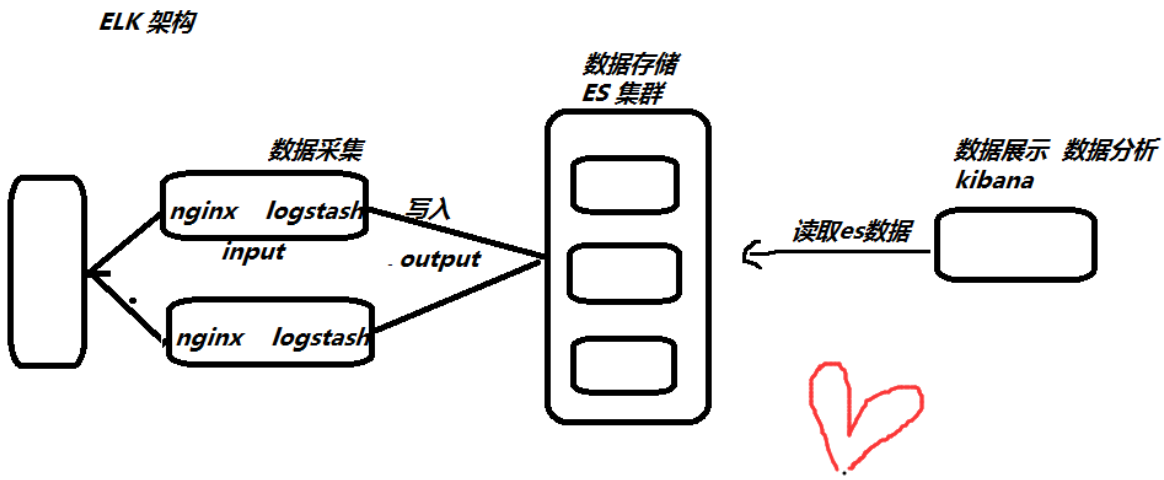
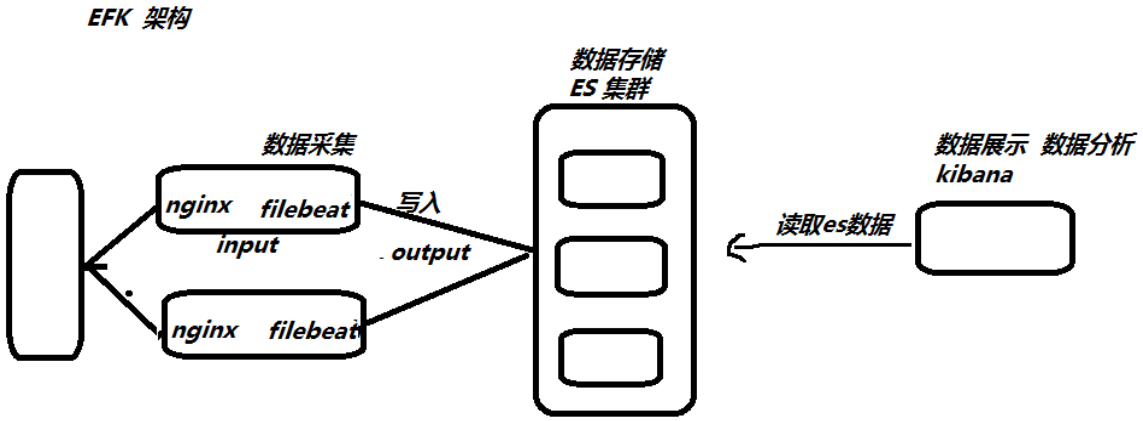
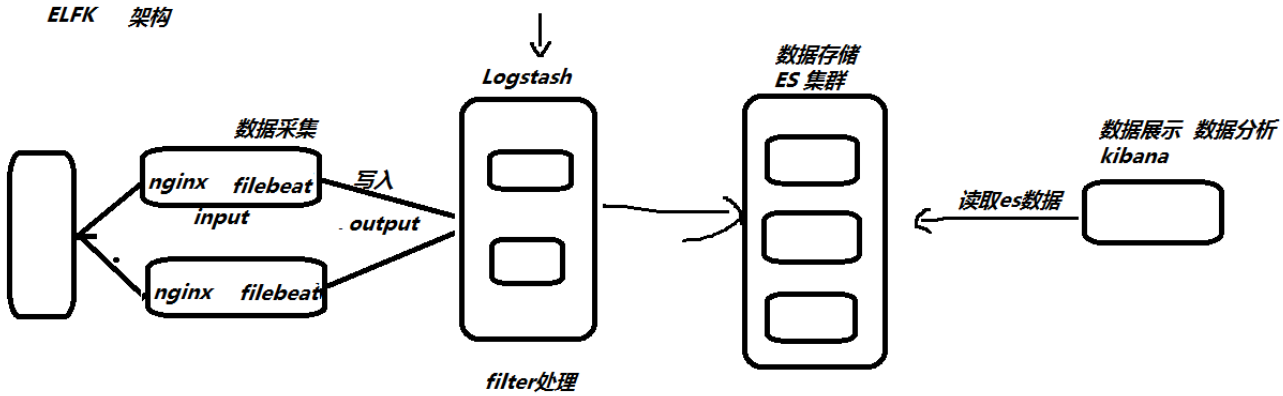
ELK-day01

ELK基础架构



E: elastcisearch 数据搜索 数据存储 java L: Logstash 数据收集 (数据解析 数据转换) 数据输出 java F: Filebeat 数据采集 (简单的数据处理) <--go K: Kibana 数据分析 数据展示





5.使用EFK收集哪些日志?

