黑客派

ELK(ElasticSearch, Logstash, Kibana)搭建实时日志分析平台

作者：unhappydepig

原文链接：https://hacpai.com/article/1470035918872

来源网站：黑客派

<h2>ELK平台介绍</h2>
在搜索ELK资料的时候，发现这篇文章比较好，于是摘抄一小段：
以下内容来自：http://baidu.blog.5cto.com/71938/1676798
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，性能全性，从而及时采取措施纠正错误。
通常，日志被分散的储存不同的设备上。如果你管理数十上百台服务器，你还在使用依次登录每机器的传统方法查阅日志。这样是不是感觉很繁琐和效率低下。当务之急我们使用集中化的日志管理例如：开源的syslog，将所有服务器上的日志收集汇总。
集中化管理日志后，日志的统计和检索又成为一件比较麻烦的事情，一般我们使用grep、awk和c等Linux命令能实现检索和统计，但是对于要求更高的查询、排序和统计等要求和庞大的机器数量依使用这样的方法难免有点力不从心。
开源实时日志分析ELK平台能够完美的解决我们上述的问题，ELK由ElasticSearch、Logstash和Kabana三个开源工具组成。官方网站：https://www.elastic co/products

 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。

----------摘抄内容结束----------
画了一个ELK工作的原理图：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/123049_meyt_1434710.png">
如图：Logstash收集AppServer产生的Log，并存放到ElasticSearch集群中，而Kibana则从ES群中查询数据生成图表，再返回给Browser。

<h2>ELK平台搭建</h2>
<h4>系统环境</h4>
System: Centos release 6.7 (Final)
ElasticSearch: 2.1.0
Logstash: 2.1.1
Kibana: 4.3.0
Java: openjdk version "1.8.0_65"
注：由于Logstash的运行依赖于Java环境， 而Logstash 1.5以上版本不低于java 1.7，因此推荐用最新版本的Java。因为我们只需要Java的运行环境，所以可以只安装JRE，不过这里我依然使用JD，请自行搜索安装。

ELK下载：https://www.elastic.co/downloads
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/123318_IHwC_1434710.png">

<h4>ElasticSearch</h4>
配置ElasticSearch：
<pre><code class="hljs css">tar <span class

"hljs-selector-tag">-zxvf elasticsearch-2</sp
n>.1.0<
pan class="hljs-selector-class">.tar.gz
cd elasticsea
ch-2.1.
</code></pre>
安装Head插件（Optional）：
<pre><code class="hljs sql">./bin/plugin install
nbsp;mobz/elasticsearch-head</code></pre>
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://stat
c.oschina.net/uploads/space/2015/1218/123433_8VxM_1434710.png">
然后编辑ES的配置文件：
<pre><code class="hljs nginx">>vi config/elastic
earch.yml</code></pre>
修改以下配置项：
<pre><code class="hljs haskell">cluster.name=es_cluster
node.name=node0
path.>
ata=/tmp/elasticsearch/data
path.logs=/tmp/elasticsearch/logs
#当前hostname或IP，我这里是centos2
network.host=centos2
network.port=9200</
ode></pre>
其他的选项保持默认，然后启动ES：
<pre><code class="hljs groovy">./bin/elasticsearch</co
e></pre>
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://stat
c.oschina.net/uploads/space/2015/1218/123556_6dOb_1434710.png">
可以看到，它跟其他的节点的传输端口为9300，接受HTTP请求的端口为9200。
使用ctrl+C停止。当然，也可以使用后台进程的方式启动ES：
<pre><code class="hljs groovy">./bin/elasticsearch&nb
p;&</code></pre>
然后可以打开页面localhost:9200，将会看到以下内容：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://stat
c.oschina.net/uploads/space/2015/1218/123709_mUHy_1434710.png">
返回展示了配置的cluster_name和name，以及安装的ES的版本等信息。
刚刚安装的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do
内容、执行搜索和普通的Rest请求等。现在也可以使用它打开<a href="https://link.hacpai.com/for
ard?goto=http%3A%2F%2Flocalhost%3A9200%2F_plugin%2Fhead" rel="nofollow ugc" targe
="_blank">localhost:9200/_plugin/head页面来查看ES集群状态：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://stat
c.oschina.net/uploads/space/2015/1218/124024_bnOW_1434710.png">
可以看到，现在，ES集群中没有index，也没有type，因此这两条是空的。</
>

