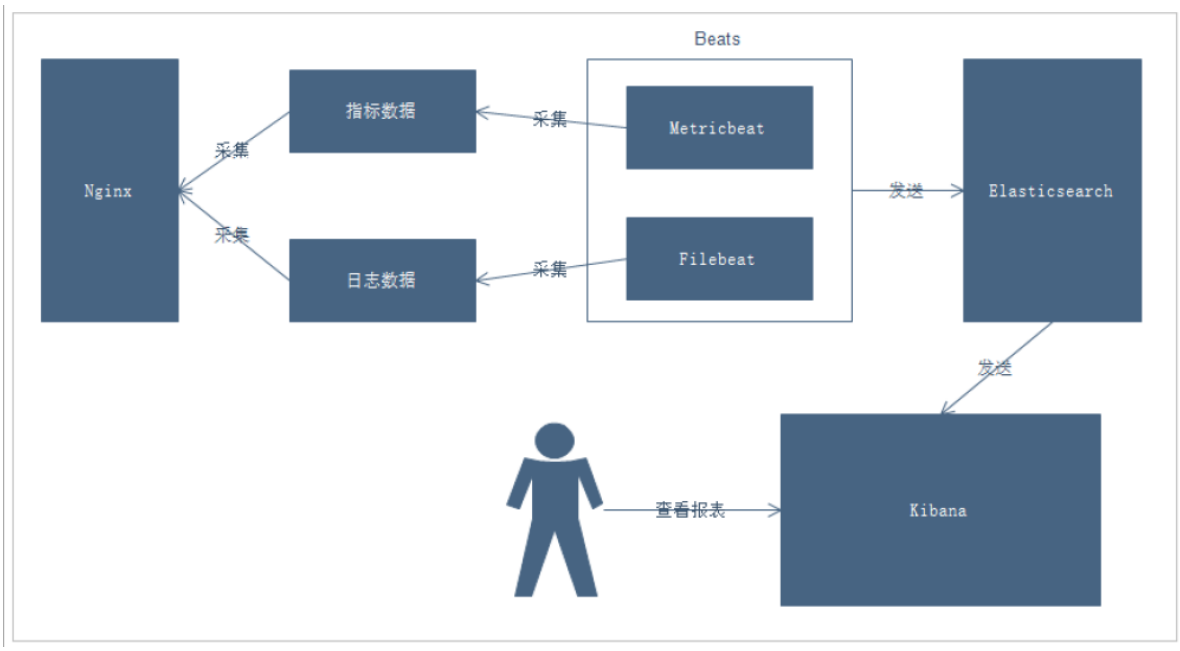
Beats入门简介

使用Beat收集nginx日志和指标数据

项目需求

Nginx是一款非常优秀的web服务器，往往nginx服务会作为项目的访问入口，那么，nginx的性能保障就变得非常重要了，如果nginx的运行出现了问题就会对项目有较大的影响，所以，我们需要对nginx的运行有监控措施，实时掌握nginx的运行情况，那就需要收集nginx的运行指标和分析nginx的运行日志了。

业务流程



说明：

- 通过Beats采集Nginx的指标数据和日志数据
- Beats采集到数据后发送到Elasticsearch中
- Kibana读取数据进行分析
- 用户通过Kibana进行查看分析报表

部署Nginx

部署教程可以参考这篇博客：CentOS下如何安装Nginx?

部署完成后，我们就可以启动nginx了

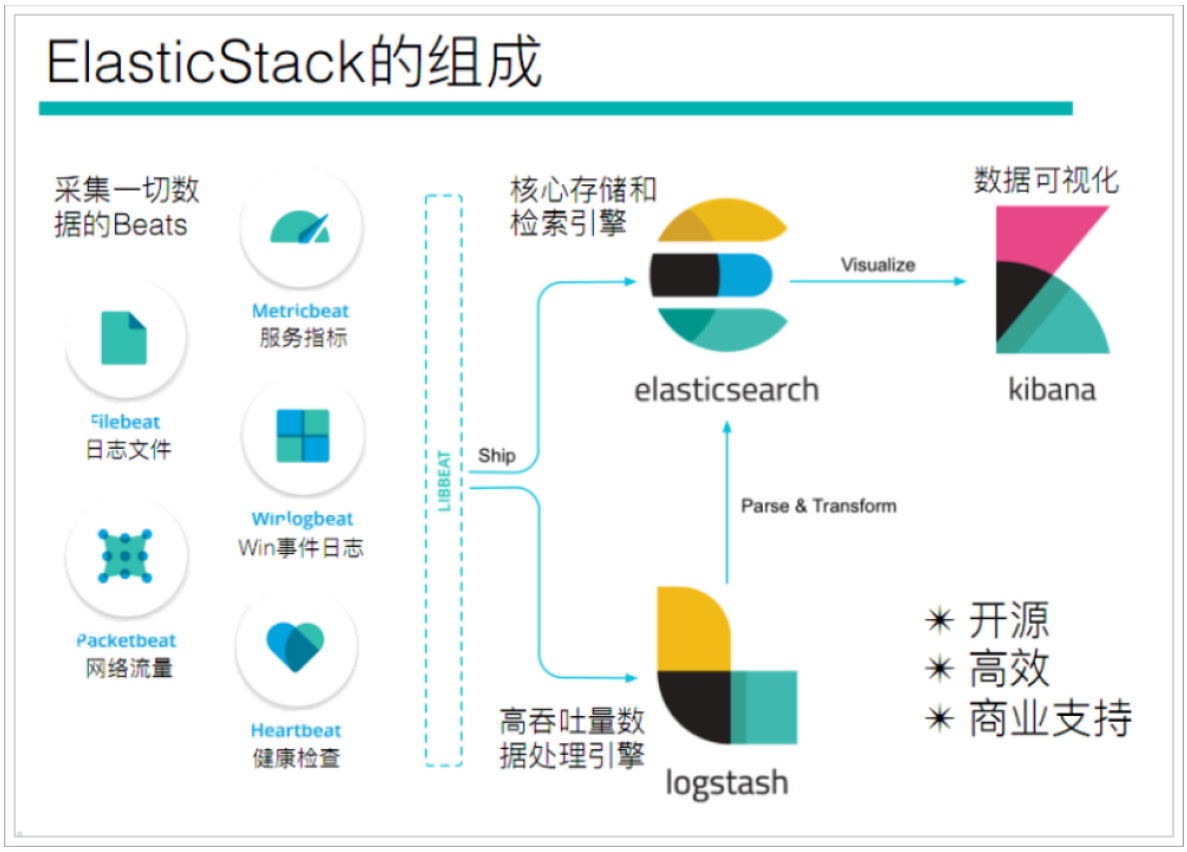
启动完成后，我们通过下面命令，就可以获取到nginx中的内容了

```
tail -f /var/log/nginx/access.log
```