容器：docker

代理： Haproxy、Nginx

web：Nginx、Tomcat、Httpd、PHP

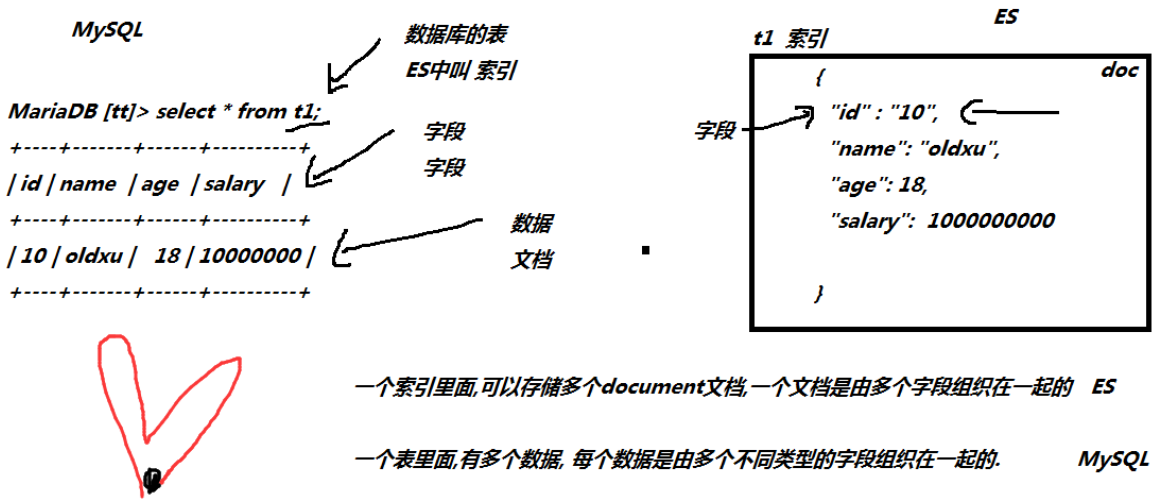
db：mysql、redis、mongo、elasticsearch

存储：nfs、glusterfs、fastdfs

系统：message、security

业务：app

1.ElasticSearch基本使用



ES7

10.0.0.7 172.16.1.7 nginx+filebeat

10.0.0.8 172.16.1.8 nginx+filebeat

10.0.0.141 172.16.1.141 kafka

10.0.0.141 172.16.1.141 kafka

10.0.0.141 172.16.1.141 kafka

10.0.0.151 172.16.1.151 Logstash

10.0.0.152 172.16.1.152 Logstash

10.0.0.161 172.16.1.161 es-node1 ES Kibana

10.0.0.162 172.16.1.162 es-node2

10.0.0.163 172.16.1.163 es-node3

2.ES单机安装:

```
[root@es-node1 ~]# yum install java -y
[root@es-node1 ~]# rpm -ivh elasticsearch-7.4.0-x86_64.rpm  kibana-7.4.0-x86_64.rpm

[root@es-node1 ~]# vim /etc/elasticsearch/jvm.options
-Xms512m    #实验环境  生产环境最少内存一半以上    官方建议 最高32Gb
-Xmx512m

[root@es-node1 ~]# systemctl enable elasticsearch.service
[root@es-node1 ~]# systemctl start elasticsearch.service

#测试es是否启动
[root@es-node1 ~]# curl 127.0.0.1:9200
```

```
{
  "name" : "es-node1",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "l8qVNdklSvSqjUJSvUh4dw",
  "version" : {
    "number" : "7.4.0",
    "build_flavor" : "default",
    "build_type" : "rpm",
    "build_hash" :
"22e1767283e61a198cb4db791ea66e3f11ab9910",
    "build_date" : "2019-09-27T08:36:48.569419Z",
    "build_snapshot" : false,
    "lucene_version" : "8.2.0",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-
beta1"
  },
  "tagline" : "You Know, for Search"
}

#修改kibana的配置
[root@es-node1 ~]# vim /etc/kibana/kibana.yml
server.host: "0.0.0.0"
i18n.locale: "zh-CN"

#启动kibana
[root@es-node1 ~]# systemctl enable kibana
[root@es-node1 ~]# systemctl start kibana


#访问kibana

```

3.ES索引基本操作

```
# 创建一个索引
PUT /oldxu_es

# 查看所有的索引
GET _cat/indices

# 删除索引
DELETE /oldxu_es
```

```
# 创建一个索引
PUT /oldxu_es

# 查看所有的索引
GET _cat/indices

# 删除索引
DELETE /oldxu_es


#给oldxu_es索引录入一个文档
POST /tt/_doc/1
{
  "name": "oldxu",
  "age": 18,
  "salary": 1000000000
}

POST /oldxu_es/_doc/2
```

```
{
  "name": "oldguo",
  "age": 35,
  "salary": 100
}

#获取指定的id数据
GET /oldxu_es/_doc/1

#获取所有的文档  默认前10个
GET /oldxu_es/_search

#模糊查询
GET /oldxu_es/_search
{
  "query": {
    "term": {
      "name": "oldxu"
    }
  }
}

#删除指定id的文档
DELETE /oldxu_es/_doc/1



#
POST _bulk
{"index":{"_index":"tt","_id":"1"}}
{"name":"oldxu","age":"18"}
{"create":{"_index":"tt","_id":"2"}}
{"name":"oldqiang","age":"30"}
{"delete":{"_index":"tt","_id":"2"}}
```

```
{"update":{"_id":"1","_index":"tt"}}
{"doc":{"age":"20"}}


#一次查询多个文档
GET _mget
{
  "docs": [
    {
      "_index": "tt",
      "_id": "1"
    },
    {
      "_index": "tt",
      "_id": "2"
    }
  ]
}
```

4.ES集群环境搭建

1配置集群

```
#删除所有的es相关的数据（集群无法组件的情况）
[root@es-node1 ~]# rm -rf /var/lib/elasticsearch/*
[root@es-node1 ~]# systemctl stop elasticsearch.service
[root@es-node1 ~]# systemctl stop kibana

配置node1
[root@es-node1 ~]# grep '^[a-z]'
/etc/elasticsearch/elasticsearch.yml
cluster.name: my-oldxu
```

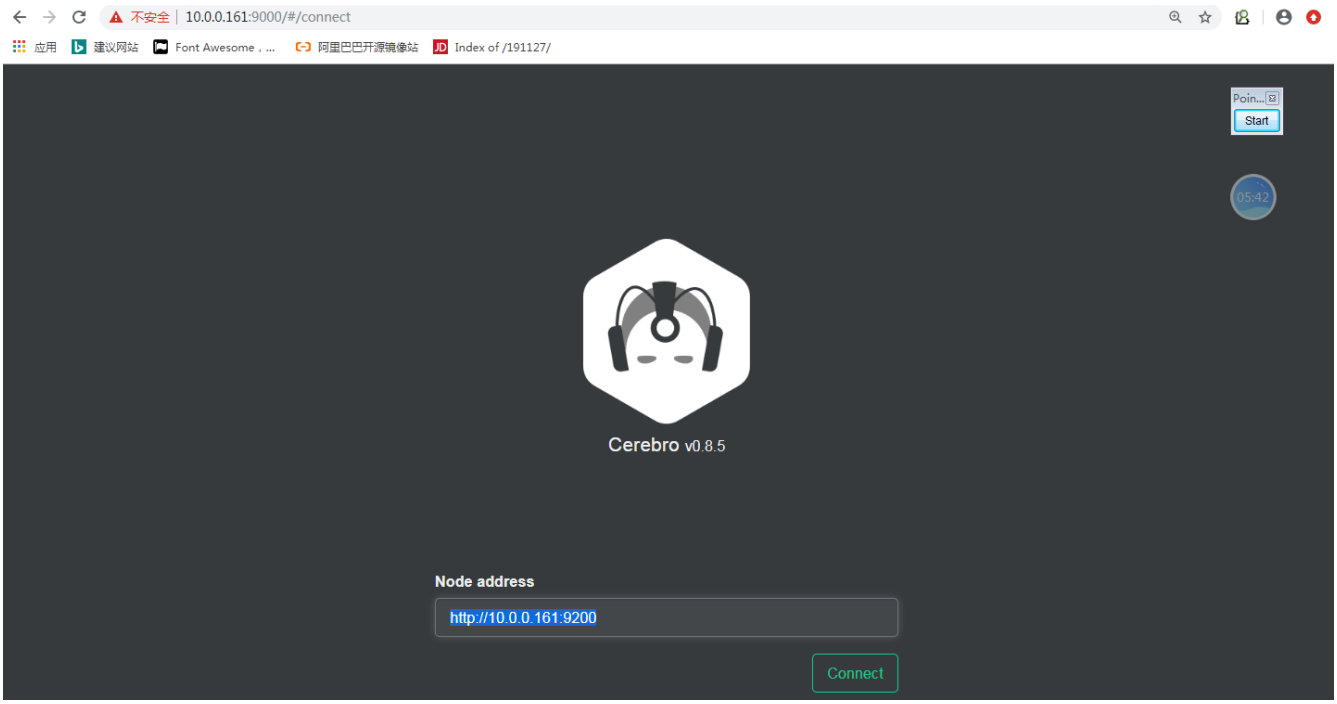
```
node.name: node1
path.data: /var/lib/elasticsearch
path.logs: /var/log/elasticsearch
network.host: 0.0.0.0
http.port: 9200
discovery.seed_hosts: ["10.0.0.161", "10.0.0.162", "10.0.0.163"]
cluster.initial_master_nodes: ["10.0.0.161", "10.0.0.162", "10.0.0.163"]

scp -rp /etc/elasticsearch/elasticsearch.yml root@172.16.1.162:/etc/elasticsearch/elasticsearch.yml
 scp -rp /etc/elasticsearch/elasticsearch.yml root@172.16.1.163:/etc/elasticsearch/elasticsearch.yml

scp  /etc/elasticsearch/jvm.options root@172.16.1.162:/etc/elasticsearch/jvm.options
scp  /etc/elasticsearch/jvm.options root@172.16.1.163:/etc/elasticsearch/jvm.options


配置node2
[root@es-node2 ~]# grep  "^[a-z]" /etc/elasticsearch/elasticsearch.yml
cluster.name: my-oldxu
node.name: node2
path.data: /var/lib/elasticsearch
path.logs: /var/log/elasticsearch
network.host: 0.0.0.0
http.port: 9200
discovery.seed_hosts: ["10.0.0.161", "10.0.0.162", "10.0.0.163"]
cluster.initial_master_nodes: ["10.0.0.161", "10.0.0.162", "10.0.0.163"]
```

