2、中间件

2.1 概述

日志的生命周期一般分为采集，传输，存储和分析四个环节，每个步骤有可用的中间件和工具。

选用中间件时所关注的角度：性能、可靠性、插件支持程度、配置复杂度

2.1.1 汇总

名称	优势	劣势	常用
Logstash	插件多，资料充足，配置简单	性能一般，不支持缓存	采集，传输
Filebeat	可靠，性能好，多种对接	存在注册表过大问题	采集
Fluentd	基于Ruby，插件多，易于编写和维护	灵活性相对差，性能一般	采集
Logtail	资源占用少	尽做收集，类型解析较弱	采集
Logagent	轻量快速，支持缓存	灵活性差	采集，传输
rsyslog	速度快，轻量化，规则灵活	配置繁杂，文档差，版本差异	采集
Syslog-ng	性能好，文档相对齐全	开源版本功能有限	采集
Kafka	可靠，稳定，高性能	容易重复消费	传输
Flume	可靠，容错性高，易管理，实时	配置繁琐	传输
ElasticSearch	分布式部署，自动故障转移	个人开发，还不够自动	存储
HDFS	高容错，大数据存储	延迟高，不适于小文件	存储
kibana	多种图标配置简单	仅支持es	展示，分析
grafana	多数据源支持，自带警报	偏监控方向	分析，预警

2.1.2 总结

- 企业实际实战中，elk是成熟且广泛使用的方案。
- logstash因为性能弱于filebeat，并不直接运用于采集起点，一般使用filebeat。

- 进入elk前，经验性角度，放置kafka，一方面作为队列和缓冲，另一方面提供了统一的入口渠道。

2.2 部署

2.2.1 ES

1）简介

> 许多年前，一个刚结婚的名叫 Shay Banon 的失业开发者，跟着他的妻子去了伦敦，他的妻子在那里学习厨师。 为了给他的妻子做一个食谱搜索引擎，他开始使用 Lucene 的一个早期版本。直接使用 Lucene 是很难的，因此 Shay 开始做一个抽象层，Java 开发者使用它可以很简单的给他们的程序添加搜索功能。 他发布了他的第一个开源项目 Compass。
>
> 后来 Shay 获得了一份工作，主要是高性能，分布式环境下的内存数据网格。这个对于高性能，实时，分布式搜索引擎的需求尤为突出， 他决定重写 Compass，把它变为一个独立的服务并取名 Elasticsearch。
>
> 第一个公开版本在2010年2月发布，从此以后，Elasticsearch 已经成为了 Github 上最活跃的项目之一，他拥有超过300名 contributors(目前736名 contributors)。 一家公司已经开始围绕 Elasticsearch 提供商业服务，并开发新的特性，但是，Elasticsearch 将永远开源并对所有人可用。
>
> 据说，Shay 的妻子还在等着她的食谱搜索引擎… 0_0!

Elasticsearch 是一个开源的搜索引擎，建立在一个全文搜索引擎库Lucene基础之上。 Lucene 可以说是当下最先进、高性能、全功能的搜索引擎库，缺点是Lucene的使用非常的复杂。Elasticsearch 也是使用 Java 编写的，它的内部使用 Lucene 做索引与搜索，但是它的目的是使全文检索变得简单，通过隐藏 Lucene 的复杂性，取而代之的提供一套简单一致的 RESTful API。

2）相关链接

官网：

https://www.elastic.co/cn/downloads/elasticsearch

下载：

wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.5.1-linux-x86_64.tar.gz

分词器：

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

3）部署

系统环境：以下课程假设测试机器ip为172.17.0.203

解压：

tar xvf elasticsearch-7.5.1-linux-x86_64.tar.gz

修改：

config/elasticsearch.yml

```
#主机名，通过 hostname 命令查询到
cluster.initial_master_nodes: ["主机名"]

network.host: 0.0.0.0
http.port: 9200
http.cors.enabled: true
http.cors.allow-origin: "*"
```