Beats简介

通过查看ElasticStack可以发现，Beats主要用于采集数据

官网地址：https://www.elastic.co/cn/beats/



Beats平台其实是一个轻量性数据采集器，通过集合多种单一用途的采集器，从成百上千台机器中向Logstash或ElasticSearch中发送数据。

通过Beats包含以下的数据采集功能

- Filebeat：采集日志文件
- Metricbeat：采集指标
- Packetbeat：采集网络数据

如果我们的数据不需要任何处理，那么就可以直接发送到ElasticSearch中

如果们的数据需要经过一些处理的话，那么就可以发送到Logstash中，然后处理完成后，在发送到ElasticSearch

最后在通过Kibana对我们的数据进行一系列的可视化展示

Filebeat

介绍

Filebeat是一个轻量级的日志采集器

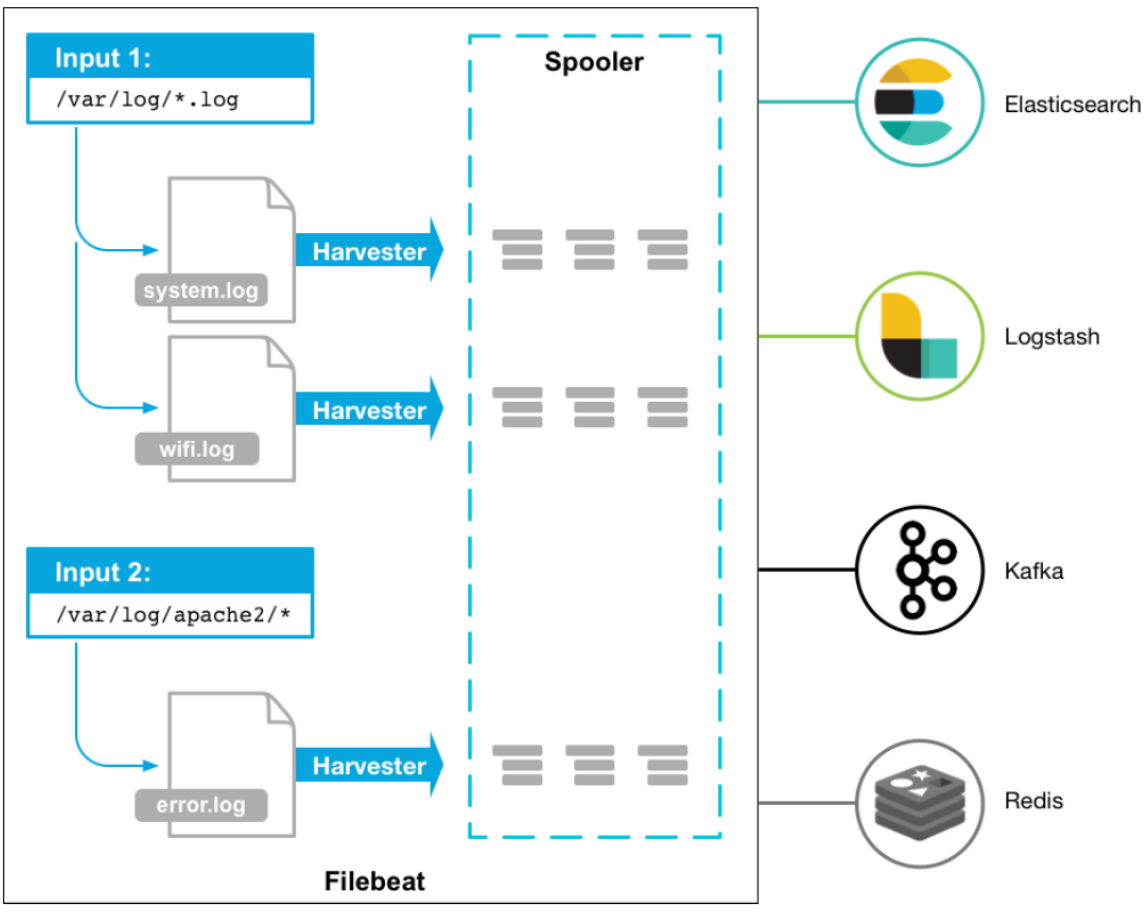
为什么要用Filebeat?

当你面对成百上千、甚至成千上万的服务器、虚拟机和溶气气生成的日志时，请告别SSH吧！Filebeat将为你提供一种轻量型方法，用于转发和汇总日志与文件，让简单的事情不再繁华，关于Filebeat的记住以下两点：

- 轻量级日志采集器
- 输送至ElasticSearch或者Logstash，在Kibana中实现可视化

架构

用于监控、收集服务器日志文件.



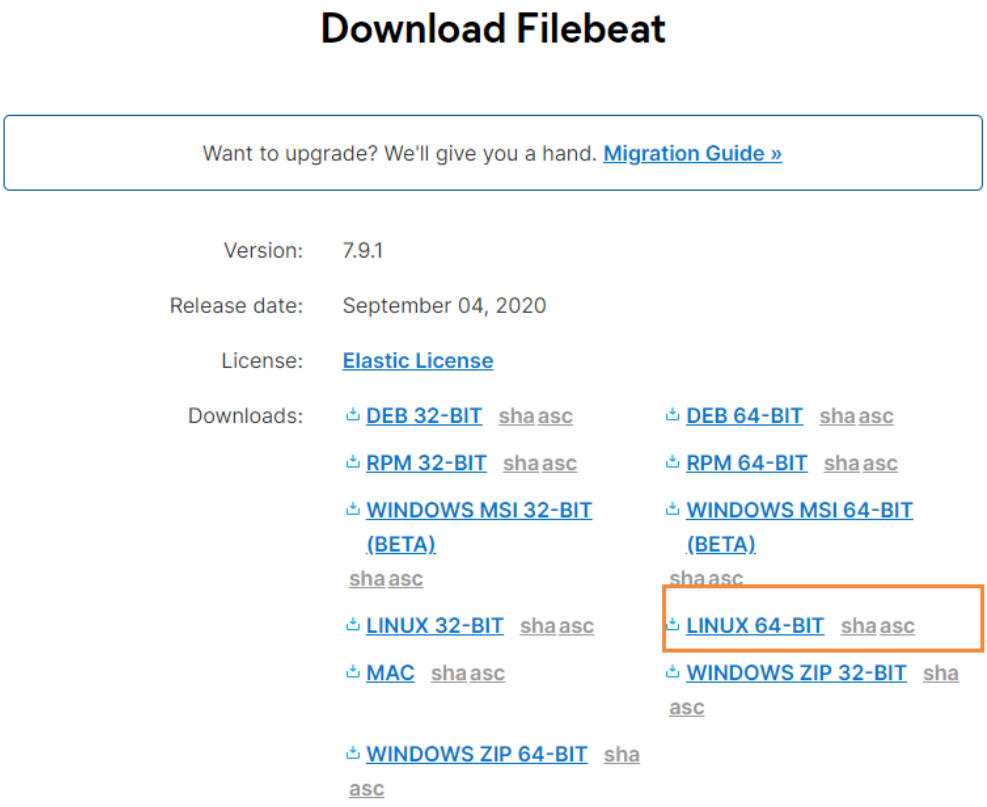
流程如下：

- 首先是input输入，我们可以指定多个数据输入源，然后通过通配符进行日志文件的匹配
- 匹配到日志后，就会使用Harvester（收割机），将日志源源不断的读取到来
- 然后收割机收割到的日志，就传递到Spooler（卷轴），然后卷轴就在将他们传到对应的地方

下载

官网地址：https://www.elastic.co/cn/downloads/beats/filebeat

选中对应版本的Filebeat，我这里是Centos部署的，所以下载Linux版本



下载后，我们上传到服务器上，然后创建一个文件夹

```
# 创建文件夹
mkdir -p /soft/beats
# 解压文件
tar -zxvf filebeat-7.9.1-linux-x86_64.tar.gz
# 重命名
mv filebeat-7.9.1-linux-x86_64/ filebeat
```

然后我们进入到filebeat目录下，创建对应的配置文件

```
# 进入文件夹
cd filebeats
# 创建配置文件
vim mogublog.yml
```

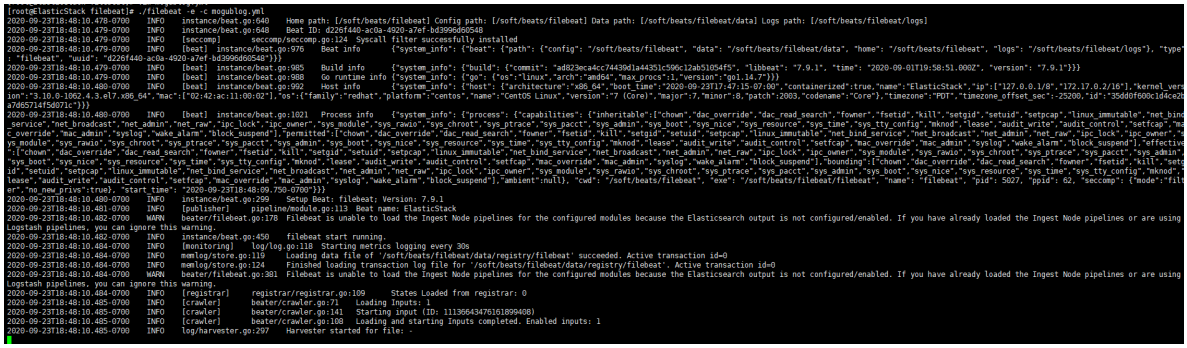
添加如下内容

```
filebeat.inputs: # filebeat input输入
- type: stdin     # 标准输入
  enabled: true  # 启用标准输入
setup.template.settings:
  index.number_of_shards: 3 # 指定下载数
output.console:  # 控制台输出
  pretty: true   # 启用美化功能
  enable: true
```