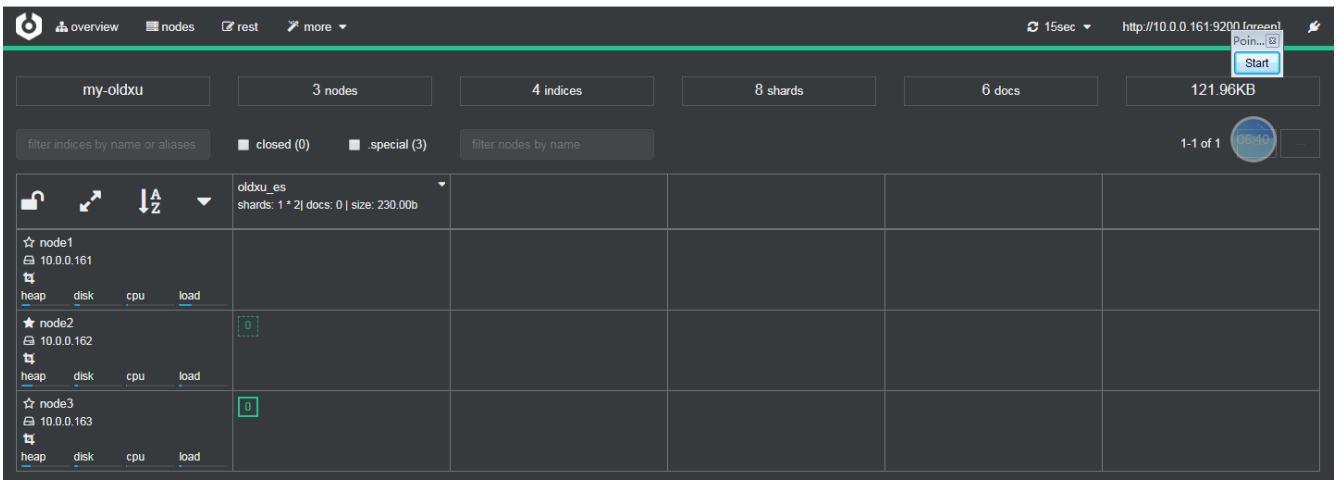
```
34
35
36 配置node3
37 [root@es-node3 ~]# grep "^[a-z]"
   /etc/elasticsearch/elasticsearch.yml
38 cluster.name: my-oldxu
39 node.name: node3
40 path.data: /var/lib/elasticsearch
41 path.logs: /var/log/elasticsearch
42 network.host: 0.0.0.0
43 http.port: 9200
44 discovery.seed_hosts: ["10.0.0.161", "10.0.0.162",
   "10.0.0.163"]
45 cluster.initial_master_nodes: ["10.0.0.161",
   "10.0.0.162", "10.0.0.163"]
46
47
48 启动所有节点
49 systemctl start elasticsearch
50
51 通过 curl  检查集群环境是否正常
52 curl http://10.0.0.163:9200/_cluster/health?pretty
53
54
55 kibana
56     GET /_cluster_health
57
58
```

5.cerebro状态检查

cerebro插件来检查整个集群的环境 默认监听9000端口

```
[root@es-node1 ~]# rpm -ivh cerebro-0.8.5-1.noarch.rpm
[root@es-node1 ~]# vim /etc/cerebro/application.conf
data.path = "/tmp/cerebro.db"
[root@es-node1 ~]# systemctl enable cerebro
```





6.集群节点

master角色: 负责控制整个集群的操作, 通过cluter_status状态维护集群.

选举: cluster.initial_master_nodes `master-eligible`

可以不参与选举: node.master: false cluster_state: 节点信息 索引信息

data角色: 存储数据 (默认都是data节点) 关闭data: node.data: false

coordinating角色 : 负责路由 不能取消

7.ES集群状态检查

5.ES集群健康检查

Cluster Health 获取集群的健康状态，整个集群状态包括以下三种:

1）green 健康状态，指所有主副分片都正常分配

2）yellow 指所有主分片都正常分配，但是有副本分片未正常分配

3）red 有主分片未分配，表示索引不完备，写也可能有问题。（但不代表不能存储数据和读取数据）

4）可以通过 `GET _cluster/health?pretty=true` *方式获取集群状态*

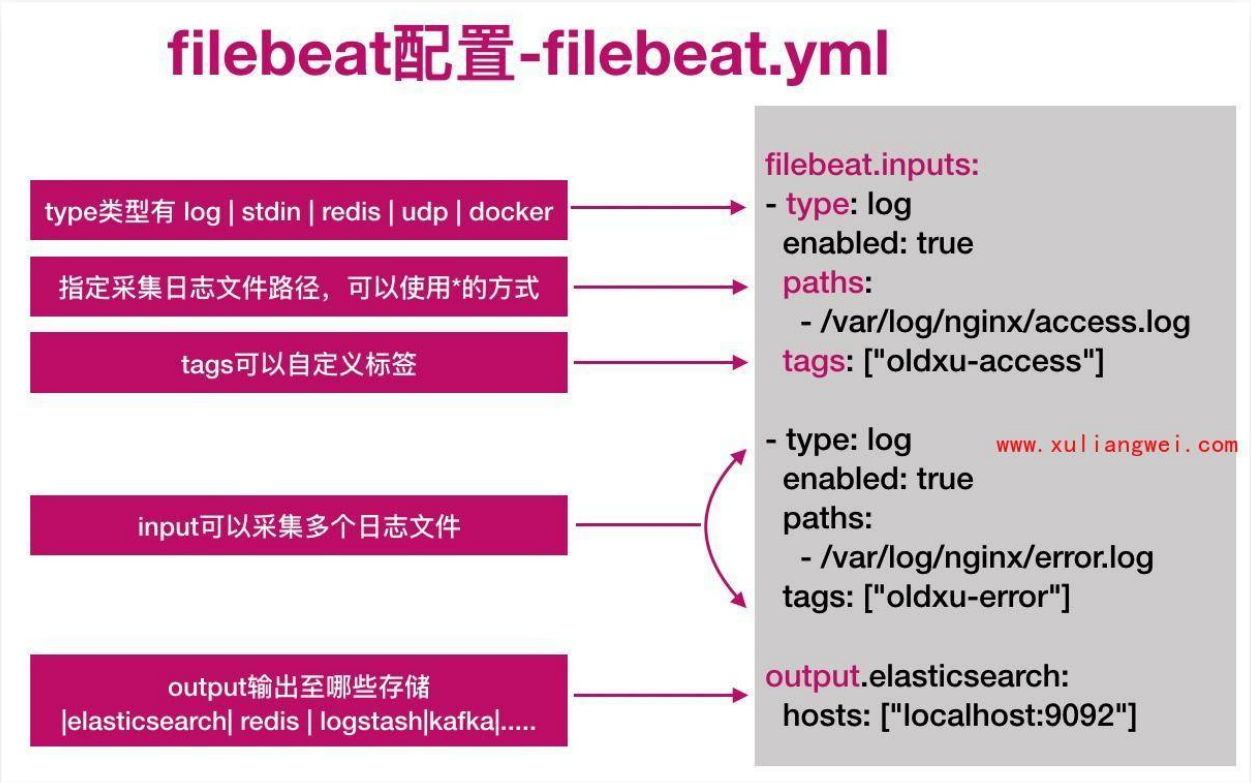
10 % 3 = 0 分片 10 % 3 = 1 分片 10 % 3 = 2 分片

```
shard = hash(routing) % number_of_primary_shards
# hash                                算法保证将数据均匀分散在分片中
# routing                             是一个关键参数，默认是文档id，也可以自定义。
# number_of_primary_shards   主分片数

# 注意：该算法与主分片数相关，一但确定后便不能更改主分片。
# 因为一旦修改主分片修改后，Share的计算就完全不一样了。
```