修改文件描述符：

vim /etc/sysctl.conf

```
vm.max_map_count=64000
sysctl -p
```

启动：

```
#es不允许root用户启动，需要添加新用户身份

#创建elsearch用户组及elsearch用户

groupadd elsearch

useradd elsearch -g elsearch -p elasticsearch

#更改elasticsearch文件夹及内部文件的所属用户及组为elsearch:elsearch

chown -R elsearch:elsearch elasticsearch

#切换到elsearch用户再启动

su elsearch

#守护进程运行
./bin/elasticsearch -d

#验证启动进程
ps aux | grep elasticsearch
```

4）验证：

访问 http://172.17.0.203:9200 ， 启动成功

```
{
  "name" : "bj-yjy-java-wsw",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "LT1y6fQSRPykQv3BUdolZg",
  "version" : {
    "number" : "7.5.1",
    "build_flavor" : "default",
    "build_type" : "tar",
    "build_hash" : "3ae9ac9a93c95bd0cdc054951cf95d88e1e18d96",
    "build_date" : "2019-12-16T22:57:37.835892Z",
    "build_snapshot" : false,
    "lucene_version" : "8.3.0",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

5）中文分词器安装：

es默认分词器对中文分词非常不友好，需要安装ik分词器

Analyzer: ik_smart , ik_max_word , Tokenizer: ik_smart , ik_max_word

```
wget -c https://github.com/medcl/elasticsearch-analysis-ik/releases/download/v7.5.1/elasticsearch-analysis-ik-7.5.1.zip

mkdir $ES_HOME/plugins/ik

unzip elasticsearch-analysis-ik-7.5.1.zip -d $ES_HOME/plugins/ik

#安装完成重启es
```

6）验证分词器：

POST: http://172.17.0.203:9200/_analyze

BODY: {"text":"测试分词效果","analyzer": "ik_smart"}

指令：curl http://localhost:9200/_analyze -X POST -H 'Content-Type:application/json' -d '{"text":"test elasticsearch 测试分词效果","analyzer": "ik_smart"}'

RESULT:

```
{
```

```
  "tokens": [
    {
      "token": "test",
      "start_offset": 0,
      "end_offset": 4,
      "type": "ENGLISH",
      "position": 0
    },
    {
      "token": "elasticsearch",
      "start_offset": 5,
      "end_offset": 18,
      "type": "ENGLISH",
      "position": 1
    },
    {
      "token": "测试",
      "start_offset": 19,
      "end_offset": 21,
      "type": "CN_WORD",
      "position": 2
    },
    {
      "token": "分词",
      "start_offset": 21,
      "end_offset": 23,
      "type": "CN_WORD",
      "position": 3
    },
    {
      "token": "效果",
      "start_offset": 23,
      "end_offset": 25,
      "type": "CN_WORD",
      "position": 4
    }
  ]
}
```


2.2.2 es-head

1）简介

我们可以方便的使用curl等客户端工具，通过Restful API对Elasticsearch进行操作，但也有一些客户端工具提供对于ElasticSearch更加友好的可视化操作支持，elasticsearch-head就是其中很优秀的代表。

早期版本的elasticsearch-head可以直接以插件的方式在Elasticsearch中进行安装，在Elasticsearch 5之后则需要将elasticsearch-head服务单独运行，并且支持Chrome的插件方式或者Docker容器运行方式。

2）部署

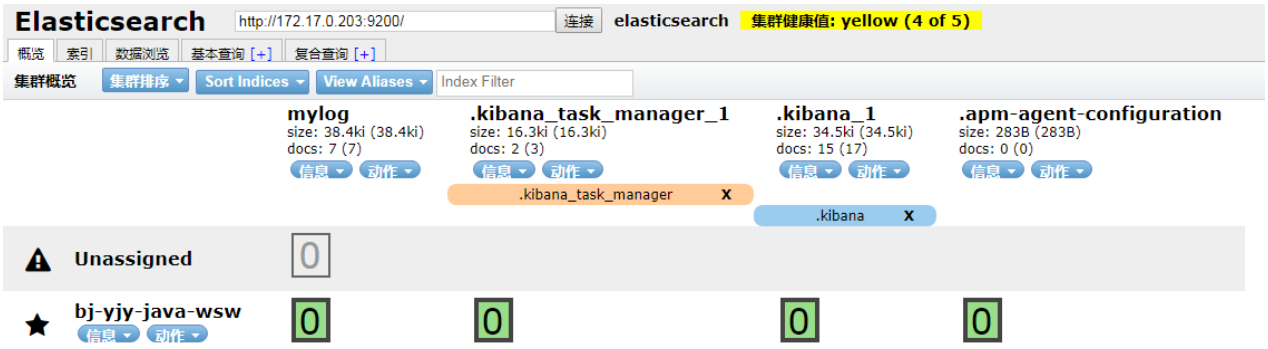
```
#采用docker启动
#查看镜像
docker images | grep elasticsearch-head