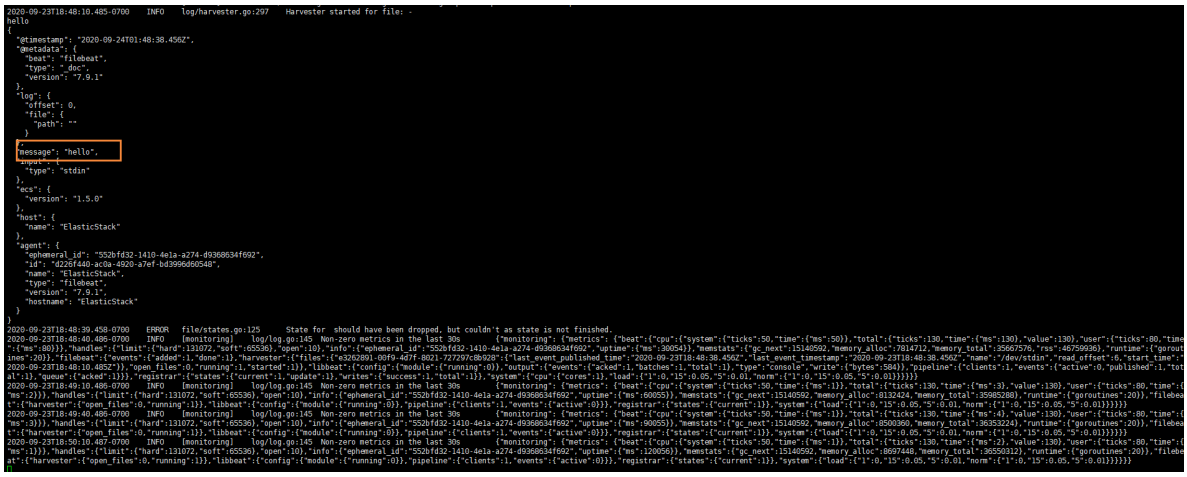
启动

在我们添加完配置文件后，我们就可以对filebeat进行启动了

```
./filebeat -e -c mogublog.yml
```



然后我们在控制台输入hello，就能看到我们会有一个json的输出，是通过读取到我们控制台的内容后输出的



内容如下

```
{
    "@timestamp":"2019-01-12T12:50:03.585Z",
    "@metadata":{ #元数据信息
        "beat":"filebeat",
        "type":"doc",
        "version":"6.5.4"
    },
    "source":"",
    "offset":0,
    "message":"hello", #元数据信息
    "prospector":{
        "type":"stdin" #元数据信息
```

```
    },
    "input":{ #控制台标准输入
        "type":"stdin"
    },
    "beat":{  #beat版本以及主机信息
        "name":"itcast01",
        "hostname":"ElasticStack",
        "version":"6.5.4"
    },
    "host":{
        "name":"ElasticStack"
    }
}
```

读取文件

我们需要再次创建一个文件，叫 mogublog-log.yml，然后在文件里添加如下内容

```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /soft/beats/logs/*.log
setup.template.settings:
  index.number_of_shards: 3
output.console:
  pretty: true
  enable: true
```

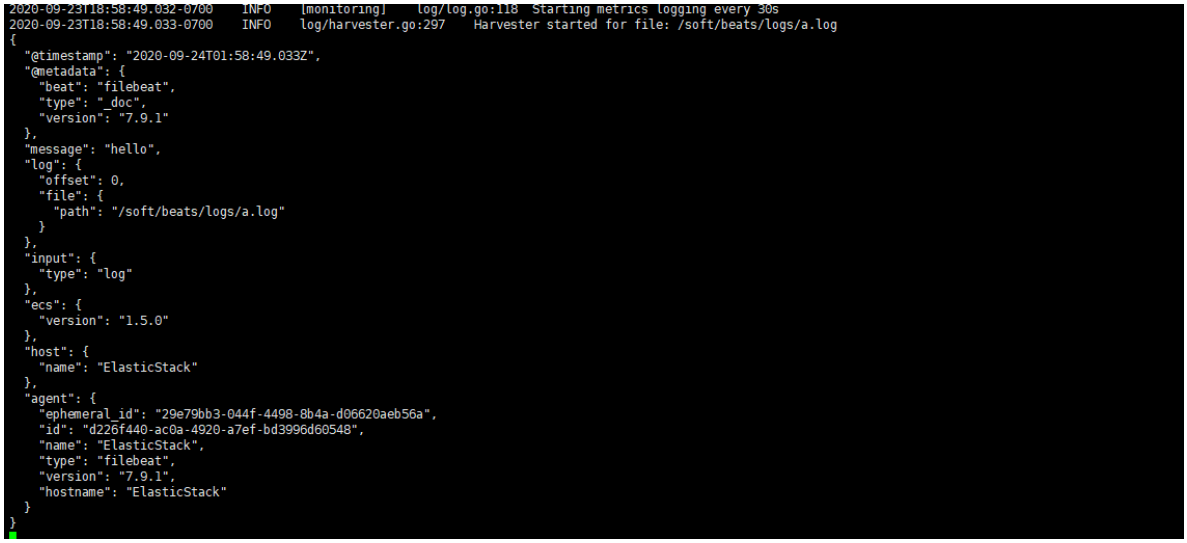
添加完成后，我们在到下面目录创建一个日志文件

```
# 创建文件夹
mkdir -p /soft/beats/logs

# 进入文件夹
cd /soft/beats/logs

# 追加内容
echo "hello" >> a.log
```

然后我们再次启动filebeat

```
 ./filebeat -e -c mogublog-log.yml
```

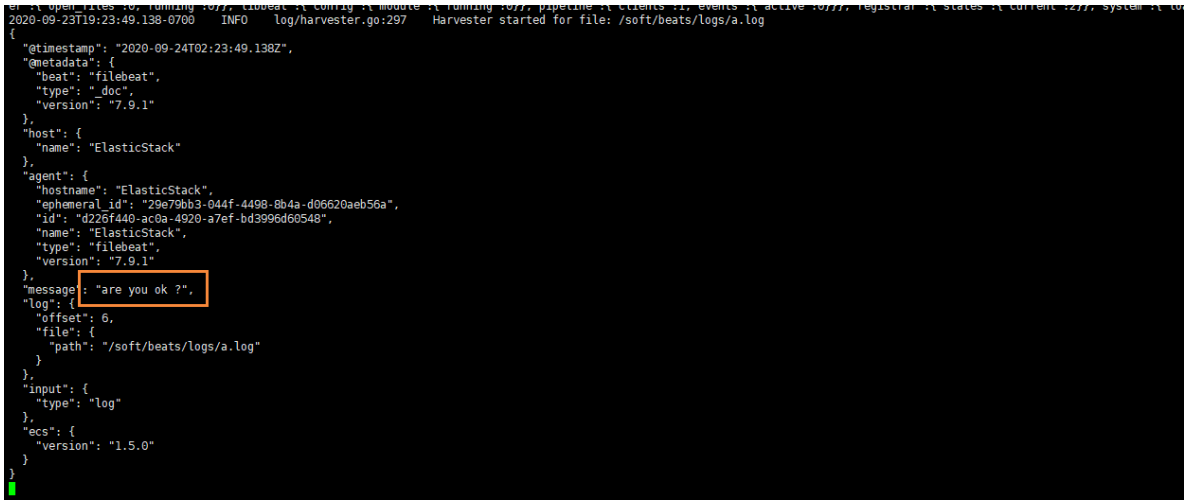
能够发现，它已经成功加载到了我们的日志文件 a.log



同时我们还可以继续往文件中追加内容

```
echo "are you ok ?" >> a.log
```

追加后，我们再次查看filebeat，也能看到刚刚我们追加的内容



可以看出，已经检测到日志文件有更新，立刻就会读取到更新的内容，并且输出到控制台。

自定义字段

但我们的元数据没办法支撑我们的业务时，我们还可以自定义添加一些字段

```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /soft/beats/logs/*.log
  tags: ["web", "test"]  #添加自定义tag，便于后续的处理
  fields:  #添加自定义字段
    from: test-web
  fields_under_root: true #true为添加到根节点，false为添加到子节点中
setup.template.settings:
  index.number_of_shards: 3
output.console:
  pretty: true
  enable: true
```

