Nginx日志管理实践

本次实践内容为对nginx进行日志管理的全部操作步骤。

安装Nginx

这里直接对nginx进行简易的yum安装即可

```
# yum -y install nginx
```

修改Nginx配置文件，启动Nginx

基本不用修改什么内容，添加好server_name就差不多了

启动nginx服务

```
# systemctl start nginx
# systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; vendor preset:
disabled)
   Active: active (running) since Mon 2019-02-11 14:45:02 CST; 1h 27min ago
  Process: 6764 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
  Process: 6759 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
  Process: 6757 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited,
status=0/SUCCESS)
 Main PID: 6767 (nginx)
   CGroup: /system.slice/nginx.service
           ├─6767 nginx: master process /usr/sbin/nginx
           ├─6768 nginx: worker process
           └─6769 nginx: worker process
```

调试grok

调试grok是很关键的一步，如果没有能够调试成功，也就不能获取到我们需要的数据，需要根据不同的日志结构来修改grok pattern，才能得到我们真正需要的数据。

1. 先在nginx的日志中，截取一段日志用来进行调试。
2. 在kibana6.5版本中，Dev tools已经加入了grok debugger，可以直接在这里进行调试。
3. 以下为调试的日志、grok pattern和结果

   log：

```
14.153.187.107 - - [11/Feb/2019:14:45:59 +0800] "GET /favicon.ico HTTP/1.1" 404
3650 "-" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:64.0) Gecko/20100101
Firefox/64.0" "-"
```

grok pattern：

```
%{IPORHOST:clientip} - %{NOTSPACE:remote_user} \[%{HTTPDATE:timestamp}\] \"(?:%
{WORD:verb} %{NOTSPACE:request}(?: HTTP/%{NUMBER:httpversion})?|%
{DATA:rawrequest})\" %{NUMBER:response} (?:%{NUMBER:bytes}|-) \"-\" %{QS:agent} \"-
\"
```

result（Structured Data）：

```
{
  "request": "/favicon.ico",
  "agent": "\"Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:64.0) Gecko/20100101
Firefox/64.0\"",
  "verb": "GET",
  "remote_user": "-",
  "response": "404",
  "bytes": "3650",
  "clientip": "14.153.187.107",
  "httpversion": "1.1",
  "timestamp": "11/Feb/2019:14:45:59 +0800"
}
```

编写filebeat配置文件

```
# cat /usr/local/filebeat/filebeat.nginx.yml
filebeat.prospectors:
- type: log
  input_type: log
  paths:
    - /var/log/nginx/access.log
  tags: ["nginx"]
  fields:
    logIndex: nginx
    docType: nginx-access
  fields_under_root: true
  tail_files: false

output.logstash:
  hosts: ["127.0.0.1:5044"]
```

指定好input的类型，output到Logstash

编写Logstash配置文件

```
# cat /usr/local/logstash/config/nginx_access.conf
input {
    beats {
        port => 5044
    }
}

filter {
    grok {
        match => { "message" => "%{IPORHOST:clientip} - %{NOTSPACE:remote_user} \[%
{HTTPDATE:timestamp}\] \"(?:%{WORD:verb} %{NOTSPACE:request}(?: HTTP/%
{NUMBER:httpversion})?|%{DATA:rawrequest})\" %{NUMBER:response} (?:%{NUMBER:bytes}|-)
\"-\" %{QS:agent} \"%{DATA:x_forwarded_for}\"" }
        remove_field => "message"
    }
}

output {
    elasticsearch {
        hosts => ["127.0.0.1:9200"]
        index => "test-nginx-%{type}-%{+YYYY.MM.dd}"
        document_type => "%{type}"
    }
    stdout { codec => rubydebug }
}
```

input指定为来自beats的日志，filter中指定grok插件，match的添加好之前调试好的grok pattern，然后移除掉"message"，避免重复信息，output指定输出到elasticsearch中去。

output中的index为指定索引，这里是按每天的日志建立不同的索引。

启动filebeat

进入filebeat的安装目录，直接前台启动，方便测试

```
# cd /usr/local/filebeat/
# ./filebeat -c filebeat.nginx.yml
```

启动Logstash

指定配置文件为 filebeat.nginx.yml 启动Logstash，让Logstash直接在前台运行方便查看结果

```
# cd /usr/local/logstash/
# ./bin/logstash -f config/nginx_access.conf
```

