|  |
| --- |
| ELK收集日志 |
| **作者：老男孩Linux教育-张亚**  **归档：上课文档**  **2019/6/30** |
| **快捷键：**  Ctrl + 1 标题1  Ctrl + 2 标题2  Ctrl + 3 标题3  Ctrl + 4 实例  Ctrl + 5 程序代码  Ctrl + 6 正文 |
| **格式说明：**  蓝色字体：注释  黄色背景：重要  绿色背景：注意 |

目 录

[第1章 ELK介绍 1](#_Toc10366827)

[第2章 日志收集分类 1](#_Toc10366828)

[第3章 安装部署ELK 1](#_Toc10366829)

[3.1 官方地址 1](#_Toc10366830)

[3.2 安装配置java 1](#_Toc10366831)

[3.3 更新时间 1](#_Toc10366832)

[3.4 安装配置elasticsearch 1](#_Toc10366833)

[3.4.1 相关配置目录及配置文件 2](#_Toc10366834)

[3.4.2 ES配置文件解读 2](#_Toc10366835)

[3.4.3 启动失败报错 3](#_Toc10366836)

[3.4.4 锁定内存失败解决方案 3](#_Toc10366837)

[3.4.5 检查启动是否成功 3](#_Toc10366838)

[3.5 安装配置es-head插件 4](#_Toc10366839)

[3.5.1 插件官方地址 4](#_Toc10366840)

[3.5.2 使用docker部署elasticsearch-head 4](#_Toc10366841)

[3.5.3 使用nodejs编译安装 4](#_Toc10366842)

[3.5.4 修改ES配置文件支持跨域 5](#_Toc10366843)

[3.5.5 网页访问 5](#_Toc10366844)

[3.6 安装配置kibana 5](#_Toc10366845)

[3.6.1 安装配置kibana 5](#_Toc10366846)

[3.7 安装filebeat和logstash 5](#_Toc10366847)

[3.8 docker安装elk 5](#_Toc10366848)

[第4章 使用filebeat配置日志收集 7](#_Toc10366849)

[4.1 收集nginx日志 7](#_Toc10366850)

[4.1.1 安装nginx和ab工具 7](#_Toc10366851)

[4.1.2 启动测试 7](#_Toc10366852)

[4.1.3 查看日志 7](#_Toc10366853)

[4.1.4 配置filebeat收集普通日志 7](#_Toc10366854)

[4.1.5 kibana配置 8](#_Toc10366855)

[4.1.6 修改nginx日志为json格式 8](#_Toc10366856)

[4.1.7 查看日志是否为json格式 8](#_Toc10366857)

[4.1.8 收集多个日志并分类创建索引 9](#_Toc10366858)

[4.2 收集tomcat日志 9](#_Toc10366859)

[4.2.1 安装tomcat 9](#_Toc10366860)

[4.2.2 启动检查 10](#_Toc10366861)

[4.2.3 访问测试 10](#_Toc10366862)

[4.2.4 修改日志为json格式 10](#_Toc10366863)

[4.2.5 重启确认日志是否为json格式 10](#_Toc10366864)

[4.2.6 filebeat配置 11](#_Toc10366865)

[4.3 收集java日志 12](#_Toc10366866)

[4.3.1 官方地址 12](#_Toc10366867)

[4.3.2 filebeat配置 12](#_Toc10366868)

[4.4 收集docker日志 12](#_Toc10366869)

[4.4.1 安装docker 12](#_Toc10366870)

[4.4.2 运行nginx镜像 14](#_Toc10366871)

[4.4.3 配置filebeat收集单个docker日志 14](#_Toc10366872)

[4.4.4 配置filebeat通过标签收集多个容器日志 14](#_Toc10366873)

[4.4.5 配置filebeat通过服务类型和日志类型多条件创建不同索引 17](#_Toc10366874)

[4.4.6 验证提交新镜像运行后日志收集情况 18](#_Toc10366875)

[第5章 使用模版配置 20](#_Toc10366876)

[5.1 官方网址 20](#_Toc10366877)

[5.2 使用模版配置nginx正常日志 20](#_Toc10366878)

[5.2.1 filebeat配置文件 20](#_Toc10366879)

[5.2.2 filebeat modules配置 20](#_Toc10366880)

[5.3 导入kibana视图 21](#_Toc10366881)

[第6章 使用缓存收集日志 22](#_Toc10366882)

[6.1 使用redis作为缓存 22](#_Toc10366883)

[6.1.1 安装启动测试redis 22](#_Toc10366884)

[6.1.2 配置nginx的json日志 22](#_Toc10366885)

[6.1.3 配置filebeat写入到不同的key中 23](#_Toc10366886)

[6.1.4 配置logstash读取不同的key 23](#_Toc10366887)

[6.1.5 filebeat收集日志写入到一个key中 24](#_Toc10366888)

[6.1.6 logstash根据tag区分一个key里的不同日志 25](#_Toc10366889)

[6.2 使用kafka作为缓存 26](#_Toc10366890)

[6.2.1 filebeat配置 26](#_Toc10366891)

[6.2.2 logstash配置 27](#_Toc10366892)

[第7章 画图展示 27](#_Toc10366893)

[7.1 kibana页面介绍 27](#_Toc10366894)

[7.2 kibana创建图标 28](#_Toc10366895)

[7.3 kibana创建面板视图 28](#_Toc10366896)

[第8章 日志分析脚本 28](#_Toc10366897)

[第9章 待解决问题 32](#_Toc10366898)

# ELK介绍

没有日志收集平台的时候如何分析日志？什么是ELK/EFLK? ELK的数据流转流程?