#下载镜像
docker pull alivv/elasticsearch-head

#启动
docker run -d --name eshead -p 9100:9100 alivv/elasticsearch-head
```

3）验证

访问9100端口，并连接es地址：



4）功能演示

- 概览：健康状态、集群信息、索引信息、分区信息
- 索引：索引概览、新建索引
- 数据浏览：索引清单、字段清单与筛选
- 数据查询：基本查询、复杂查询、查询基本语法

2.2.3 logstash

1）简介

Logstash诞生于2009年8有2日，其作者是世界著名的虚拟主机托管商DreamHost的运维工程师Jordan Sissel。在2013年，被ElasticSearch公司收购，作为日志收集工具，成为elk的一员。

2）相关链接

项目主页：

https://www.elastic.co/cn/downloads/logstash

下载地址：

wget https://artifacts.elastic.co/downloads/logstash/logstash-7.5.1.tar.gz

3）部署

解压：

tar xvf logstash-7.5.1.tar.gz

配置：

在conf.d目录下新建一个config1.conf文件

```
input {
  file {
    path => "/root/logs/*.log"
    start_position => beginning
    add_field => {"from" => "localfile"}
  }
}
filter {

}
output {
  elasticsearch {
    hosts => "localhost:9200"
    index => "mylog"
  }
  stdout {
  }

