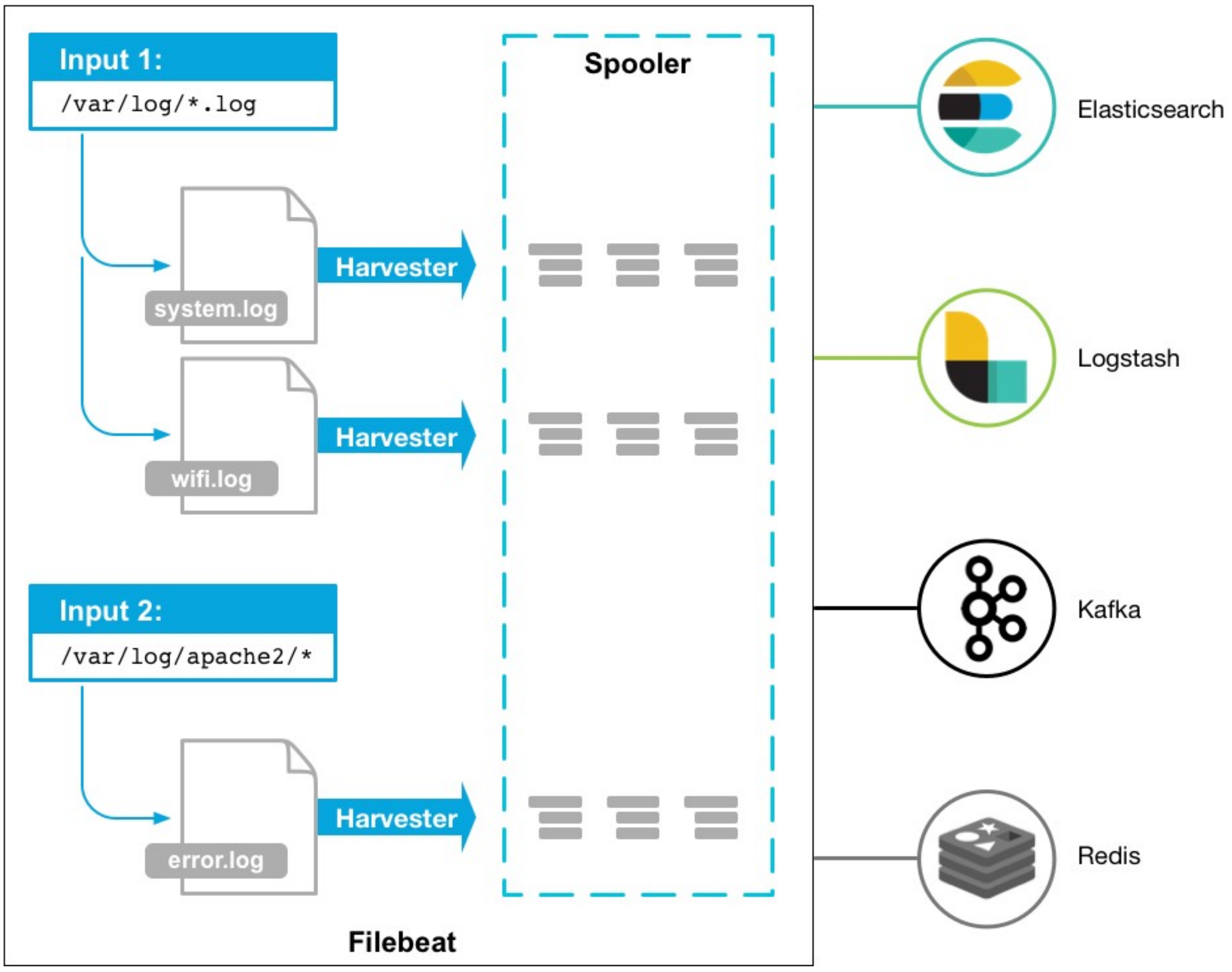
Fast ElasticSearch Development with Ingest pipeline and filebeat

● Filebeat Overview

Filebeat is a lightweight shipper for forwarding and centralizing log data. Installed as an agent on your servers, Filebeat monitors the log files or locations that you specify, collects log events, and forwards them either to Elasticsearch or Logstash for indexing.



● Install Filebeat

I. Install filebeat

1. https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-{{ version }}-linux-x86_64.tar.gz

2. extract tar.gz to your filebeat path

II. create Config directory in filebeat

1. ../filebeat-{{ version }}-linux-x86_64/CustomConfig

2. ../filebeat-{{ version }}-linux-x86_64/CustomConfig/inputs.d

III.create filebeat.yml in CustomConfig directory

```
filebeat.config.inputs:
   enabled: true
   path: ../filebeat-{{ version }}-linux-x86_64/CustomConfig/inputs.d/*.yml

output.elasticsearch:
   hosts: ["{{ elasticsearch_host:port}}"]
   username: "********"
   password: "********"

monitoring:
   enabled: true
   elasticsearch:
        hosts: ["{{ elasticsearch_host:port}}"]
   cluster_uuid: ********************************

setup.ilm.enabled: false
```