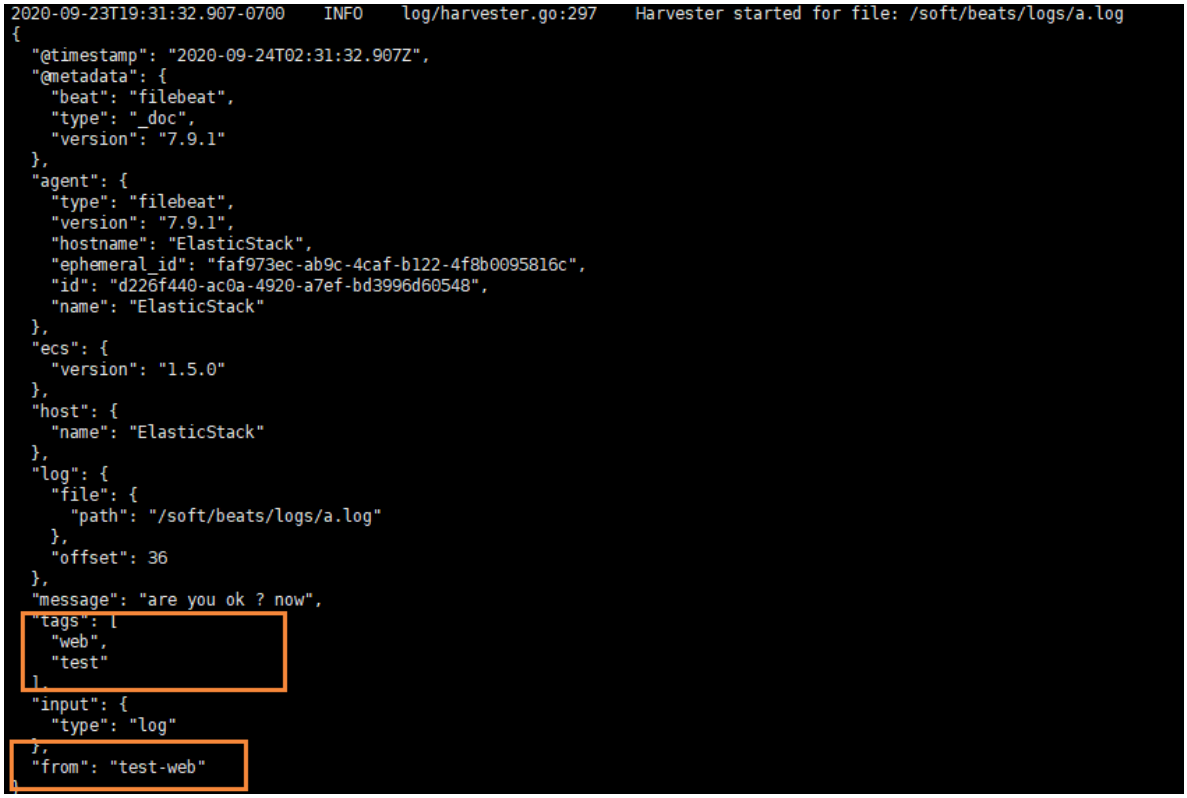
添加完成后，我们重启 filebeat

```
./filebeat -e -c mogublog-log.yml
```

然后添加新的数据到 a.log中

```
echo "test-web" >> a.log
```

我们就可以看到字段在原来的基础上，增加了两个

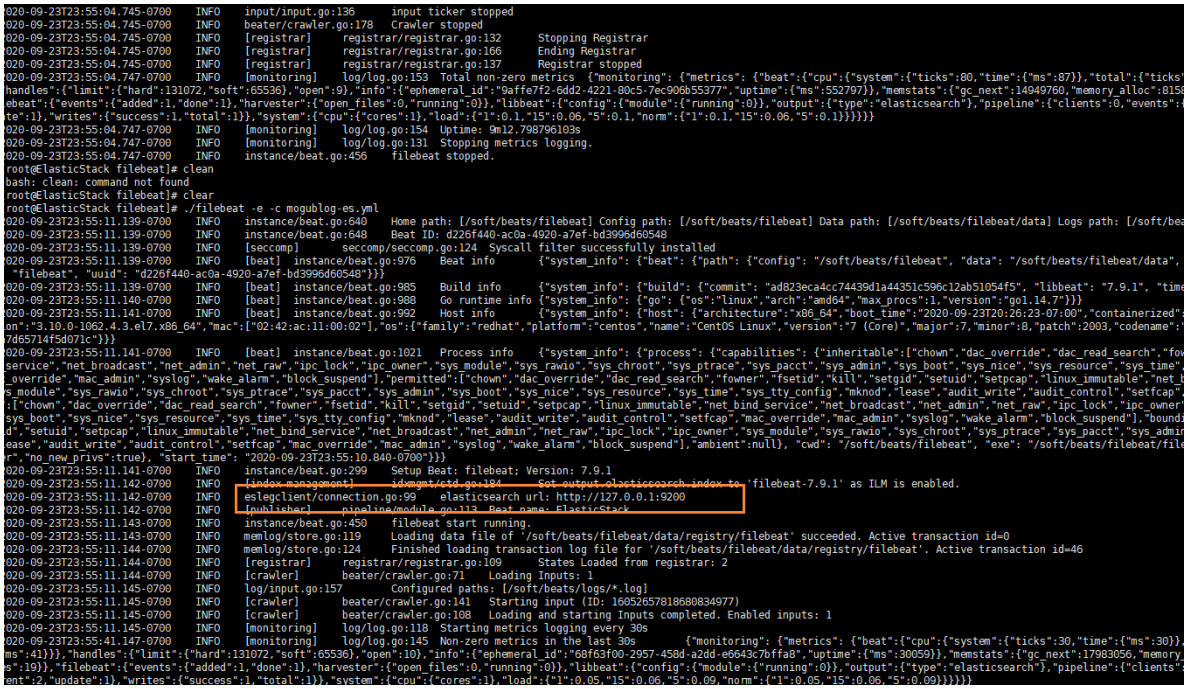


输出到ElasticSearch

我们可以通过配置，将修改成如下所示

```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /soft/beats/logs/*.log
  tags: ["web", "test"]
  fields:
    from: test-web
  fields_under_root: false
setup.template.settings:
  index.number_of_shards: 1
output.elasticsearch:
  hosts: ["127.0.0.1:9200"]
```

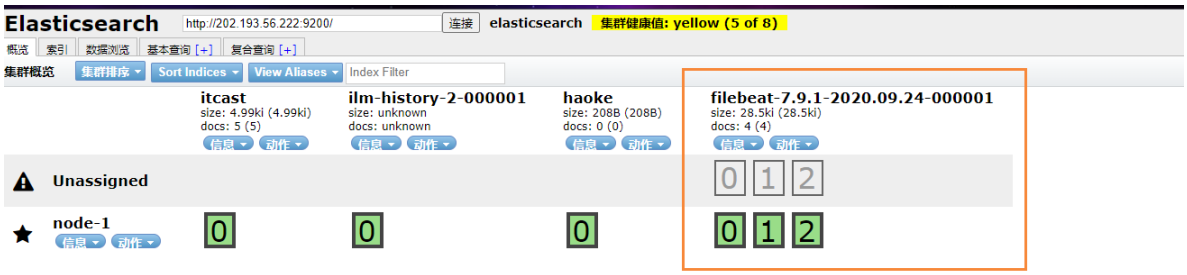
启动成功后，我们就能看到它已经成功连接到了es了



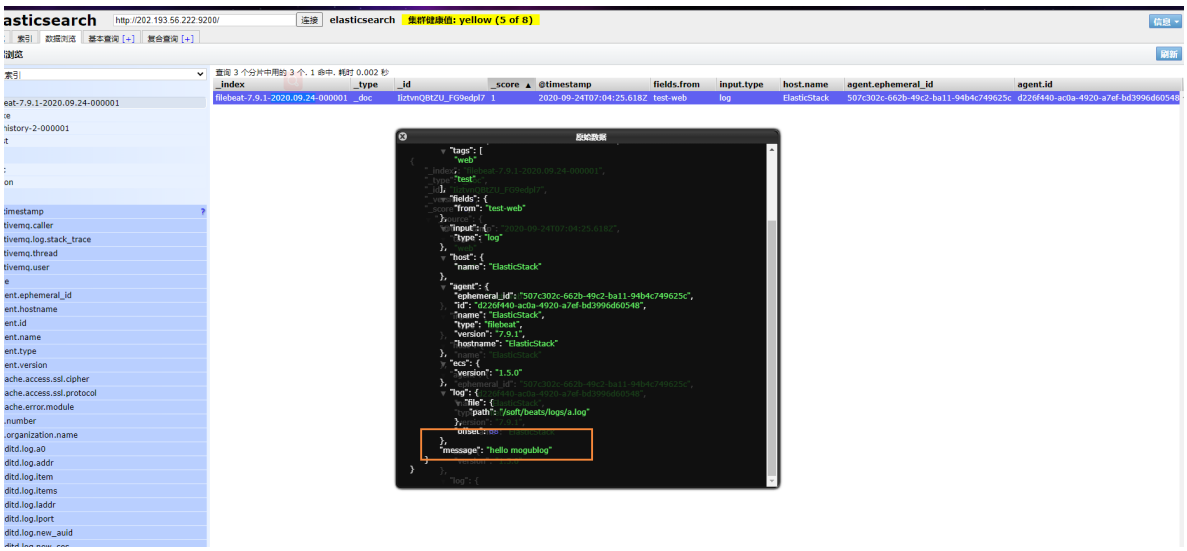
然后我们到刚刚的 logs文件夹向 a.log文件中添加内容

```
echo "hello mogublog" >> a.log
```