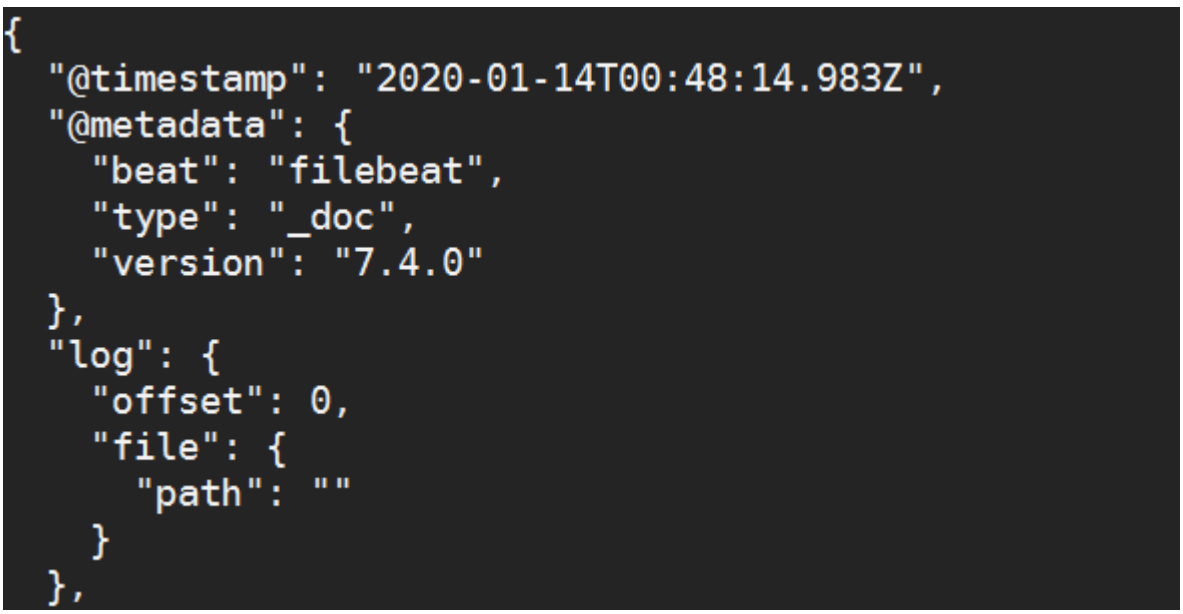
8.ES集群扩展

ELK-day02

input 我们要采集的日志文件路径, 收割机 *harvester* 监听文件的变化 --> splooer程序 --> 转发 es | logstash | kafka | redis



```
filebeat.inputs:
  - type: stdin          #标准输入
    enabled: true        #启用

output.console:          #标准输出
  pretty: true
  enable: true

```




```
#将文件最新发生变化的内容,存入ES

[root@web01 ~]# cat /etc/filebeat/file.yml
filebeat.inputs:
  - type: log
    paths: /var/log/nginx/access.log
    enabled: true

output.elasticsearch:
  hosts:
["10.0.0.161:9200","10.0.0.162:9200","10.0.0.163:9200"]
```

收集系统日志

特别分散--> syslog --> file.txt

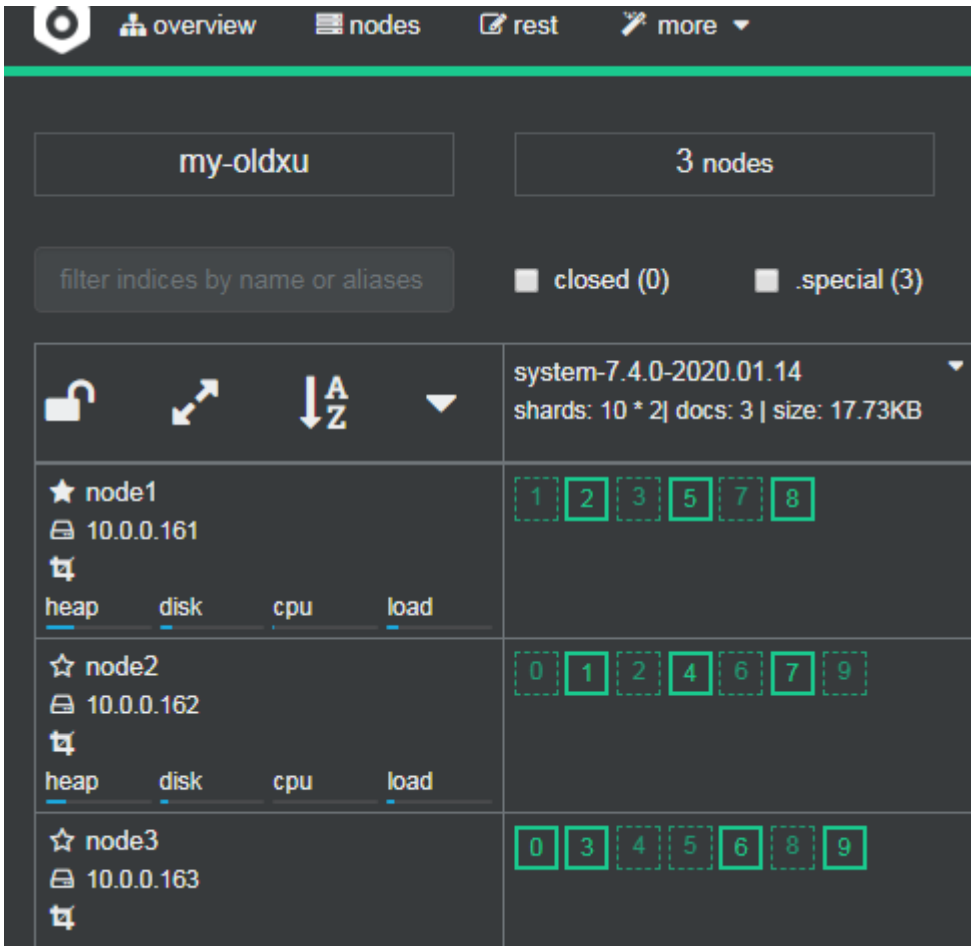
1.减少无用的数据

2.调整索引名称

3.测试调整模板,设定分片

```
[root@web01 ~]# cat /etc/filebeat/filebeat.yml
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/log/oldxu.log
  include_lines: ['^ERR', '^WARN', 'sshd']   #只看指定的日志

output.elasticsearch:
  hosts: ["10.0.0.161:9200","10.0.0.162:9200","10.0.0.163:9200"]
  index: "system-%{[agent.version]}-%{+yyyy.MM.dd}"

setup.ilm.enabled: false
setup.template.name: system   #索引关联的模板名称
setup.template.pattern: system-*



方式一:
```

```
###设定system模板的分片数和副本数
#setup.template.settings:                    #定义索引分片数和副本
#  index.number_of_shards: 3
#  index.number_of_replicas: 1

方式二:

   "number_of_routing_shards": "30",
   "number_of_shards": "10",
   "number_of_replicas": "1",

    1.修改system模板    --->  添加 shards 分片数数量,replicas的数量
    2.删除模板关联的索引
    3.删除filebeat自行指定的分片数和副本数
    4.重启filebeat
    5.产生新的日志
```



收集Nginx

```
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"'
    '}';

    access_log /var/log/nginx/access.log json;

```

配置filebeat

```
[root@web01 filebeat]# cat /etc/filebeat/filebeat.yml
```

```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/log/nginx/access.log
  json.keys_under_root: true    #默认Flase，还会将json解析的日志存储至messages字段
  json.overwrite_keys: true     #覆盖默认的key，使用自定义json格式的key

output.elasticsearch:
  hosts: ["10.0.0.161:9200","10.0.0.162:9200","10.0.0.163:9200"]
  index: "nginx-%{[agent.version]}-%{+yyyy.MM.dd}"

setup.ilm.enabled: false
setup.template.name: nginx   #索引关联的模板名称
setup.template.pattern: nginx-*
```

收集nginx访问日志和错误日志

```
[root@web01 filebeat]# cat filebeat.yml
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/log/nginx/access.log
  json.keys_under_root: true    #默认Flase，还会将json解析的日志存储至messages字段
  json.overwrite_keys: true     #覆盖默认的key，使用自定义json格式的key
  tags: ["access"]


```

```
- type: log
  enabled: true
  paths:
    - /var/log/nginx/error.log
  tags: ["error"]



output.elasticsearch:
  hosts:
["10.0.0.161:9200","10.0.0.162:9200","10.0.0.163:9200"]
  indices:
    - index: "nginx-access-%{[agent.version]}-%
{+yyyy.MM.dd}"
      when.contains:
        tags: "access"

    - index: "nginx-error-%{[agent.version]}-%
{+yyyy.MM.dd}"
      when.contains:
        tags: "error"


setup.ilm.enabled: false
setup.template.name: nginx   #索引关联的模板名称
setup.template.pattern: nginx-*
```