IV.create your project log yml

```
- type: log
   paths:
      -    {{ log_path }}
   fields_under_root: true
   scan_frequency: 3s
   idle_timeout: 3s
   pipeline: {{ pipeline_name }}

   processors:
      - dissect:
            tokenizer:
"[userId]:%{userId}[createdDate]:%{createdDate}[loginDate]:%{loginDate}[logoutDate]:%{logou
Date}[sessionId]:%{sessionId}[ip]:%{ip}[device]:%{device}[loginActionType]:%
{loginActionType}
[content]:%{content}"
            field: "message"
            target_prefix: ""
      - drop_event:
            when:
               has_fields: ["log.flags"]
```

● Filebeat pipeline Dissect

I. Dissect instructions and yml Important reminder

II. Yml Important reminder

1. Case sensitive
2. Use indentation to represent hierarchical relationships
3. Tabs are not allowed when indenting, only spaces are allowed
4. The number of indented spaces is not important, as long as elements of the same level are aligned to the left

III.Sample pipeline yml

```
- type: log
   paths:
      -   {{ log_path }}
   fields_under_root: true
   scan_frequency: 3s
   idle_timeout: 3s
   pipeline: {{ pipeline_name }}

   processors:
      - dissect:
            tokenizer:
"[userId]:%{userId}[createdDate]:%{createdDate}[loginDate]:%{loginDate}[logoutDate]:%
logoutDate}[sessionId]:%{sessionId}[ip]:%{ip}[device]:%{device}[loginActionType]:%
loginActionType} [content]:%{content}"
            field: "message"
            target_prefix: ""
      - drop_event:
            when:
               has_fields: ["log.flags"]
```

IV. Paths in pipeline yml

1. If there are multiple paths in Paths, it is recommended to separate different yml for future debugging and maintenance . The following is an example

Multiple paths

```
$ vi filebeatDissect.yml
- type: log
   paths:
       -   {{ log_path }}
       -   {{ log_path_1 }}
   fields_under_root: true
   scan_frequency: 3s
   idle_timeout: 3s
   pipeline: {{ pipeline_name }}
```

Separate yml

```
$ vi filebeatDissect.yml
- type: log
   paths:
       -  {{ log_path }}
   fields_under_root: true
   scan_frequency: 3s
   idle_timeout: 3s
   pipeline: awc_fullrecordapi

$ vi filebeatDissect_1.yml
- type: log
   paths:
       -   {{ log_path_1 }}
   fields_under_root: true
   scan_frequency: 3s
   idle_timeout: 3s
   pipeline: {{ pipeline_name }}
```

V. Dissect

1. One of the Processors used by Filebeat to cut logs

2. Dissect mainly cuts out the key through% {key_name}, and the corresponding content is the value of this key

3. Tips for cutting the log: do not need to cut the text or special characters in the log, please write it into the dissect processor

The following demonstrates a log dissect processor

Log

```
[2020/02/11    10:00:00.175][BaseService.java:2566][ERROR][http-nio-1213-exec-306]    [Status]:{"status":"1028"},[Agent_ID]:002,[Request]:{"Method":"POST","IP":"10.10.80.2"},[Response]:{"status":"1028"}
```

Dissect processor

```
[%{logdate}][%{class}][%{severity}][%{thread}][Status]:%{status},[Agent_ID]:%{agentId},[Request]:%{request},[Response]:%{response}
```

VI. Debugging procedure

1. First comment out the drop event of test pipeline yml

```
- type: log
   paths:
      -   {{ log_path }}
   fields_under_root: true
   scan_frequency: 3s
   idle_timeout: 3s
   pipeline: {{ pipeline_name }}

   processors:
      - dissect:
            tokenizer:
"[%{logdate}][%{class}][%{severity}][%{thread}] [Status]:%{Status},[Agent_ID]:%{Agent_ID},[Request]:%{Request},[Response]:%{Response}"
            field: "message"
            target_prefix: ""
#       - drop_event:
#            when:
#                has_fields: ["log.flags"]
```

2. Secondly, rewrite the path in test.yml to the pipeline yml path to be tested

```
filebeat.config.inputs:
   enabled: true
   path: ../test_log.yml
output.console:
   pretty: true
```

3. Finally use command ./filebeat -c CustomConfig/test.yml to see debug message if you see the message like following. it is mean your dissect not success cut log

```
{
   "@timestamp":  "2020-04-10T12:28:54.081Z",
   "@metadata":  {
      "beat":  "filebeat",
      "type":  "_doc",
      "version":  "7.3.2",
      "pipeline":  "ckf_manager"
   },
   "ecs":  {
      "version":  "1.0.1"
   },
   "message":  "kqpowkepowqkepoqkwepo12po21po,qwpd[2019/12/13  23:22:20.969][Match
BO.java.closeToSettled:693][INFO][QuartzScheduler_Worker-17]  [Function]:Change  Match  St
atus-Settle  Match(Match  ID=865445),[Action]:Edit,[Website]:null,[Previous  Value]:96,[Curren
t  Value]:288,[Update  By]:System",
   "log":  {
      "offset":  0,
      "file":  {
         "path":  "/var/data/log/ckf_manager/2.log"
      },
      "flags":  [
         "dissect_parsing_error"
      ]
   },
   "input":  {
      "type":  "log"
   },
```