官方演示地址：

https://demo.elastic.co/

# 日志收集分类

代理层：nginx,haproxy

web层：nginx,tomcat

数据库层：mysql,redis,mongo,elasticsearch

操作系统层：source,message

# 安装部署ELK

## 官方地址

https://www.elastic.co/guide/index.html

## 安装配置java

[root@elk-175 ~]# yum install java-1.8.0-openjdk.x86\_64 -y

[root@elk-175 ~]# java -version

openjdk version "1.8.0\_212"

OpenJDK Runtime Environment (build 1.8.0\_212-b04)

OpenJDK 64-Bit Server VM (build 25.212-b04, mixed mode)

## 更新时间

yum install ntpdate -y

ntpdate time1.aliyun.com

## 安装配置elasticsearch

[root@elk-175 ~]# mkdir /data/soft/ -p

[root@elk-175 ~]# cd /data/soft/

[root@elk-175 soft]# rz

[root@elk-175 soft]# ll

总用量 506120

-rw-r--r-- 1 root root 114059630 2月 25 11:09 elasticsearch-6.6.0.rpm

-rw-r--r-- 1 root root 36581177 4月 27 12:29 elasticsearch-head.tar.gz

-rw-r--r-- 1 root root 11790119 2月 25 11:08 filebeat-6.6.0-x86\_64.rpm

-rw-r--r-- 1 root root 185123116 2月 25 11:11 kibana-6.6.0-x86\_64.rpm

-rw-r--r-- 1 root root 170703770 2月 25 11:38 logstash-6.6.0.rpm

[root@elk-175 soft]# rpm -ivh elasticsearch-6.6.0.rpm

警告：elasticsearch-6.6.0.rpm: 头V4 RSA/SHA512 Signature, 密钥 ID d88e42b4: NOKEY

准备中... ################################# [100%]

Creating elasticsearch group... OK

Creating elasticsearch user... OK

正在升级/安装...

1:elasticsearch-0:6.6.0-1 ################################# [100%]

### NOT starting on installation, please execute the following statements to configure elasticsearch service to start automatically using systemd

sudo systemctl daemon-reload

sudo systemctl enable elasticsearch.service

### You can start elasticsearch service by executing

sudo systemctl start elasticsearch.serviceCreated elasticsearch keystore in /etc/elasticsearch

### 相关配置目录及配置文件

[root@elk-175 ~]# rpm -qc elasticsearch

/etc/elasticsearch/elasticsearch.yml

/etc/elasticsearch/jvm.options

/etc/elasticsearch/log4j2.properties

/etc/elasticsearch/role\_mapping.yml

/etc/elasticsearch/roles.yml

/etc/elasticsearch/users

/etc/elasticsearch/users\_roles

/etc/init.d/elasticsearch

/etc/sysconfig/elasticsearch

/usr/lib/sysctl.d/elasticsearch.conf

/usr/lib/systemd/system/elasticsearch.service

### ES配置文件解读

[root@elk-175 soft]# vim /etc/elasticsearch/elasticsearch.yml

[root@elk-175 soft]# grep ^[a-z] /etc/elasticsearch/elasticsearch.yml

node.name: node-1

path.data: /var/lib/elasticsearch

path.logs: /var/log/elasticsearch

bootstrap.memory\_lock: true

network.host: 192.168.47.175,127.0.0.1

http.port: 9200

[root@elk-175 soft]# systemctl daemon-reload

[root@elk-175 soft]# systemctl enable elasticsearch.service

Created symlink from /etc/systemd/system/multi-user.target.wants/elasticsearch.service to /usr/lib/systemd/system/elasticsearch.service.

[root@elk-175 soft]# systemctl start elasticsearch.service

### 启动失败报错

此时启动会发现失败，原因是内存锁定失败

[2019-05-12T14:39:58,764][ERROR][o.e.b.Bootstrap ] [node-1] node validation exception

[1] bootstrap checks failed

[1]: memory locking requested for elasticsearch process but memory is not locked

### 锁定内存失败解决方案

官方解决方案

https://www.elastic.co/guide/en/elasticsearch/reference/6.4/setup-configuration-memory.html

https://www.elastic.co/guide/en/elasticsearch/reference/6.4/setting-system-settings.html#sysconfig

### 修改启动配置文件

vim /usr/lib/systemd/system/elasticsearch.service

### 增加如下参数

[Service]

LimitMEMLOCK=infinity

### 重新启动

systemctl daemon-reload

systemctl restart elasticsearch

### 检查启动是否成功

[root@elk-175 ~]# netstat -lntup|grep 9200

tcp6 0 0 192.168.47.175:9200 :::\* LISTEN 15824/java

tcp6 0 0 127.0.0.1:9200 :::\* LISTEN 15824/java

[root@elk-175 ~]# curl localhost:9200

{

"name" : "node-1",

"cluster\_name" : "elasticsearch",

"cluster\_uuid" : "As5ZlEQ2Syq0ktLL0hg5XA",

"version" : {

"number" : "6.6.0",

"build\_flavor" : "default",

"build\_type" : "rpm",

"build\_hash" : "a9861f4",

"build\_date" : "2019-01-24T11:27:09.439740Z",

"build\_snapshot" : false,

"lucene\_version" : "7.6.0",

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

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

},

"tagline" : "You Know, for Search"

}

## 服务模式安装配置es-head插件

### 插件官方地址

https://github.com/mobz/elasticsearch-head

### 使用docker部署elasticsearch-head

docker pull alivv/elasticsearch-head

docker run --name es-head -p 9100:9100 -dit alivv/elasticsearch-head

### 使用nodejs编译安装

官网地址

https://nodejs.org/en/download/package-manager/

https://nodejs.org/dist/latest-v10.x/

http://npm.taobao.org

下载安装

yum install nodejs npm openssl screen -y

node -v

npm -v

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

cd /opt/

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

cd elasticsearch-head/

cnpm install

screen -S es-head

cnpm run start

Ctrl+A+D

### 修改ES配置文件支持跨域

官方说明：

https://www.elastic.co/guide/en/elasticsearch/reference/6.6/modules-http.html

配置参数：

http.cors.enabled: true

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

### 网页访问

IP地址:9100

## 谷歌浏览器插件形式安装es-head

使用服务模式安装es-head插件过程比较繁琐，网络不好的时候还会经常卡住

幸运的是es-head插件官方还提供了另外一种更简便的方式，就是google chrome的插件

优势如下；

1.免安装，直接下载插件安装在浏览器即可

2.只要浏览器和服务器可以通信就能使用

## 安装配置kibana

### 安装配置kibana

[root@elk-175 soft]# rpm -ivh kibana-6.6.0-x86\_64.rpm

[root@elk-175 soft]# grep "^[a-z]" /etc/kibana/kibana.yml

server.port: 5601

server.host: "192.168.47.175"

elasticsearch.hosts: ["http://localhost:9200"]

kibana.index: ".kibana

[root@elk-175 soft]# systemctl start kibana

[root@elk-175 soft]# systemctl status kibana

[root@elk-175 soft]# netstat -lntup|grep 5601

tcp 0 0 192.168.47.175:5601 0.0.0.0:\* LISTEN 16442/node

## 安装filebeat和logstash

rpm -ivh filebeat-6.6.0-x86\_64.rpm

rpm -ivh logstash-6.6.0.rpm

## docker安装elk

version: '3'

services:

elasticsearch:

image: elasticsearch:v1

container\_name: elasticsearch

environment:

- cluster.name=docker-cluster

- bootstrap.memory\_lock=true

- "ES\_JAVA\_OPTS=-Xms8g -Xmx8g"

- "discovery.zen.ping.unicast.hosts=elasticsearch"

ulimits:

memlock:

soft: -1

hard: -1

volumes:

- /data/docker\_es\_data:/usr/share/elasticsearch/data

ports:

- 19200:9200

networks:

- esnet

elasticsearch-head:

image: elasticsearch-head:v1

container\_name: elasticsearch-head

ports:

- 19100:9100

networks:

- esnet

kibana:

image: kibana:v1

container\_name: kibana

environment:

- ELASTICSEARCH\_URL="http://elasticsearch:9200"

- kibana.index=".kibana"

ports:

- 15601:5601

networks:

- esnet

logstash:

image: logstash:v1

container\_name: logstash

environment:

- ELASTICSEARCH\_URL="http://elasticsearch:9200"

ports:

- 16379:6379

networks:

- esnet

networks:

esnet:

# 使用filebeat配置日志收集

## 收集nginx日志

### 安装nginx和ab工具

[root@elk-175 ~]# yum install nginx httpd-tool -y

### 启动测试

[root@elk-175 ~]# systemctl start nginx

[root@elk-175 ~]# netstat -lntup|grep nginx

tcp 0 0 0.0.0.0:80 0.0.0.0:\* LISTEN 1888/nginx: master

tcp6 0 0 :::80 :::\* LISTEN 1888/nginx: master

[root@elk-175 ~]# ab -c 10 -n 100 192.168.47.175/

[root@elk-175 ~]# ab -c 10 -n 100 192.168.47.175/test.html

### 查看日志

[root@elk-175 ~]# tail -f /var/log/nginx/access.log

192.168.47.175 - - [13/May/2019:00:24:02 +0800] "GET /test.html HTTP/1.0" 404 3650 "-" "ApacheBench/2.3" "-"

192.168.47.175 - - [13/May/2019:00:24:02 +0800] "GET /test.html HTTP/1.0" 404 3650 "-" "ApacheBench/2.3" "-"

.......................................

### 配置filebeat收集普通日志

相关网址

https://www.elastic.co/guide/en/beats/filebeat/7.x/filebeat-input-log.html

https://www.elastic.co/guide/en/beats/filebeat/7.x/configuration-template.html

https://www.elastic.co/guide/en/beats/filebeat/7.x/configuration-template.html

正确配置

filebeat.inputs:

- type: log

enabled: true

paths:

- /var/log/nginx/access.log

setup.kibana:

host: "192.168.47.175:5601"

output.elasticsearch:

hosts: ["localhost:9200"]

index: "nginx-%{[beat.version]}-%{+yyyy.MM.dd}"

setup.template.name: "nginx"

setup.template.pattern: "nginx-\*"

setup.template.enabled: false

setup.template.overwrite: true

### kibana配置

management-->index patterns-->nginx-->@timestamp

### 修改nginx日志为json格式

log\_format main '{ "time\_local": "$time\_local", '

'"remote\_addr": "$remote\_addr", '

'"referer": "$http\_referer", '

'"request": "$request", '

'"status": $status, '

'"bytes": $body\_bytes\_sent, '

'"agent": "$http\_user\_agent", '

'"x\_forwarded": "$http\_x\_forwarded\_for", '

'"up\_addr": "$upstream\_addr",'

'"up\_host": "$upstream\_http\_host",'

'"upstream\_time": "$upstream\_response\_time",'

'"request\_time": "$request\_time"'

' }';

### 查看日志是否为json格式

[root@elk-175 ~]# tail -f /var/log/nginx/access.log

{ "time\_local": "13/May/2019:00:30:51 +0800", "remote\_addr": "192.168.47.175", "referer": "-", "request": "GET /test.html HTTP/1.0", "status": 404, "bytes": 3650, "agent": "ApacheBench/2.3", "x\_forwarded": "-", "up\_addr": "-","up\_host": "-","upstream\_time": "-","request\_time": "0.000" }

{ "time\_local": "13/May/2019:00:30:51 +0800", "remote\_addr": "192.168.47.175", "referer": "-", "request": "GET /test.html HTTP/1.0", "status": 404, "bytes": 3650, "agent": "ApacheBench/2.3", "x\_forwarded": "-", "up\_addr": "-","up\_host": "-","upstream\_time": "-","request\_time": "0.000" }

### 收集多个日志并分类创建索引

filebeat配置

[root@elk-175 filebeat]# cat filebeat.yml

filebeat.inputs:

- type: log

enabled: true

paths:

- /var/log/nginx/access.log

json.keys\_under\_root: true

json.overwrite\_keys: true

tags: ["access"]

- type: log

enabled: true

paths:

- /var/log/nginx/error.log

tags: ["error"]

setup.template.settings:

index.number\_of\_shards: 3

setup.kibana:

host: "192.168.47.175:5601"

output.elasticsearch:

hosts: ["localhost:9200"]

indices:

- index: "nginx\_access-%{[beat.version]}-%{+yyyy.MM.dd}"

when.contains:

tags: "access"

- index: "nginx\_error-%{[beat.version]}-%{+yyyy.MM.dd}"

when.contains:

tags: "error"

setup.template.name: "nginx"

setup.template.pattern: "nginx\_\*"

setup.template.enabled: false

setup.template.overwrite: true

## 收集tomcat日志

### 安装tomcat

yum install tomcat tomcat-webapps tomcat-admin-webapps tomcat-docs-webapp tomcat-javadoc -y