在ES中，我们可以看到，多出了一个 filebeat的索引库



然后我们浏览对应的数据，看看是否有插入的数据内容



Filebeat工作原理

Filebeat主要由下面几个组件组成： harvester、prospector 、input

harvester

- 负责读取单个文件的内容
- harvester逐行读取每个文件（一行一行读取），并把这些内容发送到输出
- 每个文件启动一个harvester，并且harvester负责打开和关闭这些文件，这就意味着harvester运行时文件描述符保持着打开的状态。
- 在harvester正在读取文件内容的时候，文件被删除或者重命名了，那么Filebeat就会续读这个文件，这就会造成一个问题，就是只要负责这个文件的harvester没用关闭，那么磁盘空间就不会被释放，默认情况下，Filebeat保存问价你打开直到close_inactive到达

prospector

- prospector负责管理harvester并找到所有要读取的文件来源
- 如果输入类型为日志，则查找器将查找路径匹配的所有文件，并为每个文件启动一个harvester
- Filebeat目前支持两种prospector类型：log和stdin
- Filebeat如何保持文件的状态
    - Filebeat保存每个文件的状态并经常将状态刷新到磁盘上的注册文件中
    - 该状态用于记住harvester正在读取的最后偏移量，并确保发送所有日志行。
    - 如果输出（例如ElasticSearch或Logstash）无法访问，Filebeat会跟踪最后发送的行，并在输出再次可以用时继续读取文件。
    - 在Filebeat运行时，每个prospector内存中也会保存的文件状态信息，当重新启动Filebat时，将使用注册文件的数量来重建文件状态，Filebeat将每个harvester在从保存的最后偏移量继续读取
    - 文件状态记录在data/registry文件中

input

- 一个input负责管理harvester，并找到所有要读取的源
- 如果input类型是log，则input查找驱动器上与已定义的glob路径匹配的所有文件，并为每个文件启动一个harvester
- 每个input都在自己的Go例程中运行
- 下面的例子配置Filebeat从所有匹配指定的glob模式的文件中读取行

```
filebeat.inputs:
- type: log
  paths:
    - /var/log/*.log
    - /var/path2/*.log
```

启动命令

```
./filebeat -e -c mogublog-es.yml
./filebeat -e -c mogublog-es.yml -d "publish"
```

参数说明

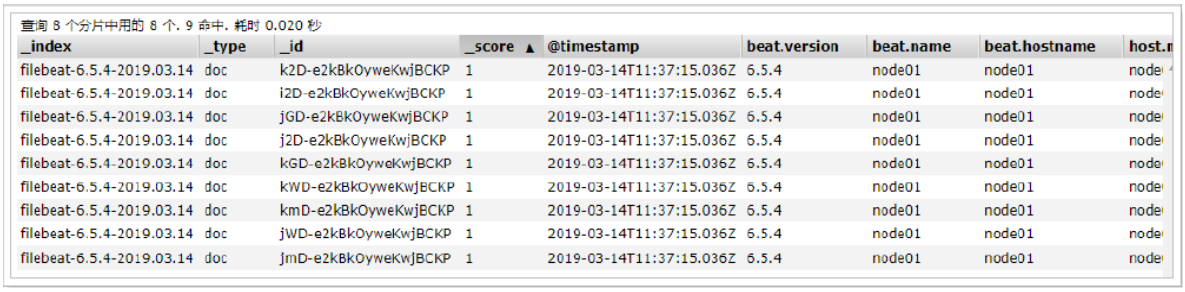
- **-e**：输出到标准输出，默认输出到syslog和logs下
- **-c**：指定配置文件
- **-d**：输出debug信息

读取Nginx中的配置文件

我们需要创建一个 mogublog-nginx.yml配置文件

```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /soft/nginx/*.log
  tags: ["nginx"]
  fields_under_root: false
setup.template.settings:
  index.number_of_shards: 1
output.elasticsearch:
  hosts: ["127.0.0.1:9200"]
```

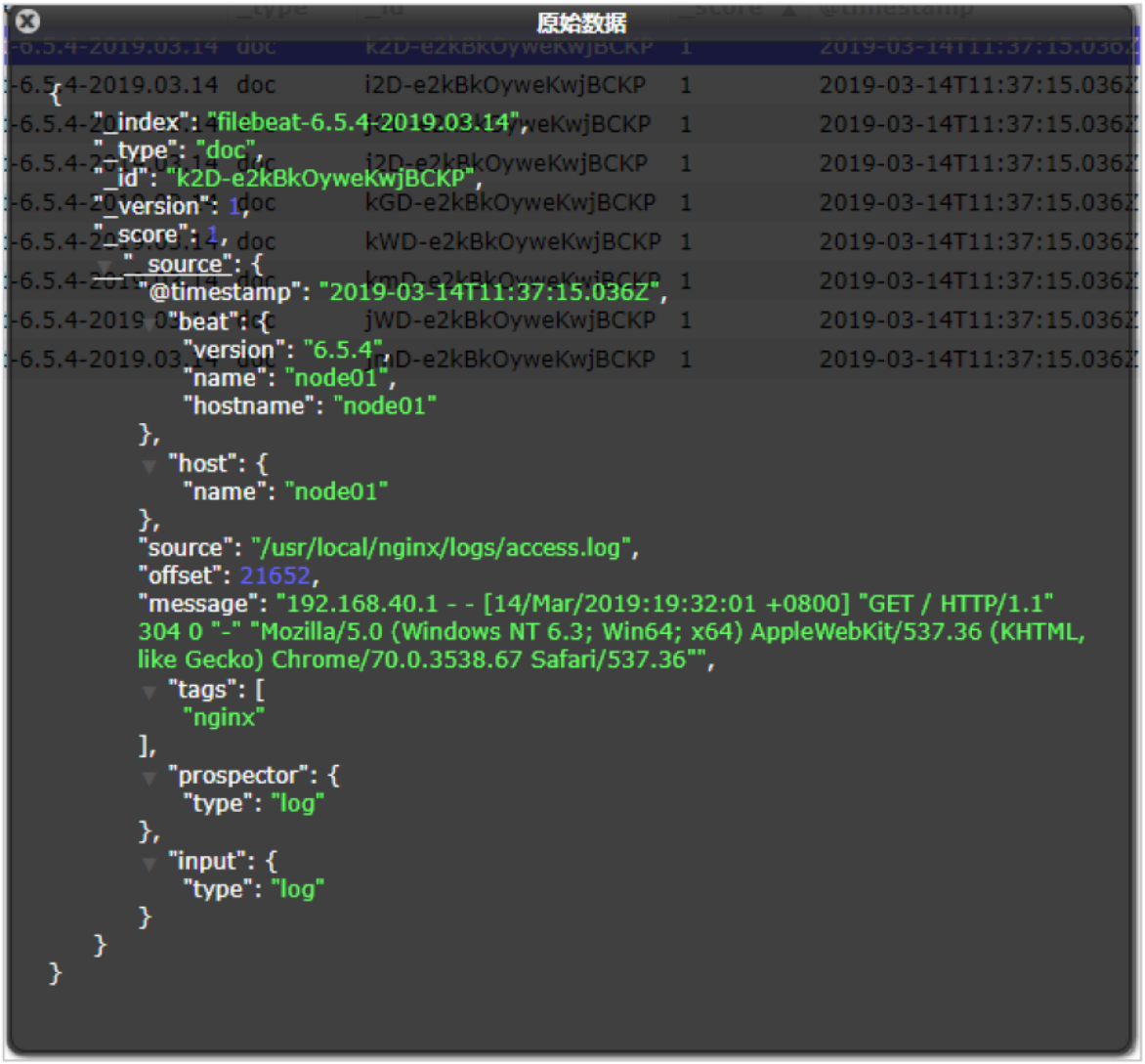
启动后，可以在Elasticsearch中看到索引以及查看数据



查询 8 个分片中用的 8 个，9 命中，耗时 0.020 秒

_index	_type	_id	_score ▲	@timestamp	beat.version	beat.name	beat.hostname	host.n
filebeat-6.5.4-2019.03.14	doc	k2D-e2kBkOyweKwjBCKP	1	2019-03-14T11:37:15.036Z	6.5.4	node01	node01	node
filebeat-6.5.4-2019.03.14	doc	i2D-e2kBkOyweKwjBCKP	1	2019-03-14T11:37:15.036Z	6.5.4	node01	node01	node
filebeat-6.5.4-2019.03.14	doc	jGD-e2kBkOyweKwjBCKP	1	2019-03-14T11:37:15.036Z	6.5.4	node01	node01	node
filebeat-6.5.4-2019.03.14	doc	j2D-e2kBkOyweKwjBCKP	1	2019-03-14T11:37:15.036Z	6.5.4	node01	node01	node
filebeat-6.5.4-2019.03.14	doc	kGD-e2kBkOyweKwjBCKP	1	2019-03-14T11:37:15.036Z	6.5.4	node01	node01	node
filebeat-6.5.4-2019.03.14	doc	kWD-e2kBkOyweKwjBCKP	1	2019-03-14T11:37:15.036Z	6.5.4	node01	node01	node
filebeat-6.5.4-2019.03.14	doc	kmD-e2kBkOyweKwjBCKP	1	2019-03-14T11:37:15.036Z	6.5.4	node01	node01	node
filebeat-6.5.4-2019.03.14	doc	jWD-e2kBkOyweKwjBCKP	1	2019-03-14T11:37:15.036Z	6.5.4	node01	node01	node
filebeat-6.5.4-2019.03.14	doc	jmD-e2kBkOyweKwjBCKP	1	2019-03-14T11:37:15.036Z	6.5.4	node01	node01	node