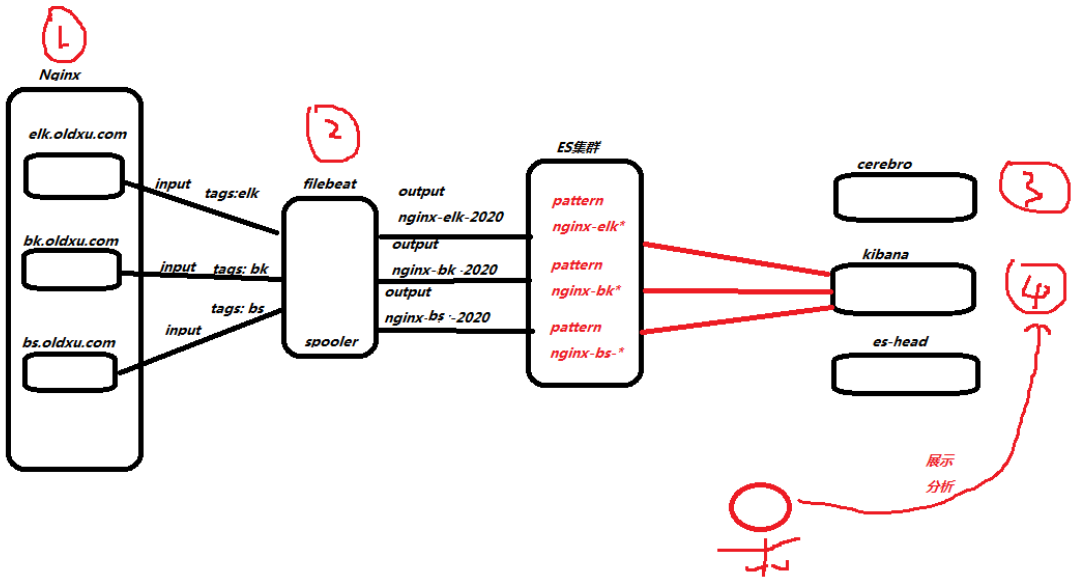
收集nginx多个虚拟主机的日志

elk.oldxu.com

bk.oldxu.com

bs.oldxu.com

error日志



1.虚拟主机

```
[root@web01 conf.d]# cat elk.oldxu.com.conf
server {
    listen 80;
    server_name elk.oldxu.com;
    root /code/elk;
    access_log /var/log/nginx/elk.oldxu.com.log json;

    location / {
        index index.html;
    }
}


[root@web01 conf.d]# cat bs.oldxu.com.conf
server {
    listen 80;
    server_name bs.oldxu.com;
```

```
    root /code/bs;
    access_log /var/log/nginx/bs.oldxu.com.log json;

    location / {
        index index.html;
    }
}

[root@web01 conf.d]# cat bk.oldxu.com.conf
server {
    listen 80;
    server_name bk.oldxu.com;
    root /code/bk;
    access_log /var/log/nginx/bk.oldxu.com.log json;

    location / {
        index index.html;
    }
}
```

2.测试,模拟产生日志

```
[root@web01 conf.d]# curl -H Host:elk.oldxu.com http://10.0.0.7
elk.oldux.com
[root@web01 conf.d]# curl -H Host:bs.oldxu.com http://10.0.0.7
bs.oldux.com
[root@web01 conf.d]# curl -H Host:bk.oldxu.com http://10.0.0.7
bk.oldux.com

```

3.配置filebeat

```
[root@web01 filebeat]# cat /etc/filebeat/filebeat.yml
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/log/nginx/elk.oldxu.com.log
  json.keys_under_root: true
  json.overwrite_keys: true
  tags: ["nginx-elk-host"]

- type: log
  enabled: true
  paths:
    - /var/log/nginx/bs.oldxu.com.log
  json.keys_under_root: true
  json.overwrite_keys: true
  tags: ["nginx-bs-host"]

- type: log
  enabled: true
  paths:
    - /var/log/nginx/bk.oldxu.com.log
  json.keys_under_root: true
  json.overwrite_keys: true
  tags: ["nginx-bk-host"]


- type: log
  enabled: true
  paths:
    - /var/log/nginx/error.log
  tags: ["nginx-error"]


```

```
output.elasticsearch:
  hosts: ["10.0.0.161:9200","10.0.0.162:9200","10.0.0.163:9200"]
  indices:
    - index: "nginx-elk-access-%{[agent.version]}-%{+yyyy.MM.dd}"
      when.contains:
        tags: "nginx-elk-host"

    - index: "nginx-bs-access-%{[agent.version]}-%{+yyyy.MM.dd}"
      when.contains:
        tags: "nginx-bs-host"

    - index: "nginx-bk-access-%{[agent.version]}-%{+yyyy.MM.dd}"
      when.contains:
        tags: "nginx-bk-host"

    - index: "nginx-error-%{[agent.version]}-%{+yyyy.MM.dd}"
      when.contains:
        tags: "nginx-error"

setup.ilm.enabled: false
setup.template.name: nginx   #索引关联的模板名称
setup.template.pattern: nginx-*
```

Tomcat日志

访问日志 ---> json格式

```
#1.修改tomcat日志格式

[root@web02 soft]# yum install java -y
[root@web02 soft]# vim tomcat/conf/server.xml
      <Host name="tomcat.oldxu.com"  appBase="webapps"
            unpackWARs="true" autoDeploy="true">
        <Valve
className="org.apache.catalina.valves.AccessLogValve"
directory="logs"
               prefix="tomcat.oldxu.com.log"
suffix=".txt"
                pattern="
{&amp;quot;clientip&amp;quot;:&amp;quot;%h&amp;quot;,&amp;quot;ClientUser&amp;quot;:&amp;quot;%l&amp;quot;,&amp;quot;authenticated&amp;quot;:&amp;quot;%u&amp;quot;,&amp;quot;AccessTime&amp;quot;:&amp;quot;%t&amp;quot;,&amp;quot;method&amp;quot;:&amp;quot;%r&amp;quot;,&amp;quot;status&amp;quot;:&amp;quot;%s&amp;quot;,&amp;quot;SendBytes&amp;quot;:&amp;quot;%b&amp;quot;,&amp;quot;Query?
string&amp;quot;:&amp;quot;%q&amp;quot;,&amp;quot;partner&amp;quot;:&amp;quot;%
{Referer}i&amp;quot;,&amp;quot;AgentVersion&amp;quot;:&amp;quot;%{User-
Agent}i&amp;quot;}" />
      </Host>
```

配置filebeat

```
[root@web02 filebeat]# cat /etc/filebeat/filebeat.yml
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /soft/tomcat/logs/tomcat.oldxu.com.log.*.txt
  json.keys_under_root: true    #默认Flase，还会将json解析
的日志存储至messages字段
  json.overwrite_keys: true     #覆盖默认的key，使用自定义
json格式的key
```

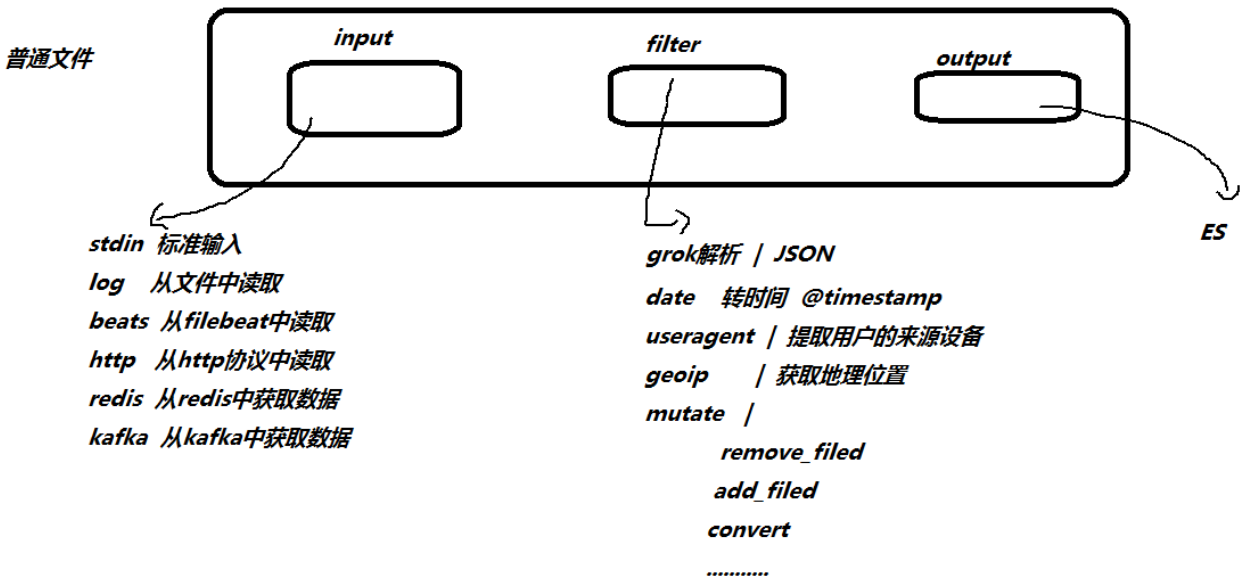
```
9
10
11 output.elasticsearch:
12   hosts: ["10.0.0.161:9200","10.0.0.162:9200"]
13   index: "tomcat-access-%{[agent.version]}-%
   {+yyyy.MM.dd}"
14
15 setup.ilm.enabled: false
16 setup.template.name: tomcat    #索引关联的模板名称
17 setup.template.pattern: tomcat-*
18
```