### 启动检查

[root@elk-175 ~]# systemctl start tomcat

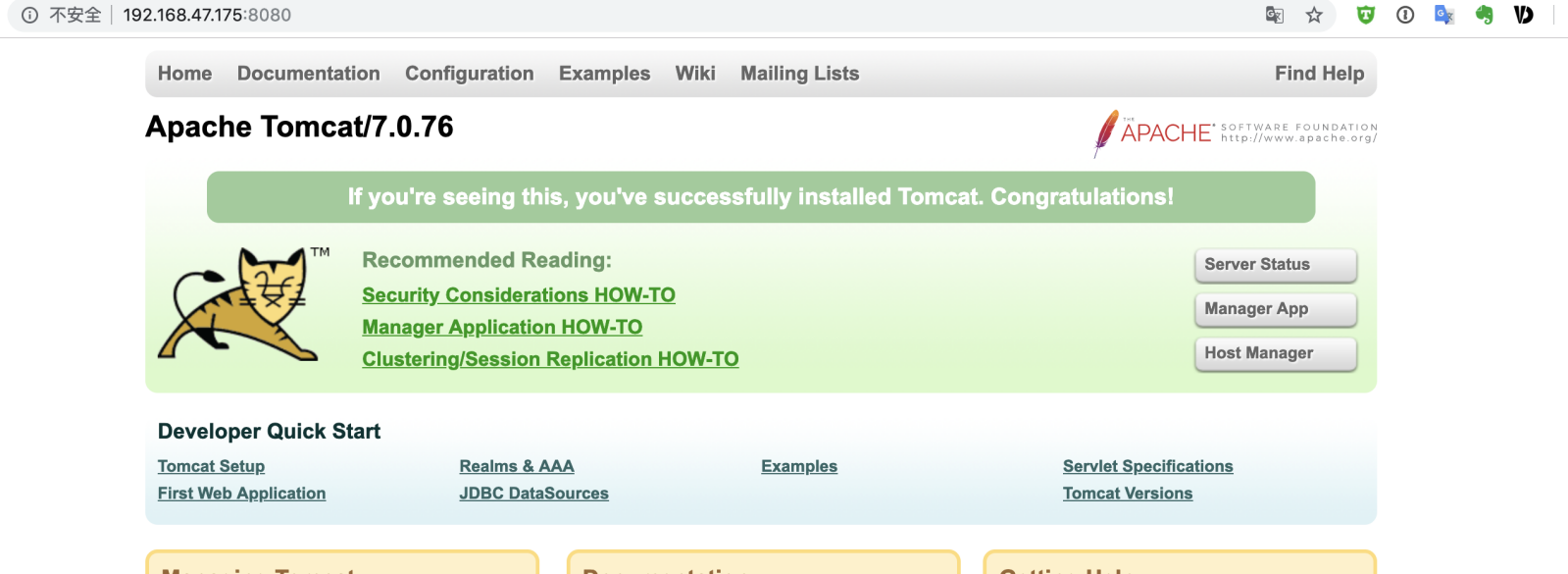
[root@elk-175 ~]# systemctl status tomcat

[root@elk-175 ~]# lsof -i:8080

COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME

java 18915 tomcat 49u IPv6 61950 0t0 TCP \*:webcache (LISTEN)

### 访问测试



### 修改日志为json格式

[root@elk-175 ~]# vim /etc/tomcat/server.xml

[root@elk-175 ~]# cat -n /etc/tomcat/server.xml

----------------

137 <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"

138 prefix="localhost\_access\_log." suffix=".txt"

139 pattern="{&quot;clientip&quot;:&quot;%h&quot;,&quot;ClientUser&quot;:&quot;%l&quot;,&quot;authenticated&quot;:&quot;%u&quot;,&quot;AccessTime&quot;:&quot;%t&quot;,&quot;method&quot;:&quot;%r&quot;,&quot;status&quot;:&quot;%s&quot;,&quot;SendBytes&quot;:&quot;%b&quot;,&quot;Query?string&quot;:&quot;%q&quot;,&quot;partner&quot;:&quot;%{Referer}i&quot;,&quot;AgentVersion&quot;:&quot;%{User-Agent}i&quot;}"/>

----------------

### 重启确认日志是否为json格式

[root@elk-175 ~]# systemctl restart tomcat

[root@elk-175 ~]# tail -f /var/log/tomcat/localhost\_access\_log.2019-05-13.txt

{"clientip":"192.168.47.1","ClientUser":"-","authenticated":"-","AccessTime":"[13/May/2019:13:18:03 +0800]","method":"GET /docs/images/tomcat.gif HTTP/1.1","status":"200","SendBytes":"2066","Query?string":"","partner":"http://192.168.47.175:8080/docs/realm-howto.html","AgentVersion":"Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_14\_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/72.0.3626.109 Safari/537.36"}

在线解析测试正确



### filebeat配置

[root@elk-175 ~]# cat /etc/filebeat/filebeat.yml

filebeat.inputs:

- type: log

enabled: true

paths:

- /var/log/tomcat/localhost\_access\_log\*

json.keys\_under\_root: true

json.overwrite\_keys: true

setup.kibana:

host: "192.168.47.175:5601"

output.elasticsearch:

hosts: ["localhost:9200"]

index: "tomcat-%{[beat.version]}-%{+yyyy.MM.dd}"

setup.template.name: "tomcat"

setup.template.pattern: "tomcatn-\*"

setup.template.enabled: false

setup.template.overwrite: true

### 配置tomcat收集多个域名的日志

配置多个host标签

## 收集java日志

### 官方地址

https://www.elastic.co/guide/en/beats/filebeat/6.6/multiline-examples.html

因为java日志的输出信息非常多，需要将多行拼成一个事件，所以需要多行匹配模式

因为elasticsearch本身就是java开发的，所以我们可以直接收集ES的日志

### filebeat配置

[root@elk-175 ~]# cat /etc/filebeat/filebeat.yml

filebeat.inputs:

- type: log

enabled: true

paths:

- /var/log/elasticsearch/elasticsearch.log

multiline.pattern: '^\['

multiline.negate: true

multiline.match: after

setup.kibana:

host: "192.168.47.175:5601"

output.elasticsearch:

hosts: ["localhost:9200"]

index: "es-%{[beat.version]}-%{+yyyy.MM.dd}"

setup.template.name: "es"

setup.template.pattern: "es-\*"

setup.template.enabled: false

setup.template.overwrite: true

## 收集docker日志

### 安装docker

rm -fr /etc/yum.repos.d/local.repo

curl -o /etc/yum.repos.d/CentOS-Base.repo http://mirrors.aliyun.com/repo/Centos-7.repo

wget -O /etc/yum.repos.d/docker-ce.repo https://mirrors.ustc.edu.cn/docker-ce/linux/centos/docker-ce.repo

sed -i 's#download.docker.com#mirrors.tuna.tsinghua.edu.cn/docker-ce#g' /etc/yum.repos.d/docker-ce.repo

yum install docker-ce -y

systemctl start docker

vi /etc/docker/daemon.json

{

"registry-mirrors": ["https://registry.docker-cn.com"]