可以看到，在message中已经获取到了nginx的日志，但是，内容并没有经过处理，只是读取到原数据，那么对于我们后期的操作是不利的，有办法解决吗？



Module

前面要想实现日志数据的读取以及处理都是自己手动配置的，其实，在Filebeat中，有大量的Module，可以简化我们的配置，直接就可以使用，如下：

```
./filebeat modules list
```

得到的列表如下所示

```
Disabled:
activemq
apache
auditd
aws
azure
barracuda
bluecoat
cef
checkpoint
cisco
coredns
crowdstrike
cylance
elasticsearch
envoyproxy
f5
fortinet
googlecloud
gsuite
haproxy
ibmmq
icinga
iis
imperva
infoblox
iptables
juniper
kafka
kibana
logstash
microsoft
misp
mongodb
mssql
mysql
nats
netflow
netscout
nginx
o365
okta
osquery
panw
postgresql
rabbitmq
radware
```

```
redis
santa
sonicwall
sophos
squid
suricata
system
tomcat
traefik
zeek
zscaler
```

可以看到，内置了很多的module，但是都没有启用，如果需要启用需要进行enable操作：

```
#启动
./filebeat modules enable nginx
#禁用
./filebeat modules disable nginx
```

可以发现，nginx的module已经被启用。

nginx module 配置

我们到下面的目录，就能看到module的配置了

```
# 进入到module目录
cd modules.d/
#查看文件
vim nginx.yml.disabled
```

得到的文件内容如下所示

```
# Module: nginx
# Docs: https://www.elastic.co/guide/en/beats/filebeat/7.9/filebeat-module-nginx.html

- module: nginx
  # Access logs
  access:
    enabled: true
    # 添加日志文件
    var.paths: ["/var/log/nginx/access.log*"]

    # Set custom paths for the log files. If left empty,
    # Filebeat will choose the paths depending on your OS.
    #var.paths:

  # Error logs
  error:
    enabled: true
    var.paths: ["/var/log/nginx/error.log*"]
```

配置filebeat

我们需要修改刚刚的mogublog-nginx.yml文件，然后添加到我们的module

```
filebeat.inputs:
setup.template.settings:
  index.number_of_shards: 1
output.elasticsearch:
  hosts: ["127.0.0.1:9200"]
filebeat.config.modules:
  path: ${path.config}/modules.d/*.yml
  reload.enabled: false
```

测试

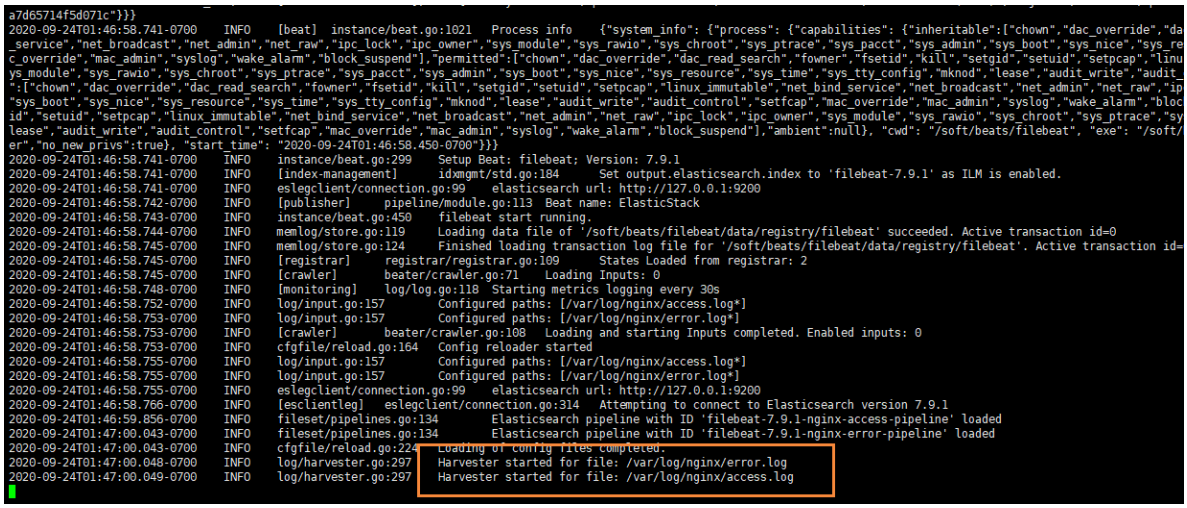
我们启动我们的filebeat

```
./filebeat -e -c itcast-nginx.yml
```

如果启动的时候发现出错了，错误如下所示，执行如图所示的脚本即可 【新版本的ES好像不会出现这个错误】

```
#启动会出错，如下
ERROR fileset/factory.go:142 Error loading pipeline: Error loading pipeline for fileset nginx/access: This module requires the following Elasticsearch plugins: ingest-user-agent, ingest-geoip. You can install them by running the following commands on all the Elasticsearch nodes:
  sudo bin/elasticsearch-plugin install ingest-user-agent
  sudo bin/elasticsearch-plugin install ingest-geoip
```