}
```

启动，允许配置文件自动刷新：

```
nohup sh /opt/app/elk/logstash-7.5.1/bin/logstash -f /opt/app/elk/logstash-7.5.1/conf.d/ --config.reload.automatic >> /opt/logs/logstash.log &
```

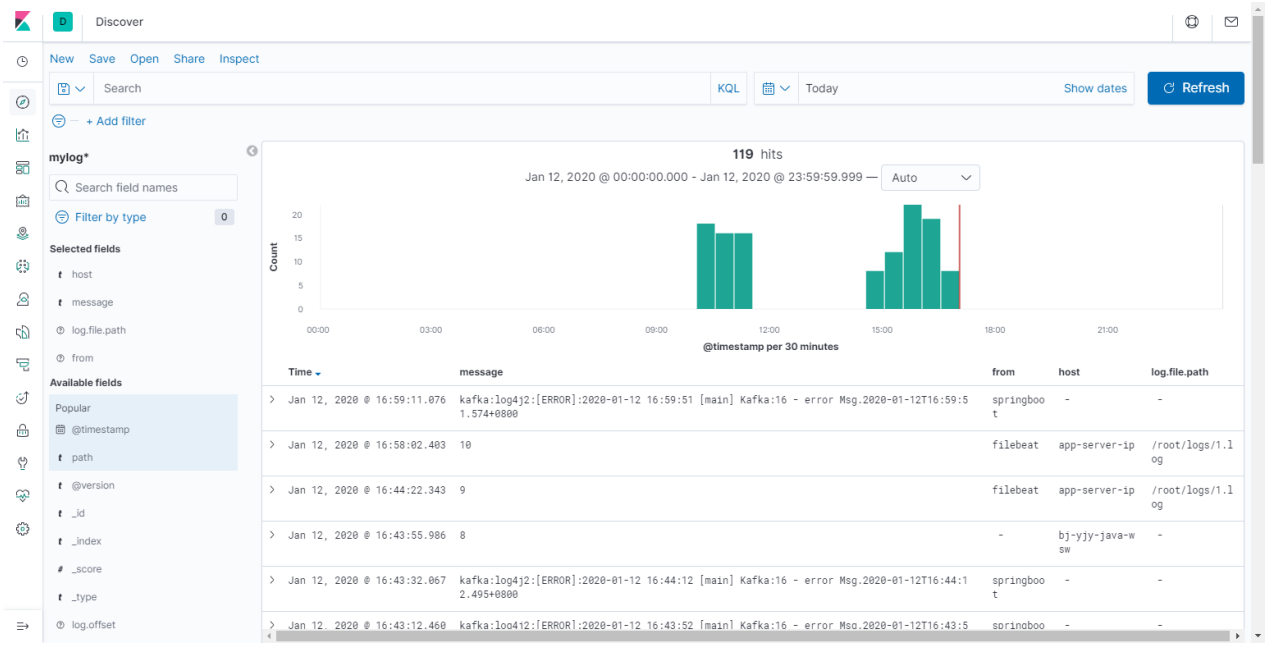
4）验证：

```
#生成一条测试日志数据，从es-head验证是否正常采集
mkdir -p /root/logs/
date >> /root/logs/1.log
```

2.2.4 kibana

1）简介

Kibana是一个开源的分析与可视化平台，主要用于和Elasticsearch一起使用。使用kibana进行简单的配置，就可以搜索、查看存放在Elasticsearch中的数据。Kibana具有各种不同的图表、表格、地图等，借助这些直观的视图，可以达到高级的数据分析与可视化的目的。



2）相关链接

https://www.elastic.co/cn/downloads/kibana

https://artifacts.elastic.co/downloads/kibana/kibana-7.5.1-linux-x86_64.tar.gz

3）安装部署

解压：

tar xvf kibana-7.5.1-linux-x86_64.tar.gz

配置：

config/kibana.yml

```
server.port: 9102
server.host: "0.0.0.0"
elasticsearch.hosts: "http://localhost:9200"
kibana.index: ".kibana"
```

启动：

```
#kibana默认不允许root用户启动，可以加--allow-root选项

nohup sh /opt/app/elk/kibana-7.5.1-linux-x86_64/bin/kibana  --allow-root  >
/opt/logs/kibana.log &
```

4）验证

启动成功，数据为空，向es提交一条数据，通过es-head查询，并通过logstash查询类比展示

下面logstash搭建完成后，会再次展示通过采集进入的数据

5）功能

- 索引配置
- 日志配置
- 检索
- 图表

2.2.5 kafka

1）简介

Kafka是最初由Linkedin公司开发，是一个分布式、分区的、多副本的、多订阅者，基于zookeeper协调的分布式日志系统（也可以当做MQ系统），常见可以用于web/nginx日志、访问日志，消息服务等等，Linkedin于2010年贡献给了Apache基金会并成为顶级开源项目。常用于日志处理场景。

2）资源

http://kafka.apache.org/downloads

3）部署

```
#docker启动
#启动zookeeper
docker run --name zookeeper \
  -v /opt/data/zksingle:/data \
```

```
  -p 2181:2181 \
  -e ZOO_LOG4J_PROP="INFO,ROLLINGFILE"  \
  -d zookeeper:3.4.13