}

systemctl restart docker

### 运行nginx镜像

docker pull nginx

docker run --name nginx -p 80:80 -d nginx

docker ps

docker logs -f nginx

### 配置filebeat收集单个docker日志

官方介绍：

https://www.elastic.co/guide/en/beats/filebeat/6.7/filebeat-input-docker.html

首先查看docker容器的ID

docker inspect nginx-test|grep -w "Id"

配置文件

filebeat.inputs:

- type: docker

containers.ids:

- '2338d5038f7a2eac96d84d6cf424fb1829bd754ec5e0df944bdd29ba6d61a54e'

tags: ["docker-nginx"]

output.elasticsearch:

hosts: ["localhost:9200"]

index: "docker-nginx-%{[beat.version]}-%{+yyyy.MM.dd}"

setup.template.name: "docker"

setup.template.pattern: "docker-\*"

setup.template.enabled: false

setup.template.overwrite: true

### 使用通配符收集所有容器的日志

新版本的filebeat增加了收集多个容器的日志的选项

https://www.elastic.co/guide/en/beats/filebeat/7.2/filebeat-input-container.html

### 配置filebeat通过标签收集多个容器日志

假如我们有多个docker镜像或者重新提交了新镜像，那么直接指定ID的就不是太方便了。

我们从当前的容器提交一个新的镜像并且运行起来

docker commit nginx nginx:v2

docker images

docker run --name nginx -p 8080:80 -d nginx:v2

此时我们的容器目录下就有了两个不同的容器目录

[root@elk-176 containers]# ls /var/lib/docker/containers/

2338d5038f7a2eac96d84d6cf424fb1829bd754ec5e0df944bdd29ba6d61a54e 3bb5250e7e50a4ed8d1fae7a43d3bf4294864ac4e0af125ae5231cd9bd76b914

如果直接配置filebeat存到es里本台机器所有的容器日志都会混在一起没有办法区分

多容器日志收集处理：

其实收集的日志本质来说还是文件，而这个日志是以容器-json.log命名存放在默认目录下的json格式的文件：

[root@elk-176 ~]# head -1 /var/lib/docker/containers/2338d5038f7a2eac96d84d6cf424fb1829bd754ec5e0df944bdd29ba6d61a54e/2338d5038f7a2eac96d84d6cf424fb1829bd754ec5e0df944bdd29ba6d61a54e-json.log

{"log":"192.168.47.1 - - [23/May/2019:19:12:04 +0000] \"GET / HTTP/1.1\" 200 612 \"-\" \"Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_14\_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.157 Safari/537.36\" \"-\"\n","stream":"stdout","time":"2019-05-23T19:12:04.939212514Z"}

但是每个容器的ID都不一样，为了区分不同服务运行的不同容器，可以使用docker-compose通过给容器添加labels标签来作为区分

然后filbeat把容器日志当作普通的json格式来解析并传输到es

操作步骤：

1.安装docker-compose

yum install -y python2-pip

2.这里使用pip安装，默认源为国外，可以使用国内加速，相关网站

https://mirrors.tuna.tsinghua.edu.cn/help/pypi/

pip加速操作命令

pip install -i https://pypi.tuna.tsinghua.edu.cn/simple pip -U

pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple

3.继续安装docker-compose

pip install docker-compose

4.检查

docker-compose version

5.编写docker-compose.yml

[root@elk-176 ~]# cat docker-compose.yml

version: '3'

services:

nginx:

image: nginx:v2

# 设置labels

labels:

service: nginx

# logging设置增加labels.service

logging:

options:

labels: "service"

ports:

- "8080:80"

db:

image: nginx:latest

# 设置labels

labels:

service: db

# logging设置增加labels.service

logging:

options:

labels: "service"

ports:

- "80:80"

6.清理镜像

docker ps -a|awk 'NR>1{print "docker rm",$1}'|bash

7.运行docker-compose.yml

docker-compose up -d

8.检查日志是否增加了lable标签

[root@elk-176 ~]# tail -1 /var/lib/docker/containers/b2c1f4f7f5a2967fe7d12c1db124ae41f009ec663c71608575a4773beb6ca5f8/b2c1f4f7f5a2967fe7d12c1db124ae41f009ec663c71608575a4773beb6ca5f8-json.log

{"log":"192.168.47.1 - - [23/May/2019:13:22:32 +0000] \"GET / HTTP/1.1\" 304 0 \"-\" \"Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_14\_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.157 Safari/537.36\" \"-\"\n","stream":"stdout","attrs":{"service":"nginx"},"time":"2019-05-23T13:22:32.478708392Z"}

9.配置filebeat

[root@elk-176 ~]# cat /etc/filebeat/filebeat.yml

filebeat.inputs:

- type: log

enabled: true

paths:

- /var/lib/docker/containers/\*/\*-json.log

json.keys\_under\_root: true

json.overwrite\_keys: true

output.elasticsearch:

hosts: ["192.168.47.175:9200"]

indices:

- index: "docker-nginx-%{[beat.version]}-%{+yyyy.MM.dd}"

when.contains:

attrs.service: "nginx"

- index: "docker-db-%{[beat.version]}-%{+yyyy.MM.dd}"

when.contains:

attrs.service: "db"

setup.template.name: "docker"

setup.template.pattern: "docker-\*"

setup.template.enabled: false

setup.template.overwrite: true

### 配置filebeat通过服务类型和日志类型多条件创建不同索引

目前为止，已经可以按服务来收集日志了，但是错误日志和正确日志混在了一起，不好区分，所以可以进一步进行条件判断，根据服务和日志类型创建不同的索引

filebeat配置文件

[root@elk-176 ~]# cat /etc/filebeat/filebeat.yml

filebeat.inputs:

- type: log

enabled: true

paths:

- /var/lib/docker/containers/\*/\*-json.log

json.keys\_under\_root: true

json.overwrite\_keys: true

output.elasticsearch:

hosts: ["192.168.47.175:9200"]

indices:

- index: "docker-nginx-access-%{[beat.version]}-%{+yyyy.MM.dd}"

when.contains:

attrs.service: "nginx"

stream: "stdout"

- index: "docker-nginx-error-%{[beat.version]}-%{+yyyy.MM.dd}"

when.contains:

attrs.service: "nginx"

stream: "stderr"

- index: "docker-db-access-%{[beat.version]}-%{+yyyy.MM.dd}"

when.contains:

attrs.service: "db"

stream: "stdout"

- index: "docker-db-error-%{[beat.version]}-%{+yyyy.MM.dd}"

when.contains:

attrs.service: "db"

stream: "stderr"

setup.template.name: "docker"

setup.template.pattern: "docker-\*"

setup.template.enabled: false

setup.template.overwrite: true

### 验证提交新镜像运行后日志收集情况

1.提交新镜像

[root@elk-176 ~]# docker ps -a

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

f92f4d747584 nginx:latest "nginx -g 'daemon of…" 45 minutes ago Exited (0) 51 seconds ago root\_db\_1

b2c1f4f7f5a2 nginx:v2 "nginx -g 'daemon of…" 45 minutes ago Exited (0) 51 seconds ago root\_nginx\_1

[root@elk-176 ~]# docker commit root\_nginx\_1 nginx:v3

sha256:4457e2b7afc719ef185c75c02031b11c1407efe2e2e57b85f0c9347d04a9ff00

[root@elk-176 ~]# docker commit root\_db\_1 nginx:v4

sha256:a7e8d8b3290c817194956aa06fc486ef928853121d9c6224fd64fe759c967dda

[root@elk-176 ~]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

nginx v4 a7e8d8b3290c 35 seconds ago 109MB

nginx v3 4457e2b7afc7 45 seconds ago 109MB

nginx v2 c181c6355cd9 2 hours ago 109MB

nginx latest 53f3fd8007f7 2 weeks ago 109MB

2.修改并运行docker-compose

[root@elk-176 ~]# cat docker-compose.yml

version: '3'

services:

nginx:

image: nginx:v3

# 设置labels

labels:

service: nginx

# logging设置增加labels.service

logging:

options:

labels: "service"

ports:

- "8080:80"

db:

image: nginx:v4

# 设置labels

labels:

service: db

# logging设置增加labels.service

logging:

options:

labels: "service"

ports:

- "80:80"

[root@elk-176 ~]# docker-compose up -d

Starting root\_nginx\_1 ...

Starting root\_nginx\_1 ... done

Starting root\_db\_1 ... done

[root@elk-176 ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

04308aa3928b nginx:v4 "nginx -g 'daemon of…" 30 seconds ago Up 1 second 0.0.0.0:80->80/tcp root\_db\_1

49d2e2210e6f nginx:v3 "nginx -g 'daemon of…" 30 seconds ago Up 1 second 0.0.0.0:8080->80/tcp root\_nginx\_1

3.访问并查看是否有新数据生成

curl logcalhost/zhangya.html

curl logcalhost:8080/zhangya.html

4.经过查看发现已经成功收集到了日志，这样我们就做到了不用修改filebeat配置文件也可以持续的收集新镜像的日志并按分类创建不同的索引

### 修改docker容器内日志类型为json

刚才收集的docker内的日志类型为普通格式，如果我们修改为json格式会如何呢？

# 使用filledeat modules配置

## 官方网址

https://www.elastic.co/guide/en/beats/filebeat/6.6/configuration-filebeat-modules.html

## 使用模版配置nginx正常日志

社区论坛：

https://discuss.elastic.co/t/filebeat-module-custom-index/181350

### filebeat配置文件

[root@elk-175 ~]# cat /etc/filebeat/filebeat.yml

filebeat.config.modules:

path: ${path.config}/modules.d/\*.yml

reload.enabled: true

setup.kibana:

host: "192.168.47.175:5601"

output.elasticsearch:

hosts: ["localhost:9200"]

indices:

- index: "nginx\_access-%{[beat.version]}-%{+yyyy.MM.dd}"

when.contains:

fileset.name: "access"

- index: "nginx\_error-%{[beat.version]}-%{+yyyy.MM.dd}"

when.contains:

fileset.name: "error"

setup.template.name: "nginx"

setup.template.pattern: "nginx\_\*"

setup.template.enabled: false

setup.template.overwrite: true

### filebeat modules配置

使用nginx模版配置需要安装2个插件，默认从官网下载速度太慢，可以提前下载然后离线安装

https://www.elastic.co/guide/en/elasticsearch/plugins/6.6/ingest-geoip.html

https://www.elastic.co/guide/en/elasticsearch/plugins/6.6/plugin-management-custom-url.html

在线安装：

[root@elk-175 ~]# /usr/share/elasticsearch/bin/elasticsearch-plugin install ingest-user-agent

[root@elk-175 ~]# /usr/share/elasticsearch/bin/elasticsearch-plugin install ingest-geoip

离线下载安装：

[root@elk-175 ~]# wget https://artifacts.elastic.co/downloads/elasticsearch-plugins/ingest-user-agent/ingest-user-agent-6.6.0.zip

[root@elk-175 ~]# wget https://artifacts.elastic.co/downloads/elasticsearch-plugins/ingest-geoip/ingest-geoip-6.6.0.zip

[root@elk-175 ~]# /usr/share/elasticsearch/bin/elasticsearch-plugin install file:///root/ingest-geoip-6.6.0.zip

[root@elk-175 ~]# /usr/share/elasticsearch/bin/elasticsearch-plugin install file:///root/ingest-user-agent-6.6.0.zip

激活nginx模块：

[root@elk-175 ~]# filebeat modules enable nginx

[root@elk-175 ~]# egrep -v "#|^$" /etc/filebeat/modules.d/nginx.yml

- module: nginx

access:

enabled: true

var.paths: ["/var/log/nginx/access.log"]

error:

enabled: true

var.paths: ["/var/log/nginx/error.log"