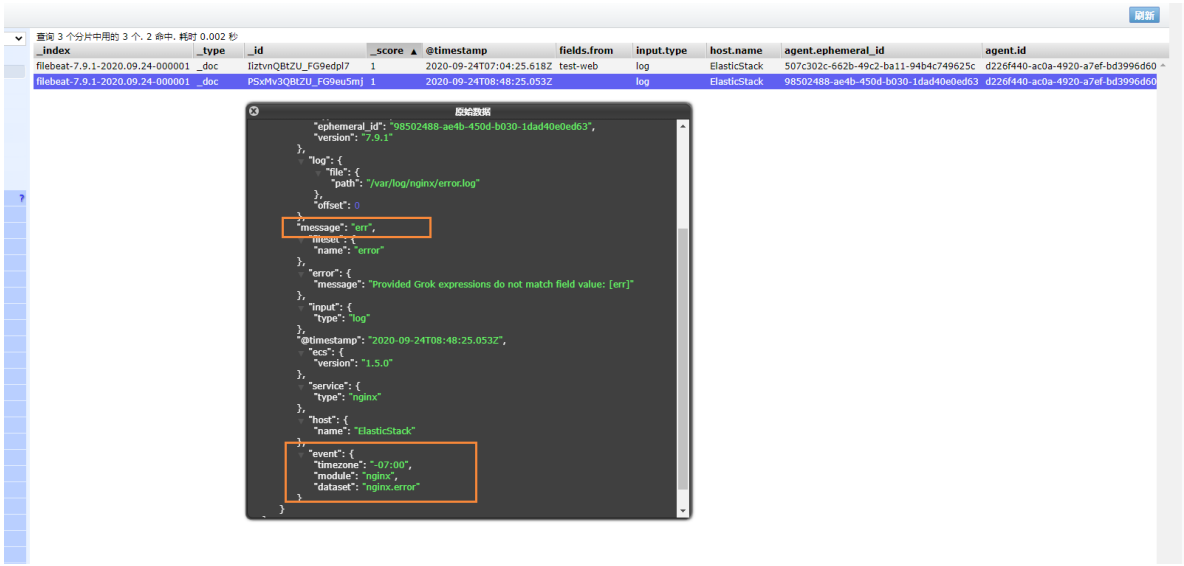
启动成功后，能看到日志记录已经成功刷新进去了



我们可以测试一下，刷新nginx页面，或者向错误日志中，插入数据

```
echo "err" >> error.log
```

能够看到，刚刚的记录已经成功插入了



关于module的其它使用，可以参考文档：

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

Metricbeat

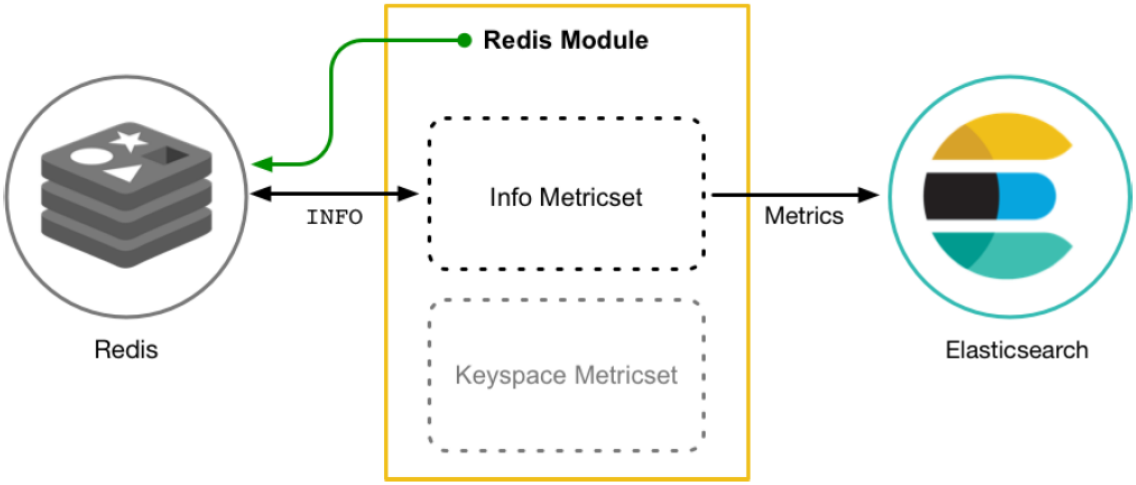
- 定期收集操作系统或应用服务的指标数据
- 存储到Elasticsearch中，进行实时分析

Metricbeat组成

Metricbeat有2部分组成，一部分是Module，另一个部分为Metricset

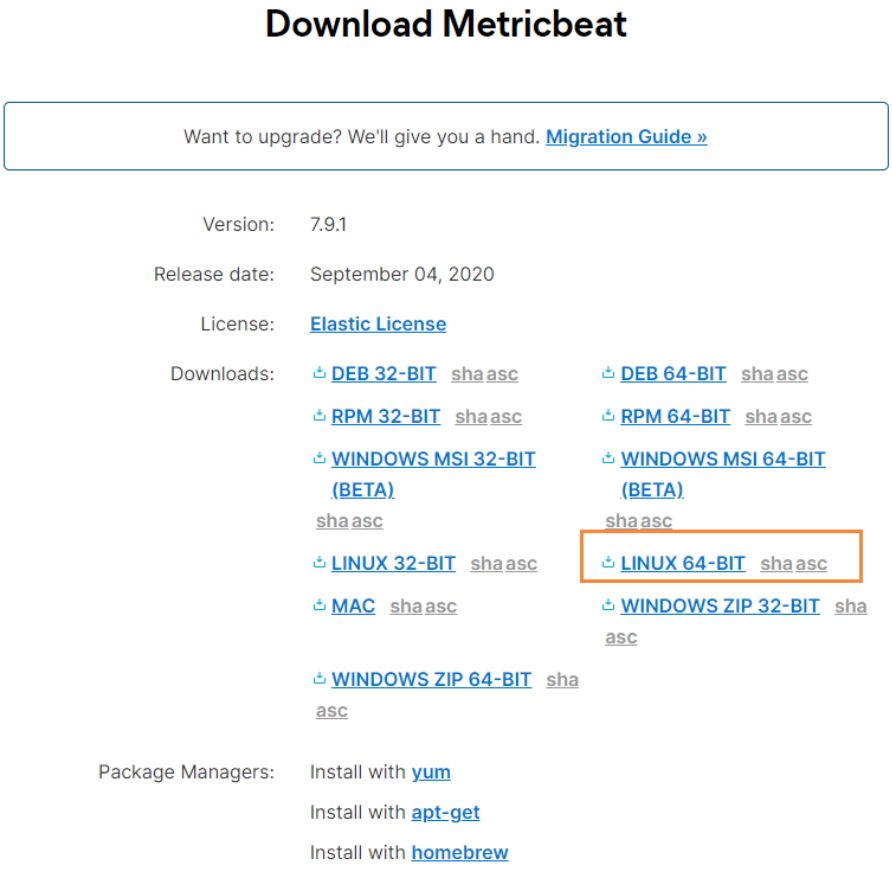
- Module
  - 收集的对象：如 MySQL、Redis、Nginx、操作系统等
- Metricset
  - 收集指标的集合：如 cpu、memory，network等

以Redis Module为例：



下载

首先我们到官网，找到Metricbeat进行下载



下载完成后，我们通过xftp工具，移动到指定的目录下

```
# 移动到该目录下
cd /soft/beats
# 解压文件
tar -zxvf
# 修改文件名
mv  metricbeat
```

然后修改配置文件

```
vim metricbeat.yml
```

添加如下内容

```
metricbeat.config.modules:
  path: ${path.config}/modules.d/*.yml
  reload.enabled: false
setup.template.settings:
  index.number_of_shards: 1
  index.codec: best_compression
setup.kibana:
output.elasticsearch:
  hosts: [""127.0.0.1:9200"]
processors:
  - add_host_metadata: ~
  - add_cloud_metadata: ~
```