#启动kafka

docker run -d --name kafka \
        -p 9103:9092 \
        --link zookeeper:zookeeper \
        --env KAFKA_BROKER_ID=100 \
        --env HOST_IP=39.98.133.153 \
        --env KAFKA_ZOOKEEPER_CONNECT=zookeeper:2181 \
        --env KAFKA_ADVERTISED_HOST_NAME=39.98.133.153 \
        --env KAFKA_ADVERTISED_PORT=9103 \
        --restart=always \
        --volume /etc/localtime:/etc/localtime \
        wurstmeister/kafka:2.12-2.2.2
```

4）验证

```
#使用zk节点数据验证启动情况
docker exec -it zookeeper sh
#进入zookeeper后查看节点信息
ls /brokers

#进入容器
docker exec -it kafka sh
/opt/kafka_2.12-2.2.2/bin

#客户端监听（该步会自动创建topic）
./kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic demo --
from-beginning

#另起一个终端，验证发送
./kafka-console-producer.sh --broker-list localhost:9092 --topic demo
```

5）实例操作

修改logstash从kafka获取，从控制台重新写入一条日志，进kibana查看数据展示情况

```
input {
  file {
    path => "/root/logs/*.log"
```

```
    start_position => beginning
    add_field => {"from" => "localfile"}
  }
  kafka {
    bootstrap_servers => ["39.98.133.153:9103"]
    group_id => "logstash"
    topics => ["demo"]
    consumer_threads => 1
    decorate_events => true
    add_field => {"from" => "demo"}
  }

}
filter {

}
output {
  elasticsearch {
    hosts => "localhost:9200"
    index => "mylog"
  }
  stdout {
  }

}
```

2.2.6 kafka-manager

1）简介

kafka-manager是目前最受欢迎的kafka集群管理工具，最早由雅虎开源，用户可以在Web界面执行一些简单的集群管理操作。具体支持以下内容：

- 管理多个集群
- 轻松检查群集状态（主题，消费者，偏移，代理，副本分发，分区分发）
- 运行首选副本选举
- 使用选项生成分区分配以选择要使用的代理
- 运行分区重新分配（基于生成的分配）
- 使用可选主题配置创建主题（0.8.1.1具有与0.8.2+不同的配置）
- 删除主题（仅支持0.8.2+并记住在代理配置中设置delete.topic.enable = true）
- 主题列表现在指示标记为删除的主题（仅支持0.8.2+）
- 批量生成多个主题的分区分配，并可选择要使用的代理
- 批量运行重新分配多个主题的分区
- 将分区添加到现有主题
- 更新现有主题的配置

2）资源

https://github.com/yahoo/kafka-manager/releases

docker库里的版本太陈旧，需要从官网下载源码包，编译成二进制包。具体编译过程参考项目主页下面的Deployment章节。

部署中直接使用打好包的kafka-manager-2.0.0.2.zip，manage版本为2.0.0.2，配置kafka cluster最高支持2.2.0 ，实际验证，可以操作kafka 2.2.2

3）部署

```
#解压
unzip kafka-manager-2.0.0.2.zip

#配置文件，修改目录下的conf/application.conf
kafka-manager.zkhosts="localhost:2181"