再次访问nginx后可以看到Logstash这边生成了新的信息

查看elasticsearch是否生成了索引

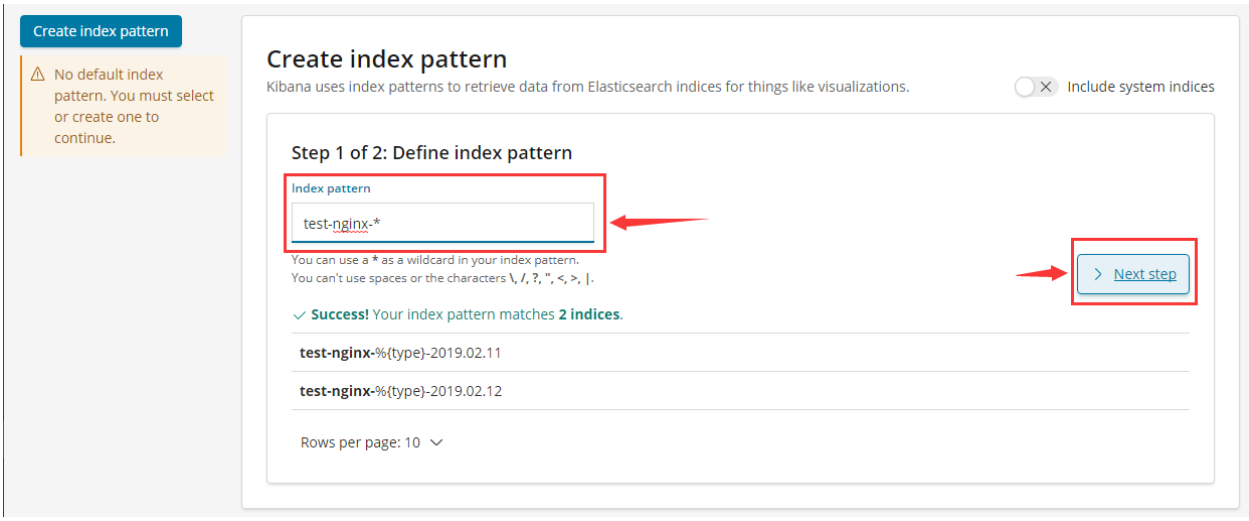
在使用客户端访问nginx后，会在服务器自动生成log，然后如果成功了的话，会在elasticsearch中按配置文件指定的格式自动生成索引，所以可以通过直接查看elasticsearch是否有自动生成索引来查看是否成功。

```
# curl http://127.0.0.1:9200/_cat/indices?v | grep test-nginx
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1270  100  1270    0     0  74486      0 --:--:-- --:--:-- --:--:-- 79375
yellow open   test-nginx-%{type}-2019.02.12   OJ-I1_naTlyOx43ARtryGQ   5   1          4
            0       460b           460b
yellow open   test-nginx-%{type}-2019.02.11   APn-Hh1qQXeWGiDL0DiK7Q   5   1         63
            0    485.1kb        485.1kb
```

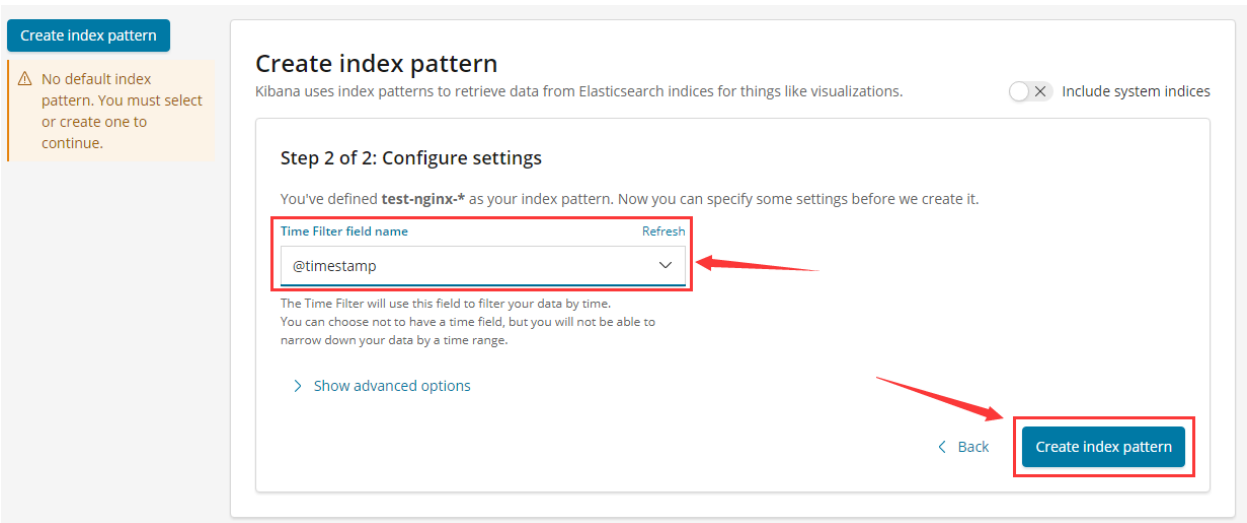
根据上述的查看结果，可以看到已经自动生成了test-nginx-%{type}-2019.02.11和 test-nginx-%{type}-2019.02.12这两个索引（P.S.今天为2019.02.12）