注意：6.7之后这两个插件默认集成到了elasticsearch，不需要单独安装了

## 使用模块收集系统日志message和secure日志

如果不需要转换，也可以直接按普通日志模式收集message和secure日志

## 导入kibana视图

默认如果使用filbeat模版导入视图会把所有的服务都导入进去,而我们实际上并不需要这么多视图，

而且默认的视图模版只能匹配filebeat-\*开头的索引，所以这里我们有2个需要需要解决：

1.通过一定处理只导入我们需要的模版

2.导入的视图模版索引名称可以自定义

解决方法：

1.备份一份filebeat的kibana视图，删除不需要的视图模版文件

2.修改视图文件里默认的索引名称为我们需要的索引名称

cp -a /usr/share/filebeat/kibana /root

find . -type f ! -name "\*nginx\*"|xargs rm -rf

sed -i 's#filebeat\-\\*#nginx\-\\*#g' Filebeat-nginx-overview.json

替换索引名称

filebeat setup --dashboards -E setup.dashboards.directory=/root/kibana/

## 使用模块收集mysql日志和慢日志

## 使用模块收集mongo日志和redis日志

# 使用缓存收集日志

当日志的数量非常多的时候，可能需要引入缓存层作为临时存储数据的地方，防止因为ES处理不过来导致日志丢失的情况。

filebeat支持将日志发送到redis或者kafka作为消息队列缓存。

但是使用了缓存层，就不能使用模版来配置日志收集了。

所以最好日志是json格式

## 使用单台redis作为缓存

这里需要说明，如果使用redis作为缓存

可以将不同的日志类型单独写成一个键，这样好处是清晰，但是缺点是logstash写起来起来复杂

也可以将所有的日志全部写入到一个键中，然后靠后端的logstash去过滤处理。

### 安装启动测试redis

[root@elk-175 ~]# yum install redis -y

[root@elk-175 ~]# systemctl status redis

[root@elk-175 ~]# redis-cli

127.0.0.1:6379> set k1 v1

OK

127.0.0.1:6379> GEt k1

"v1"

127.0.0.1:6379>

### 配置nginx的json日志

将nginx的日志调整为json格式

log\_format json '{ "time\_local": "$time\_local", '

'"remote\_addr": "$remote\_addr", '

'"referer": "$http\_referer", '

'"request": "$request", '

'"status": $status, '

'"bytes": $body\_bytes\_sent, '

'"agent": "$http\_user\_agent", '

'"x\_forwarded": "$http\_x\_forwarded\_for", '

'"up\_addr": "$upstream\_addr",'

'"up\_host": "$upstream\_http\_host",'

'"upstream\_time": "$upstream\_response\_time",'

'"request\_time": "$request\_time"' ' }';

### 配置filebeat写入到不同的key中

[root@elk-175 ~]# cat /etc/filebeat/filebeat.yml

filebeat.inputs:

- type: log

enabled: true

paths:

- /var/log/nginx/access.log

json.keys\_under\_root: true

json.overwrite\_keys: true

tags: ["access"]

- type: log

enabled: true

paths:

- /var/log/nginx/error.log

tags: ["error"]

setup.template.settings:

index.number\_of\_shards: 3

setup.kibana:

host: "192.168.47.175:5601"

output.redis:

hosts: ["localhost"]

keys:

- key: "nginx\_access"

when.contains:

tags: "access"

- key: "nginx\_error"

when.contains:

tags: "error"

### 配置logstash读取不同的key

[root@elk-175 ~]# cat /etc/logstash/conf.d/redis.conf

input {

redis {

host => "127.0.0.1"

port => "6379"

db => "0"

key => "nginx\_access"

data\_type => "list"

}

redis {

host => "127.0.0.1"

port => "6379"

db => "0"

key => "nginx\_error"

data\_type => "list"

}

}

filter {

mutate {

convert => ["upstream\_time", "float"]

convert => ["request\_time", "float"]

}

}

output {

stdout {}

if "access" in [tags] {

elasticsearch {

hosts => "http://localhost:9200"

manage\_template => false

index => "nginx\_access-%{+yyyy.MM.dd}"

}

}

if "error" in [tags] {

elasticsearch {

hosts => "http://localhost:9200"

manage\_template => false

index => "nginx\_error-%{+yyyy.MM.dd}"

}

}

}

### filebeat收集日志写入到一个key中

[root@elk-175 ~]# cat /etc/filebeat/filebeat.yml

filebeat.inputs:

- type: log

enabled: true

paths:

- /var/log/nginx/access.log

json.keys\_under\_root: true

json.overwrite\_keys: true

tags: ["access"]

- type: log

enabled: true

paths:

- /var/log/nginx/error.log

tags: ["error"]

setup.template.settings:

index.number\_of\_shards: 3

setup.kibana:

host: "192.168.47.175:5601"

output.redis:

hosts: ["localhost"]

key: "filebeat"

### logstash根据tag区分一个key里的不同日志

[root@elk-175 ~]# cat /etc/logstash/conf.d/redis.conf

input {

redis {

host => "127.0.0.1"

port => "6379"

db => "0"

key => "filebeat"

data\_type => "list"

}

}

filter {

mutate {

convert => ["upstream\_time", "float"]

convert => ["request\_time", "float"]

}

}

output {

if "access" in [tags] {

elasticsearch {

hosts => "http://localhost:9200"

manage\_template => false

index => "nginx\_access-%{+yyyy.MM.dd}"

}

}

if "error" in [tags] {

elasticsearch {

hosts => "http://localhost:9200"

manage\_template => false

index => "nginx\_error-%{+yyyy.MM.dd}"