#启动，指定端口9104
km_home=./kafka-manager-2.0.0.2
nohup $km_home/bin/kafka-manager -Dconfig.file=$km_home/conf/application.conf -
Dhttp.port=9104 > /opt/logs/kibana.log &
tail -f /opt/logs/kibana.log
```

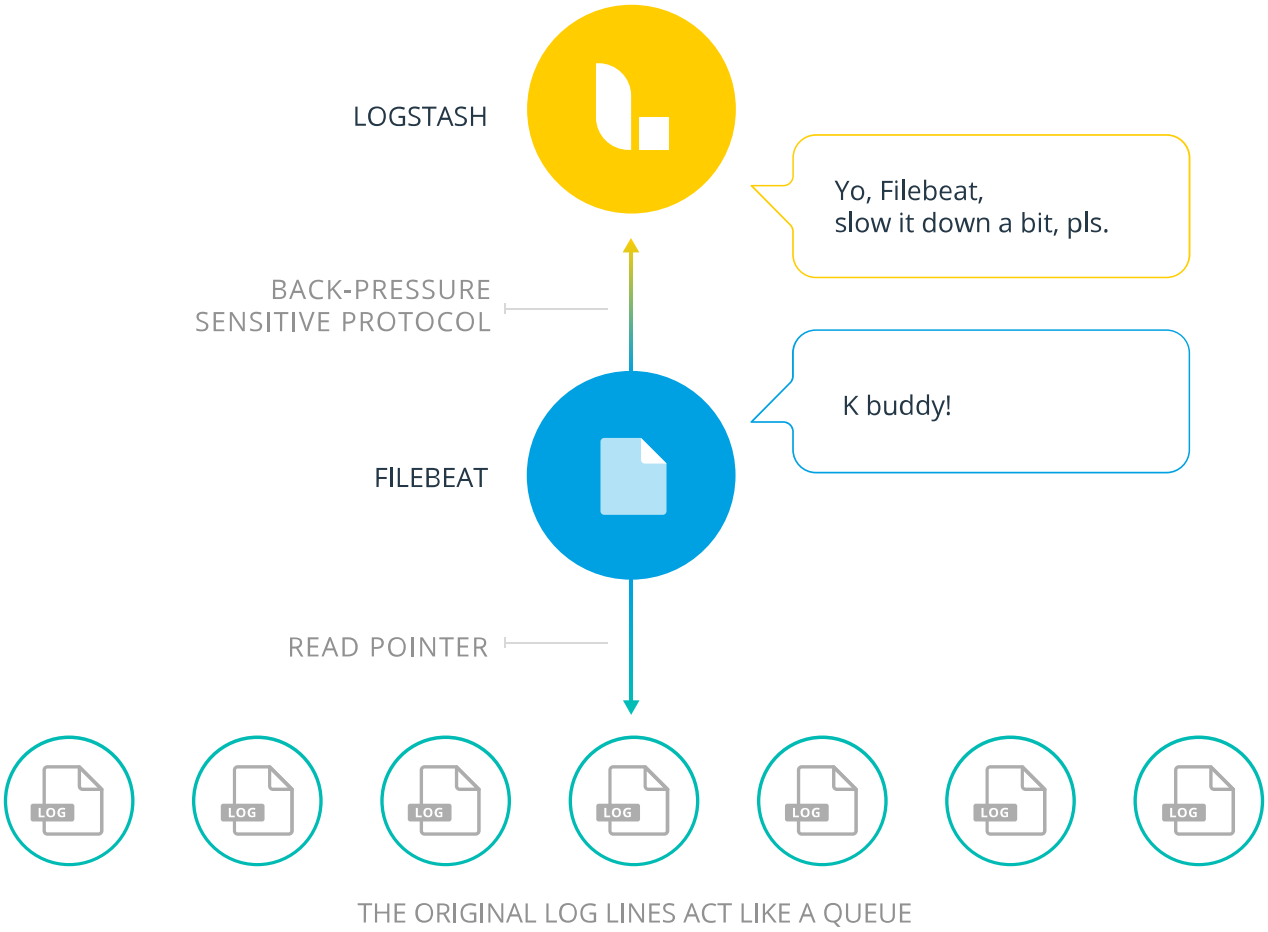
4）功能说明

- cluster创建与管理
- Brokers信息查看
- topic创建与管理
- pattern及replica配置

2.2.7 filebeat

1）简介

Filebeat是一个轻量级日志传输Agent，可以将指定日志转发到Logstash、Elasticsearch、Kafka、Redis等中。Filebeat占用资源少，而且安装配置也比较简单，支持目前各类主流OS及Docker平台。



2）资源

主页：

https://www.elastic.co/cn/products/beats/filebeat

下载：

wget https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.5.1-linux-x86_64.tar.gz

3）部署

解压：tar zxvf filebeat-7.5.1-linux-x86_64.tar.gz

使用kafka-manager创建一个filebeat队列

配置filebeat.yml：