错误日志 <--java

```
1  [root@web02 filebeat]# cat filebeat.yml
2  filebeat.inputs:
3  - type: log
4    enabled: true
5    paths:
6      - /soft/tomcat/logs/tomcat.oldxu.com.log.*.txt
7    json.keys_under_root: true     #默认Flase，还会将json解析
   的日志存储至messages字段
8    json.overwrite_keys: true      #覆盖默认的key，使用自定义
   json格式的key
9    tags: ["tomcat-access"]
10
11 - type: log
12   enabled: true
13   paths:
14     - /soft/tomcat/logs/catalina.out
15   multiline.pattern: '^\d{2}'    #匹配以2个数字开头的
16   multiline.negate: true
17   multiline.match: after
```

```
  multiline.max_lines: 10000    #默认最大合并行为500，可根据实际情况调整。
  tags: ["tomcat-error"]


output.elasticsearch:
  hosts: ["10.0.0.161:9200","10.0.0.162:9200"]
  indices:
    - index: "tomcat-access-%{[agent.version]}-%{+yyyy.MM.dd}"
      when.contains:
        tags: "tomcat-access"

    - index: "tomcat-error-%{[agent.version]}-%{+yyyy.MM.dd}"
      when.contains:
        tags: "tomcat-error"


setup.ilm.enabled: false
setup.template.name: tomcat   #索引关联的模板名称
setup.template.pattern: tomcat-*
```

ELK-day03



Logstash Input

```
1
2  [root@logstash-node1 conf.d]# cat
   input_file_output_console.conf
3  input {
4      file {
5          path => "/var/log/oldxu.log"
6          type => syslog
7          exclude => "*.gz"          #不想监听的文件规则，基于
   glob匹配语法
8          start_position => "beginning"    #第一次从头开始读
   取文件 beginning or end
9          stat_interval => "3"     #定时检查文件是否更新，默认
   1s
10     }
11 }
12
13 output {
14     stdout {
15         codec => rubydebug
```

```
    }
}



[root@logstash-node1 conf.d]# cat
input_stdin_output_console.conf
input {
    stdin {
        type => stdin
        tags => "tags_stdin"
    }

}
output {
    stdout {
        codec => "rubydebug"
    }
}
```

logstash 分析Nginx

```
66.249.73.135 - - [20/May/2015:21:05:11 +0000] "GET
/blog/tags/xsendevent HTTP/1.1" 200 10049 "-"
"Mozilla/5.0 (iPhone; CPU iPhone OS 6_0 like Mac OS X)
AppleWebKit/536.26 (KHTML, like Gecko) Version/6.0
Mobile/10A5376e Safari/8536.25 (compatible;
Googlebot/2.1; +http://www.google.com/bot.html)"
```

```
[root@logstash-node1 conf.d]# cat
input_filebeat_output_es.conf
input {
```

```
    beats {
        port => 5044
    }
}

filter {

if "nginx-access" in [tags][0] {
    grok {
        match => { "message" => "%{IPORHOST:clientip} %{USER:ident} %{USER:auth} \[%{HTTPDATE:timestamp}\] \"(?:%{WORD:verb} %{NOTSPACE:request}(?: HTTP/%{NUMBER:httpversion})?|%{DATA:rawrequest})\" %{NUMBER:response} (?:%{NUMBER:bytes}|-) %{QS:referrer} %{QS:useragent}" }
    }

    date {
        match => ["timestamp", "dd/MMM/yyyy:HH:mm:ss Z"]
        target => "@timestamp"
        timezone => "Asia/Shanghai"
    }

    geoip {
        source => "clientip"
    }

    useragent {
        source => "useragent"
        target => "useragent"
    }

    mutate {
```

```
        rename => ["%{[host][name]}" , "hostname" ]
        convert => [ "bytes", "integer" ]
        remove_field => [ "message", "agent" , "input","ecs" ]
        add_field => { "target_index" => "logstash-nginx-access-%{+YYYY.MM.dd}" }
    }
}  else if "nginx-error" in [tags][0] {
    mutate {
        add_field => { "target_index" => "logstash-nginx-error-%{+YYYY.MM.dd}" }
    }
   }

}

output {
    elasticsearch {
        hosts => ["10.0.0.161:9200","10.0.0.162:9200","10.0.0.163:9200"]
        index => "%{[target_index]}"
    }
}

```

Logstash 分析 MySQL

filebeat

```
[root@web01 filebeat]# cat filebeat.yml
filebeat.inputs:
```

```
- type: log
  enabled: true
  paths:
    - /var/log/mariadb/slow.log
  exclude_lines: ['^\# Time']
  multiline.pattern: '^\# User'
  multiline.negate: true
  multiline.match: after
  multiline.max_lines: 10000
  tags: ["mysql-slow"]

output.logstash:
  hosts: ["10.0.0.151:5044"]
```

logstash

```
[root@logstash-node1 conf.d]# cat
input_filebeat_mysql_output_es.conf
input {
    beats {
        port => 5044
    }
}


filter {

    mutate {
        gsub => ["message","\n"," "]
    }
    grok {

        match => {
```

```
        "message" => "(?m)^# User@Host: %{USER:User}\[%{USER-2:User}\] @ (?:(?<Clienthost>\S*) )?\[(?:%{IP:Client_IP})?\] # Thread_id: %{NUMBER:Thread_id:integer}\s+ Schema: (?:(?<DBname>\S*) )\s+QC_hit: (?:(?<QC_hit>\S*) )# Query_time: %{NUMBER:Query_Time}\s+ Lock_time: %{NUMBER:Lock_Time}\s+ Rows_sent: %{NUMBER:Rows_Sent:integer}\s+Rows_examined: %{NUMBER:Rows_Examined:integer} SET timestamp=%{NUMBER:timestamp}; \s*(?<Query>(?<Action>\w+)\s+.*)"
        }
    }

    date {
        match => ["timestamp","UNIX", "YYYY-MM-dd HH:mm:ss"]
        target => "@timestamp"
        timezone => "Asia/Shanghai"
    }
    mutate {
        remove_field => ["message","input","timestamp","agent","ecs","log"]
        convert => ["Lock_Time","float"]
        convert => ["Query_Time","float"]
        add_field => { "target_index" => "logstash-mysql-slow-%{+YYYY.MM.dd}" }
    }
}

output {
    elasticsearch {
        hosts => ["10.0.0.161:9200"]
        index => "%{[target_index]}"
    }
```

```
39      stdout {
40          codec => "rubydebug"
41      }
42  }
```

Logstash 分析 APP

模拟产生日志

```
java -jar app-dashboard-1.0-SNAPSHOT.jar
&>/var/log/app.log
```

filebeat

```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/log/app.log

output.logstash:
  hosts: ["10.0.0.151:5044"]
```

logstash

```
[root@logstash-node1 conf.d]# cat
input_filebeat_app_output_es.conf
input {
    beats {
        port => 5044
    }
}

filter {
```

```
    mutate {
        split => {"message" => "|"}
        add_field => {
            "UserID" => "%{[message][1]}"
            "Action" => "%{[message][2]}"
            "Date" => "%{[message][3]}"
        }

                convert => {
                        "UserID" => "integer"
                        "Action" => "string"
                        "Date"   => "string"
                }
    }

    #2020-01-15 17:04:15
    date {
        match => ["Date","yyyy-MM-dd HH:mm:ss"]
        target => "@timestamp"
        timezone => "Asia/Chongqing"
    }

    mutate {
        #remove_field => ["message","Date"]
        add_field => { "target_index" => "logstash-app-
%{+YYYY.MM.dd}" }
    }
}

output {
    elasticsearch {
        hosts => ["10.0.0.161:9200"]
        index => "%{[target_index]}"
        template_overwrite => true
```