}

}

}

## 使用nginx+keepalived代理多台redis

### 解决方案介绍

上面我们实验了单台redis作为收集日志的缓存层，但是单台redis存在一个问题，就是单点故障

虽然可以做持久化处理，但是加入服务器坏掉或者修复时间未知的情况下还是有可能会丢数据。

redis集群方案有哨兵和集群，但可惜的是filebeat和logstash都不支持这两种方案。

不过利用我们目前所学的知识完全可以解决这个问题，虽然不是彻底解决，但是已经很好了

方案如下：

1.使用nginx+keepalived反向代理负载均衡到后面的多台redis

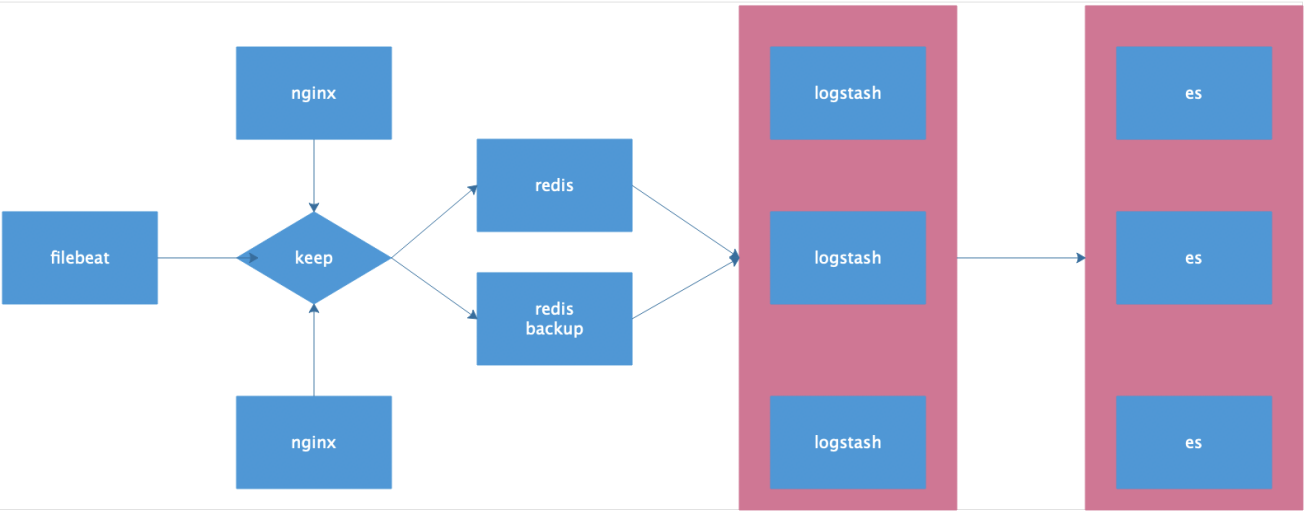
2.考虑到redis故障切换中数据一致性的问题，所以最好我们只使用2台redis,并且只工作一台，另外一台作为backup，只有第一台坏掉后，第二台才会工作。

3.filebeat的oputut的redis地址为keepalived的虚拟IP

4.logstash可以启动多个节点来加速读取redis的数据

5.后端可以采用多台es集群来做支撑

拓扑图如下：



### nginx配置文件

注意：添加stream模块，要在nginx.conf里最后添加，而不是在conf.d里面添加子配置

[root@lb02 ~]# cat /etc/nginx/nginx.conf

..........................

stream {

upstream redis {

server 10.0.0.51:6381 max\_fails=2 fail\_timeout=10s;

server 10.0.0.51:6382 max\_fails=2 fail\_timeout=10s backup;

}

server {

listen 6379;

proxy\_connect\_timeout 1s;

proxy\_timeout 3s;

proxy\_pass redis;

}

}

### redis配置

[root@db01 /opt/redis\_6381/conf]# cat /opt/redis\_6381/conf/redis\_6381.conf

bind 10.0.0.51

port 6381

daemonize yes

pidfile /opt/redis\_6381/pid/redis\_6381.pid

logfile /opt/redis\_6381/logs/redis\_6381.log

databases 16

save 900 1

save 300 10

save 60 10000

stop-writes-on-bgsave-error yes

rdbcompression yes

rdbchecksum yes

dbfilename redis\_6381.rdb

dir /data/redis\_6381

### 模拟故障测试

### logstash配置

### 使用supervisor批量管理多个logstash进程

## 使用kafka作为缓存

### 介绍

### zookeeper安装配置

### kafka安装配置

### kafka集群收发消息测试

### filebeat配置

[root@kafka-175 conf.d]# cat /etc/filebeat/filebeat.yml

filebeat.inputs:

- type: log

enabled: true

paths:

- /var/log/nginx/access.log

json.keys\_under\_root: true

json.overwrite\_keys: true

tags: ["access"]

- type: log

enabled: true

paths:

- /var/log/nginx/error.log

tags: ["error"]

setup.template.settings:

index.number\_of\_shards: 3

setup.kibana:

host: "192.168.47.175:5601"

output.kafka:

hosts: ["192.168.47.175:9092","192.168.47.176:9092","192.168.47.177:9092"]

topic: elklog

### logstash配置

[root@kafka-175 conf.d]# cat /etc/logstash/conf.d/kafka.conf

input{

kafka{

bootstrap\_servers=>"192.168.47.175:9092"

topics=>["elklog"]

group\_id=>"logstash"

codec => "json"

}

}

filter {

mutate {

convert => ["upstream\_time", "float"]

convert => ["request\_time", "float"]

}

}

output {

if "access" in [tags] {

elasticsearch {

hosts => "http://localhost:9200"

manage\_template => false

index => "nginx\_access-%{+yyyy.MM.dd}"

}

}

if "error" in [tags] {

elasticsearch {

hosts => "http://localhost:9200"

manage\_template => false

index => "nginx\_error-%{+yyyy.MM.dd}"

}

}

}

# 配置logstash同步mysql数据

使用jdbc插件可以同步mysql的数据到logstash里

# kibana画图展示

## kibana页面介绍

## kibana创建图标

## kibana创建面板视图

# grafana画图展示

除了kibana外，grafana也支持从es调取数据并展示

https://grafana.com/docs/features/datasources/elasticsearch/#using-elasticsearch-in-grafana

# 如何在公司推广ELK项目

如何在公司推广ELK的项目

1.优先表达出对其他人带来的便利和好处

- 领导

- 开发

- 测试

- DBA

-- 以后你自己相查什么数据都可以直接查

-- 以前插个日志运维得查10分钟，现在你只要点几下鼠标就ok了

-- 我专门为你们部门定制了一个图形面板，你看看效果，有啥不满意的尽管说，我优先给你们部门解决

2.对于运维来说，json日志更友好

- 告诉领导，json格式更友好，做出测试环境，并给领导演示json格式优点

- 提前统计好到底需要收集哪些日志

- 要求能转成json的就转成json

- 自己改不了的，给领导说，让开发改成json格式

- 那行，领导你看，如果不改成json，就只能是这个效果了，后续的新需求，可能就满足不了了

- 并不是说所有日志，必须是json

3.推广演示

- 提前都准备好文档和ppt还有测试环境

- 一定要对比传统分析日志和ELK之后带来的巨大改变

# 面试问题