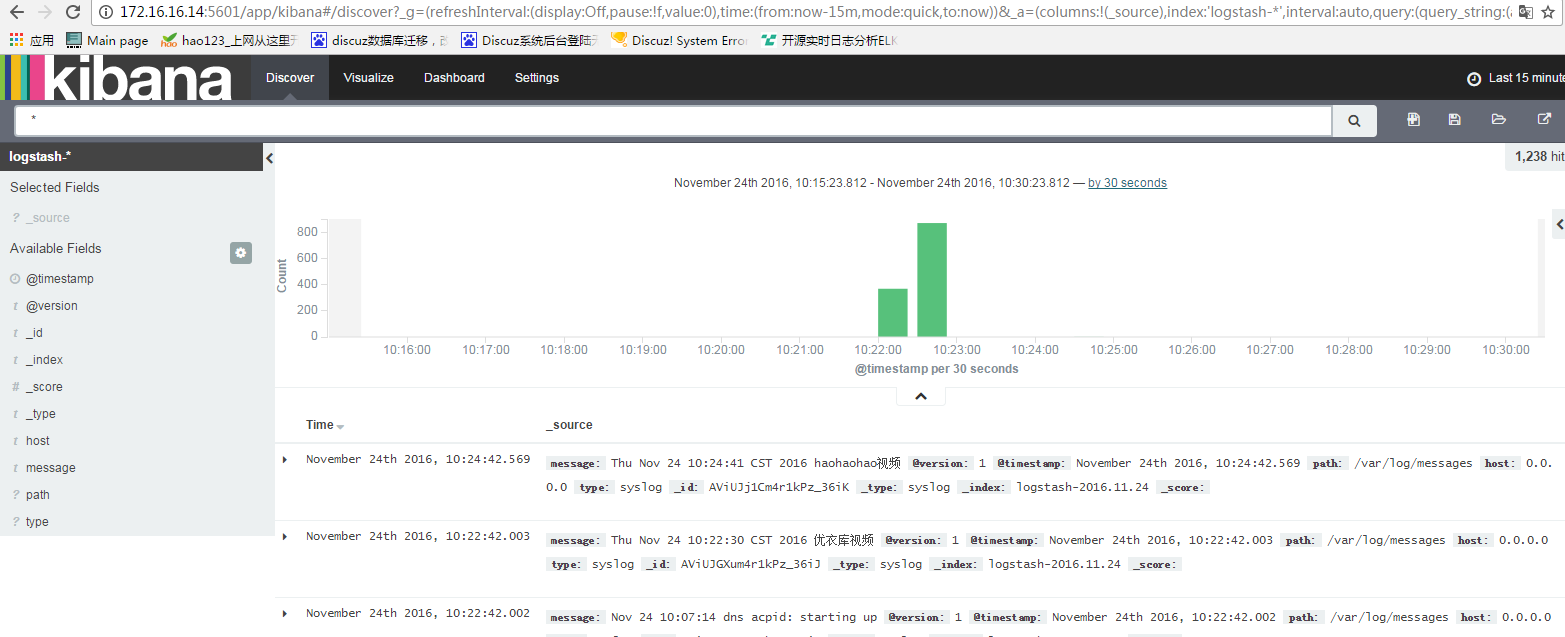
|  |
| --- |
| # cat /usr/local/logstash-1.5.2/logstash-indexer.conf  input {  file {  type =>"syslog"  path => ["/var/log/messages", "/var/log/syslog" ]  }  syslog {  type =>"syslog"  port =>"5544"  }  }  output {  stdout { codec=> rubydebug }  elasticsearch {hosts=> "172.16.16.14" }  }  # /usr/local/logstash-1.5.2/bin/logstash –f logstash-indexer.conf |