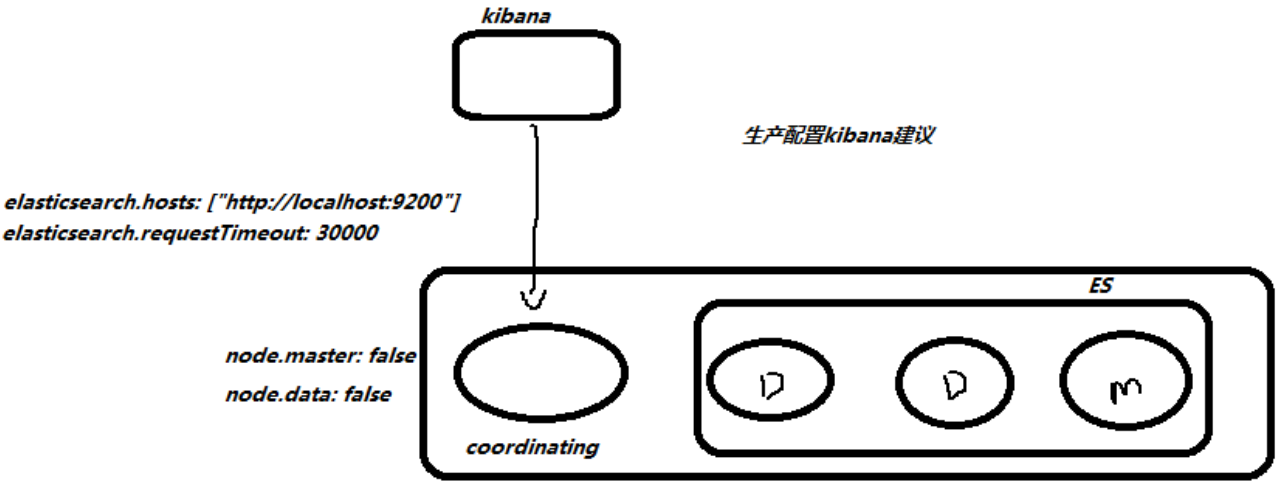
```
    }
}
```

ELK-day04

```
----------------if "access" in [tags][0]            else if
input
    beats  --> redis|kafka(解耦|缓存)  --> logstash (数据解析|数据转换)  --> ES
    http
    file
    redis
    kafka
filter
    grok   <--一推正则表达式的别名  | 将非结构化数据转为结构化数据
    useragent    mac | linux  | windows   小米  iphone
    geoip       城市分布 | 经纬度
    date        时间转换  时区
    mutate
        add_field
        remove_field
        rename_field
        split
        gsub
        convert
output
    ES
```

kibana

kibana用来做数据的展示和数据的分析,读取ES的索引获取对应的数据



- 统计网站总PV、IP
- 统计网站每天PV、IP
- 统计访问IP Top10
- 统计来源的refrer
- 统计前10的资源 标签云
- 统计访问状态码
- 统计客户端设备
- 统计大于1s请求 待会出
- 统计网站总流量
- 统计网站访问流量趋势图

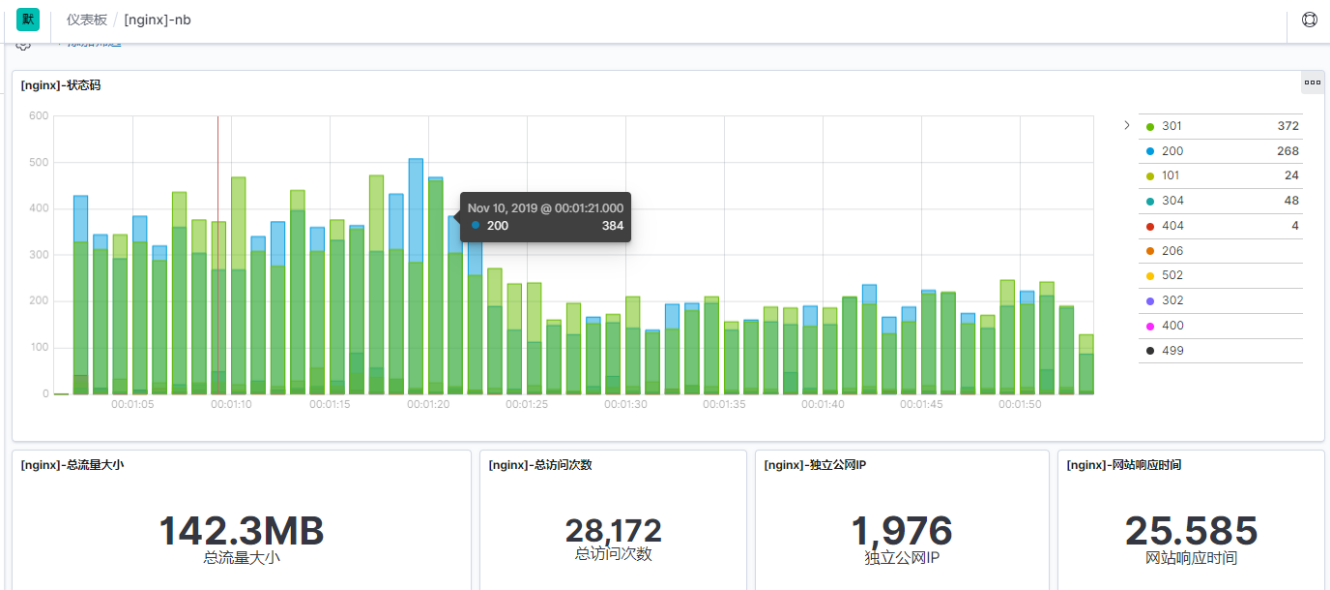
```
[root@web01 bk]# cat /etc/filebeat/filebeat.yml
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/log/nginx/access.log
  tags: ["access"]
```

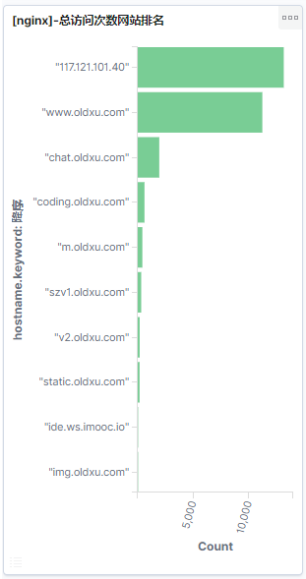
```
8
9
10 #- type: log
11 #  enabled: true
12 #  paths:
13 #    - /var/log/nginx/error.log
14 #  tags: ["error"]
15
16
17 output.logstash:
18   hosts: ["10.0.0.151:5044"]
19
20
21
22
23 tash-node1 conf.d]# cat
   input_filebeat_nginx_output_es.conf
24
25 input {
26     beats {
27         port => 5044
28     }
29 }
30
31 filter {
32     grok {
33         match => { "message" => "%
   {COMBINEDAPACHELOG}" }
34     }
35
36     geoip {
37         source => "clientip"
38     }
39
```

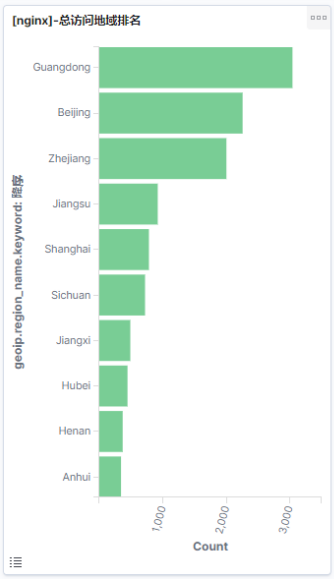
```
    #30/Dec/2019:11:59:18 +0800
    date {
        match => ["timestamp", "dd/MMM/yyyy:HH:mm:ss Z"]
        target => "@timestamp"
        timezone => "Asia/Shanghai"
    }

    useragent {
        source => "agent"
        target => "agent"
    }

    mutate {
        remove_field => [ "message","timestamp" ]
        convert => [ "bytes" , "integer" ]
    }

}

output {
    stdout {
        codec => rubydebug
    }


    elasticsearch {
        hosts => ["10.0.0.161:9200","10.0.0.162:9200","10.0.0.163:9200"]
        index => "kibana-nginx-%{+YYYY.MM.dd}"     #索引名称
        template_overwrite => true
    }
```