```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /root/logs/*.log
  fields:
    from: filebeat
```

```
output.kafka:
   enabled: true
   hosts: ["39.98.133.153:9103"]
   topic: filebeat
   compression: gzip

processors:
- drop_fields:
   fields: ["beat", "input", "source",
"offset","metadata","timestamp","agent","ecs","fields"]


filebeat.config.modules:
  path: ${path.config}/modules.d/*.yml
  reload.enabled: true

logging.level: info
name: filebeat-server-ip
```

启动：

```
cd /opt/app/elk/filebeat-7.5.1-linux-x86_64
nohup ./filebeat -e -c filebeat.yml >> /opt/logs/filebeat.log &
tail -f /opt/logs/filebeat.log
```

4）验证

修改logstash，去掉file采集，接收来自filebeat队列的消息。

请注意！这里有一个问题：filebeat默认读取字符后，输出的是json格式，上述codec即让logstash解析json，但是仍会报错。原因是filebeat里的host.name属性，需要加入filter，合并属性名字：

```
input {
  kafka {
    bootstrap_servers => ["39.98.133.153:9103"]
    group_id => "logstash"
    topics => ["filebeat"]
    consumer_threads => 1
    decorate_events => true
    add_field => {"from" => "filebeat"}
    codec => "json"
  }
}
...

filter {
  mutate {
    rename => { "[host][name]" => "host" }
  }
```

```
}
```

重新录入一条日志信息仅log文件，查看logstash stdout日志，查看kibana是否正常采集入es

附：filebeat输出json格式参考范本

```
{
        "@timestamp": "2019-05-11T07:55:02.127Z",
        "@metadata": {
                "beat": "filebeat",
                "type": "_doc",
                "version": "7.5.1",
                "topic": "app.log"
        },
        "ecs": {
                "version": "1.0.0"
        },
        "log": {
                "offset": 2661796,
                "file": {
                        "path": "/var/log/app.log"
                }
        },
        "message": "05-11 00:10:19.851[DEBUG][http-nio-39545-exec-9] ",
        "fields": {
                "log_topic": "app.log"
        },
        "host": {
                "name": "172.33.12.109"
        },
        "agent": {
                "id": "6a86e9d9-e1e8-4b32-b027-f1c936f66e4f",
                "version": "7.0.1",
                "name": "172.33.12.109",
                "type": "filebeat",
                "ephemeral_id": "8326a240-e9de-44f4-b24d-a1c8d2654e19",
                "hostname": "client-ali"
        }
}
```

附：简单文本处理命令参考

```
#列出根下的目录
ls -l

lrwxrwxrwx.   1 root root         7 Dec 17 09:20 bin -> usr/bin
dr-xr-xr-x.   5 root root      4096 Dec 24 18:33 boot
```

```
drwxr-xr-x   21 root root      3140 Dec 31 10:39 dev
drwxr-xr-x.  84 root root      8192 Jan  3 10:09 etc
drwxr-xr-x.   3 root root        22 Jan  3 09:38 home
lrwxrwxrwx.   1 root root         7 Dec 17 09:20 lib -> usr/lib
lrwxrwxrwx.   1 root root         9 Dec 17 09:20 lib64 -> usr/lib64
...

#使用grep过滤lib开头，并且不要带64的
ll | grep lib | grep -v 64

lrwxrwxrwx.   1 root root         7 Dec 17 09:20 lib -> usr/lib

#使用awk摘取目录名称
ll | grep lib | grep -v 64 | awk '{print $9}'

lib

#awk可以自定分隔符，用-F
echo '123#456'
echo '123#456' | awk -F '#' '{print $2}'
```