使用 echo 命令模拟写入日志，命令执行后看到如下图的信息

|  |
| --- |
| # echo "`date` 优衣库视频" >>/var/log/messages |



刷新 kibana ，发现最新的测试数据显示到浏览器中，如下图所示：

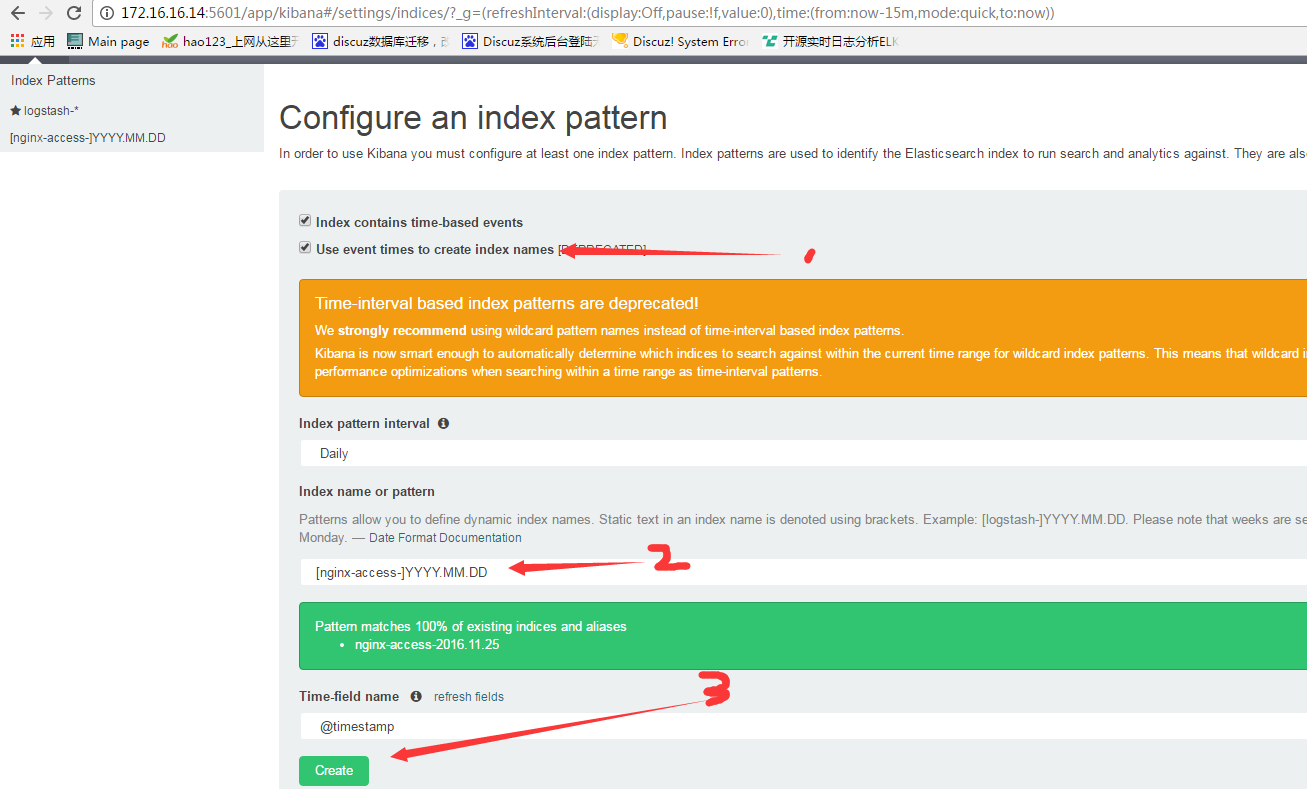


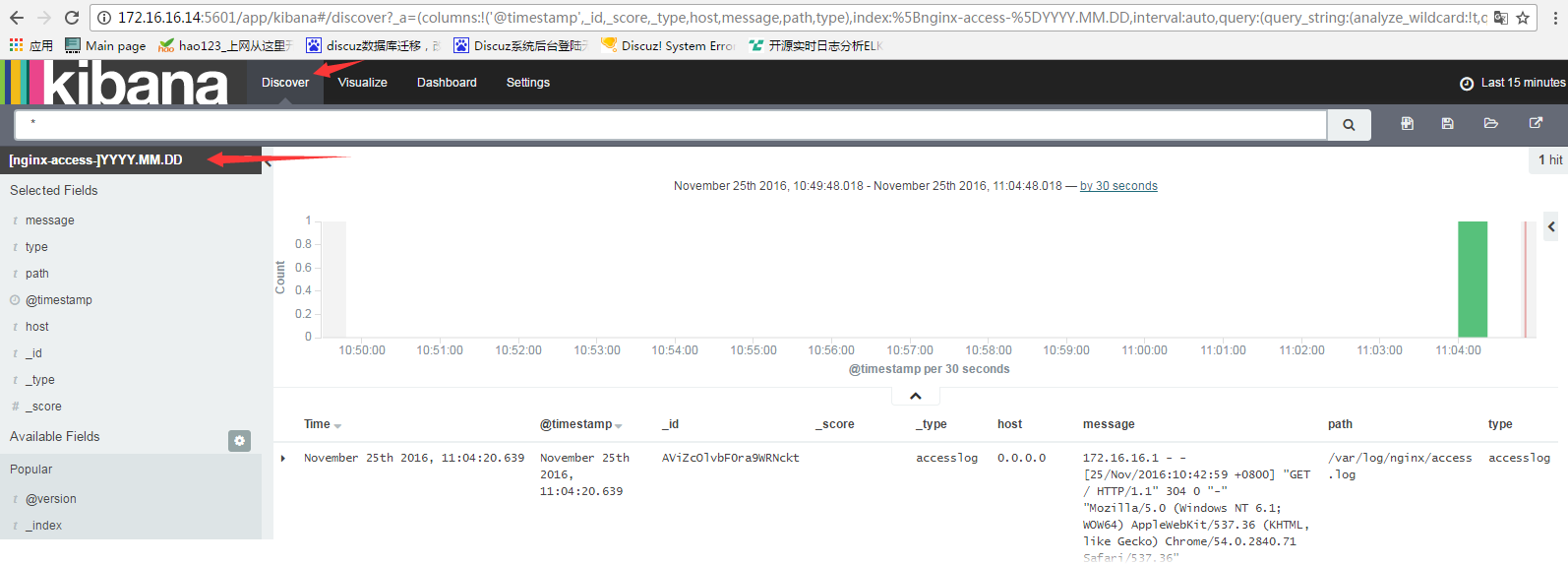
到此， ELK 平台部署和基本的测试已完成。

7 ) 收集nginx日志实例：

|  |
| --- |
| vim /usr/local/logstash-2.4.0/nginx.conf  input {  file {  type => "accesslog"  path => "/var/log/nginx/access.log"  start\_position => "beginning"  }  }  output {  if [type] == "accesslog" {  elasticsearch {  hosts => ["172.16.16.14:9200"]  index => "nginx-access-%{+YYYY.MM.dd}"  }  }  } |

添加展示的索引，就是在上面定义的  nginx-access-2016.07.03





8) 安全设置

kibana是直接访问的比较不安全，我们需要用nginx访问代理，并设置权限用户名和密码访问

在nginx里面配置

|  |
| --- |
| server  {  listen 80;  server\_name localhost;  auth\_basic "Restricted Access";  auth\_basic\_user\_file /usr/local/nginx/conf/htpasswd.users; #密码和用户  location / {  proxy\_pass http://localhost:5601; #代理kibana的5601之后就可以直接80访问了  proxy\_set\_header Host $host;  proxy\_set\_header X-Real-IP $remote\_addr;  proxy\_set\_header REMOTE-HOST $remote\_addr;  proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;    }  } |

创建密码和用户文件：htpasswd.users

需要安装httpd-tool包先安装它

|  |
| --- |
| [root@dns logstash-2.4.0]# yum install -y httpd-tool\* |

|  |
| --- |
| [root@dns logstash-2.4.0]# htpasswd -bc /usr/local/nginx/htpasswd.users john 123456 |



之后通过访问需要密码和用户并且是80端口了

9）kibana启动脚本

另，kibana有经常退出的情况，这里用一个shell脚本

脚本文件双击打开

|  |
| --- |