默认会指定的配置文件，就是在

```
${path.config}/modules.d/*.yml
```

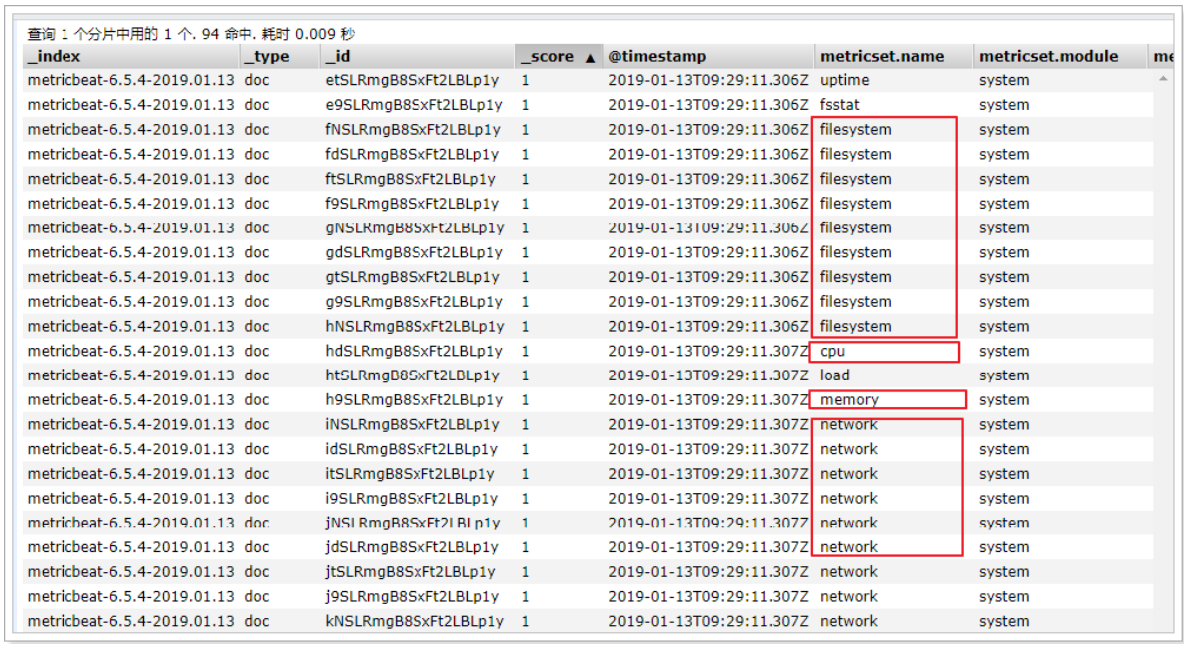
也就是 system.yml文件，我们也可以自行开启其它的收集

启动

在配置完成后，我们通过如下命令启动即可

```
./metricbeat -e
```

在ELasticsearch中可以看到，系统的一些指标数据已经写入进去了：



查询：个分片中用的 1 个，94 命中，耗时 0.009 秒

_index	_type	_id	_score ▲	@timestamp	metricset.name	metricset.module	me
metricbeat-6.5.4-2019.01.13	doc	etSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.306Z	uptime	system	
metricbeat-6.5.4-2019.01.13	doc	e9SLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.306Z	fsstat	system	
metricbeat-6.5.4-2019.01.13	doc	fNSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.306Z	filesystem	system	
metricbeat-6.5.4-2019.01.13	doc	fdSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.306Z	filesystem	system	
metricbeat-6.5.4-2019.01.13	doc	ftSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.306Z	filesystem	system	
metricbeat-6.5.4-2019.01.13	doc	f9SLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.306Z	filesystem	system	
metricbeat-6.5.4-2019.01.13	doc	gNSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.306Z	filesystem	system	
metricbeat-6.5.4-2019.01.13	doc	gdSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.306Z	filesystem	system	
metricbeat-6.5.4-2019.01.13	doc	gtSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.306Z	filesystem	system	
metricbeat-6.5.4-2019.01.13	doc	g9SLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.306Z	filesystem	system	
metricbeat-6.5.4-2019.01.13	doc	hNSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.306Z	filesystem	system	
metricbeat-6.5.4-2019.01.13	doc	hdSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	cpu	system	
metricbeat-6.5.4-2019.01.13	doc	htSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	load	system	
metricbeat-6.5.4-2019.01.13	doc	h9SLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	memory	system	
metricbeat-6.5.4-2019.01.13	doc	iNSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	network	system	
metricbeat-6.5.4-2019.01.13	doc	idSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	network	system	
metricbeat-6.5.4-2019.01.13	doc	itSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	network	system	
metricbeat-6.5.4-2019.01.13	doc	i9SLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	network	system	
metricbeat-6.5.4-2019.01.13	doc	jNSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	network	system	
metricbeat-6.5.4-2019.01.13	doc	jdSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	network	system	
metricbeat-6.5.4-2019.01.13	doc	jtSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	network	system	
metricbeat-6.5.4-2019.01.13	doc	j9SLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	network	system	
metricbeat-6.5.4-2019.01.13	doc	kNSLRmgB8SxFt2LBLp1y	1	2019-01-13T09:29:11.307Z	network	system	

system module配置

```
- module: system
  period: 10s  # 采集的频率，每10秒采集一次
  metricsets:  # 采集的内容
    - cpu
    - load
    - memory
    - network
    - process
    - process_summary
```

Metricbeat Module

Metricbeat Module的用法和我们之前学的filebeat的用法差不多

```
#查看列表
./metricbeat modules list
```

能够看到对应的列表

```
Enabled:
system #默认启用

Disabled:
aerospike
apache
ceph
couchbase
docker
dropwizard
elasticsearch
envoyproxy
etcd
golang
graphite
haproxy
http
jolokia
kafka
kibana
kubernetes
kvm
logstash
memcached
mongodb
munin
mysql
nginx
php_fpm
postgresql
prometheus
rabbitmq
redis
traefik
uwsgi
vsphere
windows
```

Nginx Module

开启Nginx Module

在nginx中，需要开启状态查询，才能查询到指标数据。

```
#重新编译nginx
./configure --prefix=/usr/local/nginx --with-http_stub_status_module
make
make install

./nginx -V #查询版本信息
nginx version: nginx/1.11.6
built by gcc 4.4.7 20120313 (Red Hat 4.4.7-23) (GCC)
configure arguments: --prefix=/usr/local/nginx --with-http_stub_status_module

#配置nginx
vim nginx.conf
location /nginx-status {
    stub_status on;
    access_log off;
}

# 重启nginx
./nginx -s reload
```

测试

结果说明：

- Active connections：正在处理的活动连接数
- server accepts handled requests
  - 第一个 server 表示Nginx启动到现在共处理了9个连接
  - 第二个 accepts 表示Nginx启动到现在共成功创建 9 次握手
  - 第三个 handled requests 表示总共处理了 21 次请求
  - 请求丢失数 = 握手数 - 连接数，可以看出目前为止没有丢失请求
- Reading: 0 Writing: 1 Waiting: 1
  - Reading：Nginx 读取到客户端的 Header 信息数
  - Writing：Nginx 返回给客户端 Header 信息数
  - Waiting：Nginx 已经处理完正在等候下一次请求指令的驻留链接（开启keep-alive的情况下，这个值等于
    Active - (Reading+Writing))

配置nginx module

```
#启用redis module
./metricbeat modules enable nginx

#修改redis module配置
vim modules.d/nginx.yml
```

然后修改下面的信息

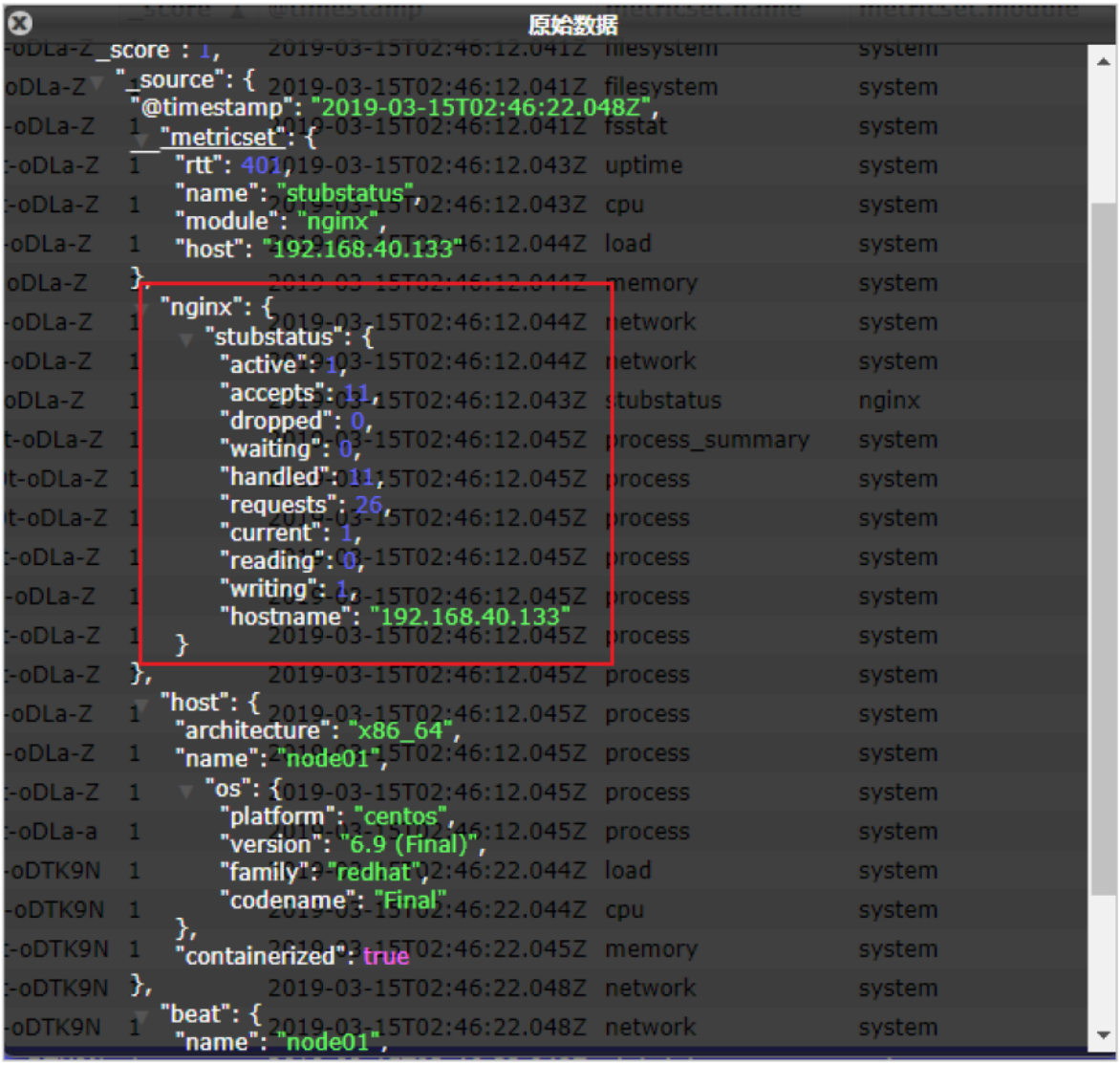
```
# Module: nginx
# Docs: https://www.elastic.co/guide/en/beats/metricbeat/6.5/metricbeat-
modulenginx.
html
  - module: nginx
#metricsets:
# - stubstatus
  period: 10s
# Nginx hosts
  hosts: ["http://127.0.0.1"]
# Path to server status. Default server-status
  server_status_path: "nginx-status"
#username: "user"
#password: "secret"
```

修改完成后，启动nginx

```
#启动
./metricbeat -e
```

测试

我们能看到，我们的nginx数据已经成功的采集到我们的系统中了



可以看到，nginx的指标数据已经写入到了Elasticsearch。

更多的Module使用参见官方文档：

https://www.elastic.co/guide/en/beats/metricbeat/current/metricbeat-modules.html

参考

Filebeat 模块与配置

Elastic Stack（ELK）从入门到实践