<h4>Logstash</h4>
Logstash的功能如下：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://stat
c.oschina.net/uploads/space/2015/1218/124206_UKKk_1434710.png">
其实它就是一个收集器而已，我们需要为它指定Input和Output（当然Inpu
和Output可以为多个）。由于我们需要把Java代码中Log4j的日志输出到ElasticSearch中，因此这里
Input就是Log4j，而Output就是ElasticSearch。
配置Logstash：

<pre><code class="hljs css">tar <span class
"hljs-selector-tag">-zxvf logstash-2
span class="hljs-selector-class">.1.1.tar.gz
cd logstash-
.1.1</s
an></code></pre>

编写配置文件(名字和位置可以随意，这里我放在config目录下，取名为log4j_to_es.conf)：

<pre><code class="hljs bash">mkdir config
vi config/log4j_to_es.conf</code></pre>

输入以下内容：

<pre><code class="hljs php"># For detail
tructure of this file
Set: https://www.elastic.co/guide/en/logstash/c
rrent/configuration-file-structure.html
input {
  # For detail config fo
 log4j as input,
  # See: https://www.elastic.co/guide
en/logstash/current/plugins-inputs-log4j.html
  log4j {
    mode => "server"</s
an>
    host => "centos2"</s
an>
    port => 4567</spa
>
  }
}
filter {
  #Only matched data are&nb
p;send to output.
}
output {
  # For detail config fo
 elasticsearch as output,
  # See: https://www.elastic.co/guide
en/logstash/current/plugins-outputs-elasticsearch.html
  elasticsearch {
    action => "index"</s
an>   #The
peration on ES
    hosts => "cen
os2:9200" #ElasticSearch host, can&nbs
;be array.
    index => "app
og"   #The
ndex to write data to.
  }
}</code></pre>

logstash命令只有2个参数：

<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://stat
c.oschina.net/uploads/space/2015/1218/124351_c5xc_1434710.png">

因此使用agent来启动它（使用-f指定配置文件）：
<pre><code class="hljs bash">./bin/logstash agent -f</s
an> config/log4j_to_es.conf</code></pre>
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://stat
c.oschina.net/uploads/space/2015/1218/124427_bwLM_1434710.png">
到这里，我们已经可以使用Logstash来收集日志并保存到ES中了，下面来看看项目代码。

<h4>Java项目</h4>
照例先看项目结构图：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://stat
c.oschina.net/uploads/space/2015/1218/124604_vZcZ_1434710.png">
pom.xml，很简单，只用到了Log4j库：
<pre><code class="hljs xml"><depend
ncy>
    <groupId</
pan>>log4j</groupId</spa
>>
    <artifactId<
span>>log4j</artifactId</sp
n>>
    <version</
pan>>1.2.17</version</spa
>>
</dependency></c
de></pre>

log4j.properties，将Log4j的日志输出到SocketAppender</spa
>，因为官网是这么说的：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://stat
c.oschina.net/uploads/space/2015/1218/125003_gGXs_1434710.png">
<pre><code class="hljs bash">>log4j.rootLogger=INFO,
onsole
for package com.demo.elk, lo
 would be sent to socket appender.
log4j.logger.com.demo.elk=DEBUG, socket
# appender socket
log4j.appender.socket=org.apache.log4j.net.SocketAppe
der
log4j.appender.socket.Port=4567
log4j.appender.socket.RemoteHost=centos2
log4j.appender.socket.layout=org.apache.log4j.PatternL
yout
log4j.appender.socket.layout.ConversionPattern=%d&nb
p;[%-5p] [%l] %m%n
log4j.appender.socket.ReconnectionDelay=10000
</code> <code class="hljs bash">># appender
console
log4j.appender.console=org.apache.log4j.ConsoleAppen
er
log4j.appender.console.target=System.out
log4j.appender.console.layout=org.apache.log4j.Pattern
ayout
log4j.appender.console.layout.ConversionPattern=%d&
bsp;[%-5p] [%l] %m%n</code> </pre>