| #!/bin/bash  ### BEGIN INIT INFO  # Provides: kibana  # Default-Start: 2 3 4 5  # Default-Stop: 0 1 6  # Short-Description: Runs kibana daemon  # Description: Runs the kibana daemon as a non-root user  ### END INIT INFO  # Process name  NAME=kibana  DESC="Kibana4"  PROG="/etc/init.d/kibana"  # Source function library.  . /etc/rc.d/init.d/functions    # Configure location of Kibana bin  KIBANA\_BIN=/usr/local/kibana-4.6.1-linux-x86\_64/bin    # PID Info  PID\_FOLDER=/var/run/kibana/  PID\_FILE=/var/run/kibana/$NAME.pid  LOCK\_FILE=/var/lock/subsys/$NAME  PATH=/bin:/usr/bin:/sbin:/usr/sbin:$KIBANA\_BIN  DAEMON=$KIBANA\_BIN/$NAME  # Configure User to run daemon process  DAEMON\_USER=root  # Configure logging location  KIBANA\_LOG=/var/log/kibana.log  # Begin Script  RETVAL=0  if [ `id -u` -ne 0 ];then  echo "You need root privileges to run this script"  exit 1  fi  start() {  echo -n "Starting $DESC : "  pid=`pidofproc -p $PID\_FILE kibana`  if [ -n "$pid" ] ; then  echo "Already running."  exit 0  else  # Start Daemon  if [ ! -d "$PID\_FOLDER" ];then  mkdir $PID\_FOLDER  fi  daemon --user=$DAEMON\_USER --pidfile=$PID\_FILE $DAEMON 1>"$KIBANA\_LOG" 2>&1 &  sleep 2  pidofproc node > $PID\_FILE  RETVAL=$?  [[ $? -eq 0 ]] && success || failure  echo  [ $RETVAL = 0 ] && touch $LOCK\_FILE  return $RETVAL  fi  }  reload()  {  echo "Reload command is not implemented for this service."  return $RETVAL  }  stop() {  echo -n "Stopping $DESC : "  killproc -p $PID\_FILE $DAEMON  RETVAL=$?  echo  [ $RETVAL = 0 ] && rm -f $PID\_FILE $LOCK\_FILE  }  case "$1" in  start)  start  ;;  stop)  stop  ;;  status)  status -p $PID\_FILE $DAEMON  RETVAL=$?  ;;  restart)  stop  start  ;;  reload)  reload  ;;  \*)  # Invalid Arguments, print the following message.  echo "Usage: $0 {start|stop|status|restart}" >&2  exit 2  ;;  esac |

添加执行权限，启动

|  |
| --- |
| # chmod +x /etc/init.d/kibana  # service kibana restart |