VII. Version control

In order to enhance the synchronization of the Filebeat settings of each project and the Elasticsearch pipeline of the ITIG team. In the future, Filebeat will add the version number.

```
- type: log
   paths:
      -    {{ log_path }}
   fields:
      version: "1.0"
   fields_under_root: true
   scan_frequency: 3s
   idle_timeout: 3s
   pipeline: {{ pipeline_name }}
   processors:
      - dissect:
            tokenizer: "[%{logdate}][%{class}][%{severity}][%{thread}] [Status]:%{Status},
[Agent_ID]:%{Agent_ID},[Request]:%{Request},[Response]:%{Response}"
            field: "message"
            target_prefix: ""
      - drop_event:
            when:
               has_fields: ["log.flags"]
```

If each project needs to modify the filebeat config. Please be sure to inform the team to modify the version number to avoid the log from failing to parse

VIII. Log add field process

If project team need to add new fields to the log , please notify Polo first and then Polo will open the Jira demand list. After the team assessed that it was feasible. The log will be tested according to the environment dev-> stg-> prod order

1. The project team has the need to add new fields in the log

2. Make a request to Polo and evaluate it

3. After Polo assessment is feasible, open a demand list through Jira to the team for evaluation

4. After evaluation, the team will start testing with the project leader in the dev environment

5. After the test is completed, it will be updated to the prod environment

IX. Notice

1. Any spaces or special characters will affect whether the log can be cut correctly, please make sure the log format is unified

2. Please use the lower camel case to name the key

● log fields definition

When your dissect config is finish , you need definition fields type . e.g int , date etc. It's be cause when your filebeat send fields to elasticsearch , thich will auto definition fields type. This is a great setting to save your time , but also have some disadvantage . e.g if your field is int type but elasticsearch is definition to string type , you will not can use this field for calculation . So we need definition type first when log deposit to elasticsearch .

● ingest node

Use an ingest node to pre-process documents before the actual document indexing happens. The ingest node intercepts bulk and index requests, it applies transformations, and it then passes the documents back to the index or bulk APIs.

● Ingest pipeline

So we can use ingest pipeline to process log . e.g setting field type , setting date format , copy field , remove field , definition index name etc. Let's see the sample for ingest pipeline

```
{
   "description" : “ sample ingest pipeline",
   "processors" : [
     {
       "json" : {
         "field" : "accountstatement",
         "target_field" : "accountstatement",
         "on_failure" : [
           {
             "drop" : { }
           }
         ]
       }
     },
     {
       "date" : {
         "field" : "logdate",
         "target_field" : "logdate",
         "formats" : [
           "yyyy/MM/dd HH:mm:ss.SSS"
         ],
         "timezone" : "UTC+8"
       }
     },
     {
       "set" : {
         "field" : "@timestamp",
         "value" : "{{accountstatement.createTime}}"
       }
     },
     {
       "remove" : {
         "field" : ["agent","class","ecs","log","severity","thread","message","input"]
       }
     },
     {
       "date_index_name" : {
         "field" : "logdate",
         "date_rounding" : "d",
         "date_formats" : [
           "ISO8601"
         ],
         "index_name_prefix" : "team-project-",
         "index_name_format" : "yyyy.MM.dd"
       }
     }
   ]
  }
```

● Index template

An index template is a way to tell Elasticsearch how to configure an index when it is created. You can use template to definition index mapping , setting , aliases. Let's see the sample template.

```
{
 "order": 0,
 "index_patterns": [
  "index_name-*"
 ],
 "settings": {
  "index": {
   "lifecycle": {
    "name": "your_lifecycle_name"
   },
   "sort": {
    "field": "logdate",
    "order": "desc"
   },
   "number_of_shards": "2",
   "number_of_replicas": "1"
  }
 },
 "mappings": {
  "dynamic": "strict",
  "properties": {
   "@timestamp": {
    "type": "date"
   },
   "accountstatement": {
    "properties": {
     "amount": {
      "type": "long"
     },
     "beforeBalance": {
      "type": "float"
     },
   "logdate": {
    "type": "date"
   },
   "host": {
    "type": "text",
    "fields": {
     "keyword": {
      "ignore_above": 256,
      "type": "keyword"
     }
    }
   }
  }
 },
 "aliases": {
  "aliases_name": {}
 }
}
```

Attention! Template is definition you index all fields type. So please make sure all of your index fields type is defined. If you have one field which not defined in template then the log which have this field will not be allow into Elasticsearch.

● Index lifecycle management (ILM)

You can configure index lifecycle management (ILM) policies to automatically manage indices according to your performance, resiliency, and retention requirements. We will use ILM for management index lifetime to control cluster disk use size. Let's see the sample ILM .

```
{
  "policy": {
    "phases": {
      "hot": {
        "min_age": "0ms",
        "actions": {
          "set_priority": {
            "priority": 100
          }
        }
      },
      "delete": {
        "min_age": "7d",
        "actions": {
          "delete": {}
        }
      }
    }
  }
}
```