注意：这里的端口号需要跟Logstash监听的端口号一致，这里是4567。

Application.java，使用Log4j的LOGGER打印日志即可：
<pre><code class="hljs java">package com.demo.elk;
import org.apache.log4j.Logger;
</code> <code class="hljs java">public class Application {
    private static final Logger LOGGER = Logger.getLogger(Application.class);
    public static void main(String[] args) throws Exception {
        for (int i = 0; i < 10; i++) {
            LOGGER.error("Info log [" + i + "].");
             Thread.sleep(500);
        }
    }
}</code> </pre>

<h4>用Head插件查看ES状态和内容</h4>
运行Application.java，先看看console的输出（当然，这个输出只是为了做验证，不输出到console也可以的）：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/125311_sWzx_1434710.png">
再来看看ES的head页面：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/125414_3JK6_1434710.png">
切换到Browser标签：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/125433_AULZ_1434710.png">
单击某一个文档（doc），则会展示该文档的所有信息：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/125446_mqmh_1434710.png">
可以看到，除了基础的message字段是我们的日志内容，Logstash还为我们增加了许多字段。在https://www.elastic.co/guide/en/logstash/current/plugins-inputs-log4j.html中也明说明了这一点：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/125504_NcYf_1434710.png">

上面使用了ES的Head插件观察了ES集群的状态和数据，但这只是个简单的用于跟ES交互的页面已，并不能生成报表或者图表什么的，接下来使用Kibana来执行搜索并生成图表。

<h4>Kibana</h4>
配置Kibana:
<pre><code class="hljs bash">tar -zxvf kibana-4.3.0-linux-x86.tar.gz
cd kibana-4.3.0-linux-x86
vi config/kibana.yml</code></pre>
修改以下几项（由于是单机版的，因此host的值也可以使用localhost来代替，这里仅仅为演示）：
<pre><code class="hljs groovy">server.port: 5601
server.host: “centos2”
elasticsearch.url: http://centos2:9200
kibana.index: “.kibana”</code></pre>
启动kibana：
<pre><code class="hljs groovy">./bin/kibana</code></pre>
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/125828_PLcj_1434710.png">
用浏览器打开该地址：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/125853_ABJt_1434710.png">
为了后续使用Kibana，需要配置至少一个Index名字或者Pattern，它用于在分析时确定ES中的Index。这里我输入之前配置的Index名字applog，Kibana会自动加载该Index下doc的field，并自动选合适的field用于图标中的时间字段：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/125922_L2FT_1434710.png">
点击Create后，可以看到左侧增加了配置的Index名字：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/130003_bcKa_1434710.png">
接下来切换到Discover标签上，注意右上角是查询的时间范围，如果没有查找到数据，那么你就能需要调整这个时间范围了，这里我选择Today：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/130040_cA0j_1434710.png">
接下来就能看到ES中的数据了：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/130130_3reO_1434710.png">
执行搜索看看呢：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/130304_xbvj_1434710.png">
点击右边的保存按钮，保存该查询为search_all_logs。接下来去Visualize页面，点击新建一个柱图（Vertical Bar Chart），然后选择刚刚保存的查询search_all_logs，之后，Kibana将生成类似于图的柱状图（只有10条日志，而且是在同一时间段的，比较丑，但足可以说明问题了：） ）：/p>
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/130447_es2k_1434710.png">
你可以在左边设置图形的各项参数，点击Apply Changes按钮，右边的图形将被更新。同理，其类型的图形都可以实时更新。
点击右边的保存，保存此图，命名为search_all_logs_visual。接下来切换到Dashboard页面：</>
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/130543_elQl_1434710.png">

单击新建按钮，选择刚刚保存的search_all_logs_visual图形，面板上将展示该图：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/130607_vH1C_1434710.png">
如果有较多数据，我们可以根据业务需求和关注点在Dashboard页面添加多个图表：柱形图，折图，地图，饼图等等。当然，我们可以设置更新频率，让图表自动更新：
<img src="https://static.hacpai.com/images/img-loading.svg" alt="" data-src="http://static.oschina.net/uploads/space/2015/1218/130638_2jqP_1434710.png">
如果设置的时间间隔够短，就很趋近于实时分析了。
到这里，ELK平台部署和基本的测试已完成。

参考：
http://baidu.blog.51cto.co/71938/1676798
http://blog.csdn.ne/cnweike/article/details/33736429

转自:http://my.oschina.net/itblog/log/547250