在kibana创建index

1. 打开kibana的web地址后，按顺序打开：`management`→`kibana`→`index pattern`→`create index pattern`
2. 在index pattern的匹配框里面填入`test-nginx-*`就可以匹配到所有以test-nginx-开头的index



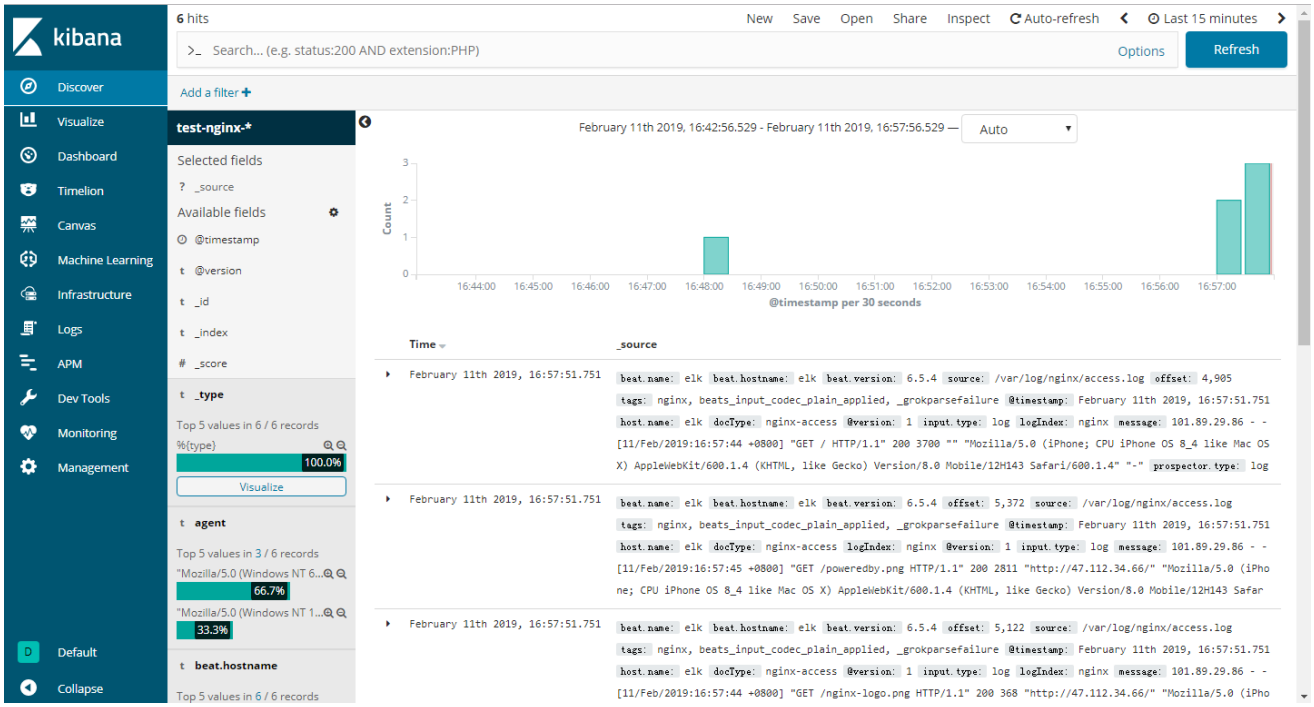
3. 选择@timestamp后创建



在kibana的discover查看结果

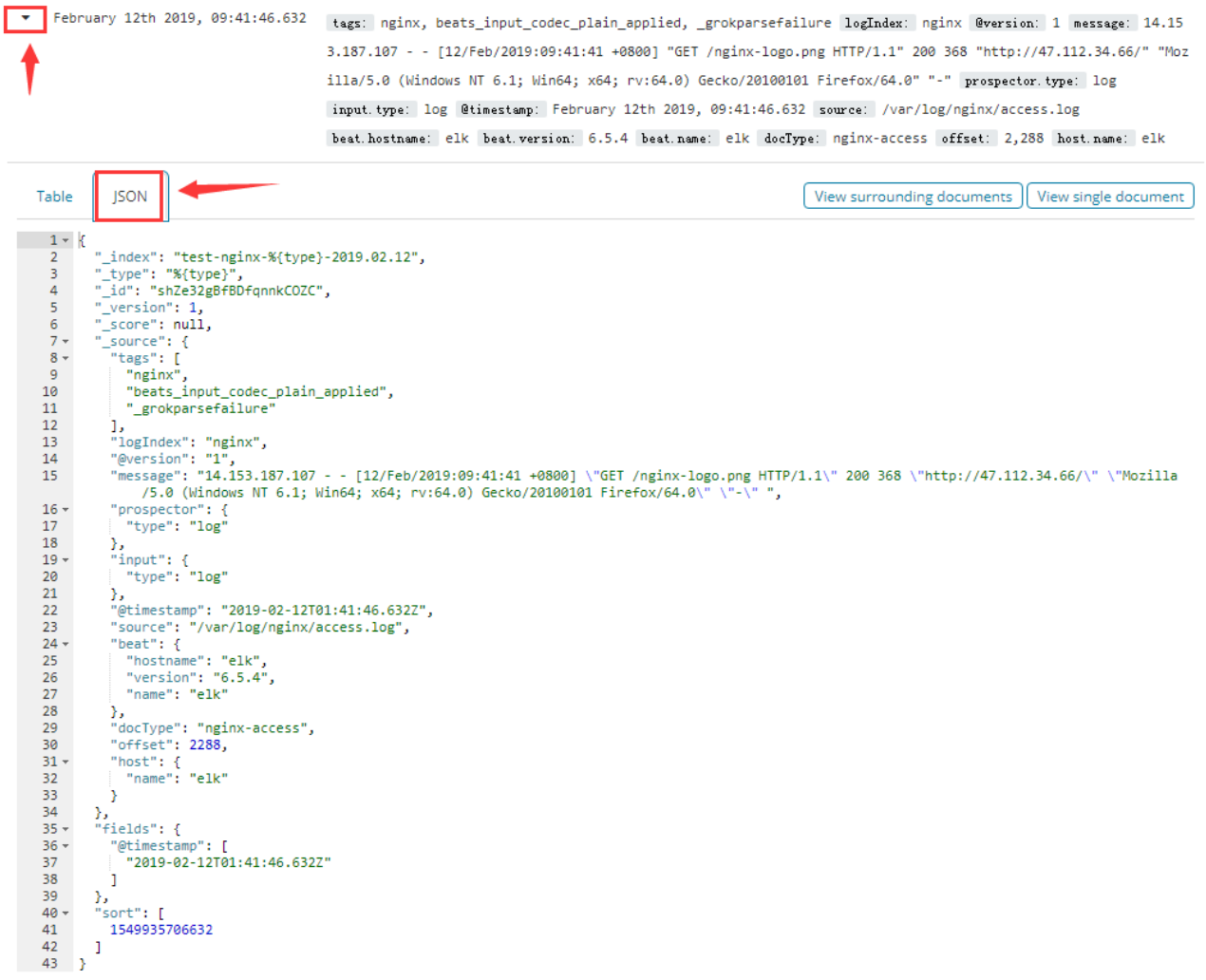
创建好之后就可以直接在kibana的discover中查看结果，能够直观的查看到不同时间产生的日志数量以及日志信息

如下图所示：



还能查看详细的字段：

也能查看json格式的结构化数据：


February 12th 2019, 09:41:46.632
tags: nginx, beats_input_codec_plain_applied, _grokparsefailure logIndex: nginx @version: 1 message: 14.153.187.107 - - [12/Feb/2019:09:41:41 +0800] "GET /nginx-logo.png HTTP/1.1" 200 368 "http://47.112.34.66/" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0" "-" prospector.type: log input.type: log @timestamp: February 12th 2019, 09:41:46.632 source: /var/log/nginx/access.log beat.hostname: elk beat.version: 6.5.4 beat.name: elk docType: nginx-access offset: 2,288 host.name: elk
Table
JSON
View surrounding documents
View single document
{
"_index": "test-nginx-%{type}-2019.02.12",
"_type": "%{type}",
"_id": "shZe32gBfBDfqnnkCOZC",
"_version": 1,
"_score": null,
"_source": {
"tags": [
"nginx",
"beats_input_codec_plain_applied",
"_grokparsefailure"
],
"logIndex": "nginx",
"@version": "1",
"message": "14.153.187.107 - - [12/Feb/2019:09:41:41 +0800] \"GET /nginx-logo.png HTTP/1.1\" 200 368 \"http://47.112.34.66/\" \"Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0\" \"-\" ",
"prospector": {
"type": "log"
},
"input": {
"type": "log"
},
"@timestamp": "2019-02-12T01:41:46.632Z",
"source": "/var/log/nginx/access.log",
"beat": {
"hostname": "elk",
"version": "6.5.4",
"name": "elk"
},
"docType": "nginx-access",
"offset": 2288,
"host": {
"name": "elk"
}
},
"fields": {
"@timestamp": [
"2019-02-12T01:41:46.632Z"
]
},
"sort": [
1549935706632
]
}