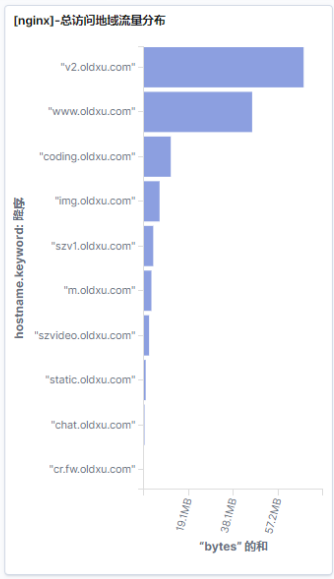
```
70 }
```

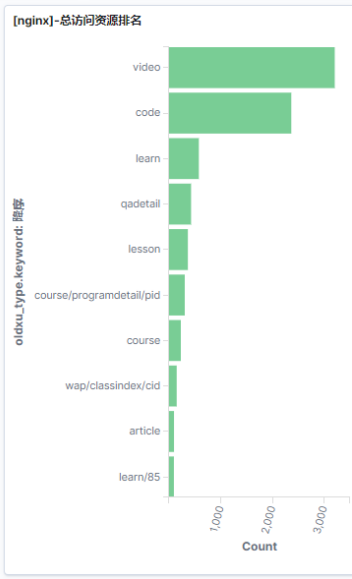
```
#www.oldxu.com 为请求的域名 api3为请求的资源 uid=为用户id
124.161.176.119 - - [10/Nov/2019:00:01:52 +0800] "POST /api3/appover HTTP/1.1" 200 103 "www.oldxu.com" "-" code=B157963E-5BDA-4090-A021-A3D46D2E6BA2&secrect=f0fbb455c7aebc69c5cc39d68c7859fe&time=9441&timestamp=1478707307012&token=12d0f0cfa1efb81e42c321f027bbe752&uid=4384521 "oldxu/5.0.2 (iPhone; iOS 10.1.1; Scale/2.00)" "-" 10.100.136.65:80 200 0.011 0.011
```

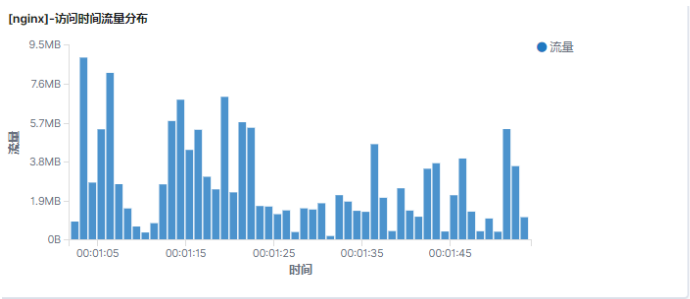



[nginx]-总访问次数网站排名
hostname.keyword: 降序
"117.121.101.40"
"www.oldxu.com"
"chat.oldxu.com"
"coding.oldxu.com"
"m.oldxu.com"
"szv1.oldxu.com"
"v2.oldxu.com"
"static.oldxu.com"
"ide.ws.imooc.io"
"img.oldxu.com"
5,000
10,000
Count


[nginx]-总访问地域排名
geoip.region_name.keyword: 降序
Guangdong
Beijing
Zhejiang
Jiangsu
Shanghai
Sichuan
Jiangxi
Hubei
Henan
Anhui
1,000
2,000
3,000
Count


[nginx]-总访问地域流量分布
hostname.keyword: 降序
"v2.oldxu.com"
"www.oldxu.com"
"coding.oldxu.com"
"img.oldxu.com"
"szv1.oldxu.com"
"m.oldxu.com"
"szvideo.oldxu.com"
"static.oldxu.com"
"chat.oldxu.com"
"cr.fw.oldxu.com"
19.1MB
38.1MB
57.2MB
"bytes" 的和

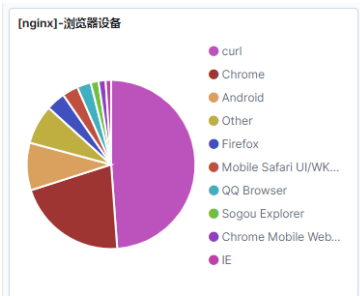

[nginx]-总访问资源排名
oldxu_type.keyword: 降序
video
code
learn
qadetail
lesson
course/programdetail/pid
course
wap/classindex/cid
article
learn/85
1,000
2,000
3,000
Count

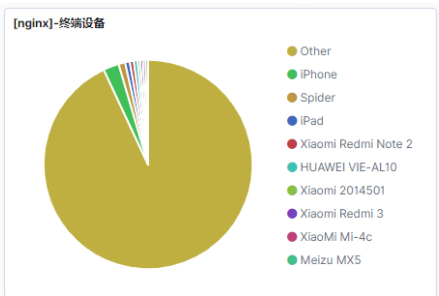

[nginx]-访问时间流量分布
流量
9.5MB
7.6MB
5.7MB
3.8MB
1.9MB
0B
00:01:05
00:01:15
00:01:25
00:01:35
00:01:45
时间

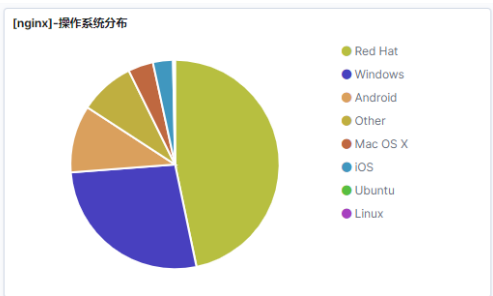

[nginx]-大于1s的请求
计数
60
50
40
30
20
10
0
00:01:05
00:01:15
00:01:25
00:01:35
00:01:45
时间

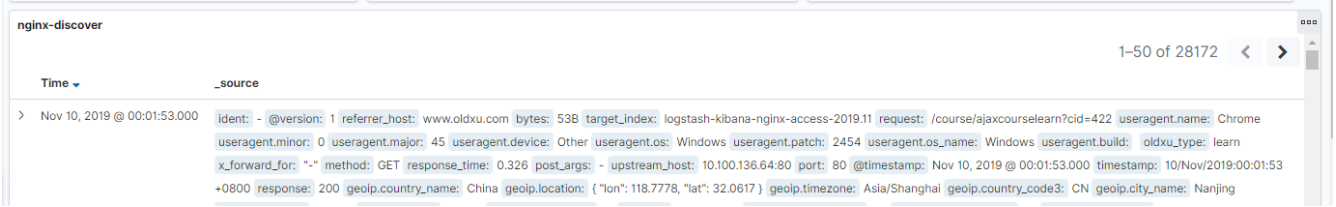

[nginx]-访问来源
www.baidu.com
www.oldxu.com
coding.oldxu.com
m.oldxu.com
blog.jobbole.com


[nginx]-地域分布
EUROPE
ASIA
AFRICA
计数
22 – 3,679.75
3,679.75 – 7,337.5


[nginx]-浏览器设备
curl
Chrome
Android
Other
Firefox
Mobile Safari UI/WK...
QQ Browser
Sogou Explorer
Chrome Mobile Web...
IE


[nginx]-终端设备
Other
iPhone
Spider
iPad
Xiaomi Redmi Note 2
HUAWEI VIE-AL10
Xiaomi 2014501
Xiaomi Redmi 3
XiaoMi Mi-4c
Meizu MX5


[nginx]-操作系统分布
Red Hat
Windows
Android
Other
Mac OS X
iOS
Ubuntu
Linux


nginx-discover
1–50 of 28172
Time
_source
Nov 10, 2019 @ 00:01:53.000
ident: - @version: 1 referrer_host: www.oldxu.com bytes: 53B target_index: logstash-kibana-nginx-access-2019.11 request: /course/ajaxcourselearn?cid=422 useragent.name: Chrome
useragent.minor: 0 useragent.major: 45 useragent.device: Other useragent.os: Windows useragent.patch: 2454 useragent.os_name: Windows useragent.build: oldxu_type: learn
x_forward_for: "-" method: GET response_time: 0.326 post_args: - upstream_host: 10.100.136.64:80 port: 80 @timestamp: Nov 10, 2019 @ 00:01:53.000 timestamp: 10/Nov/2019:00:01:53
+0800 response: 200 geoip.country_name: China geoip.location: { "lon": 118.7778, "lat": 32.0617 } geoip.timezone: Asia/Shanghai geoip.country_code3: CN geoip.city_name: Nanjing

Redis

收集容器日志

https://www.cnblogs.com/xuliangwei/p/12172960.html