elastic

Community Conference 2021

Shell Script 로 elasticsearch 운영 자동화하기

YongKyun Kang
godway1225@gmail.com

About Me

- I am interested in dealing with big data
- Programmer based on Security, ESM(SIEM) Server at AhnLab, Inc
- Blog : https://godway1225.wordpress.com

Elasticsearch 운영을 위한 작업들

Elasticsearch 운영을 위한 작업들

- Daily Job
  - 내일의 인덱스 생성하기
  - 오래된 인덱스 삭제하기
  - 어제의 인덱스 forcemerge 하기
  - Cluster 간 인덱스 백업하기
- Occasional Job
  - Unassigned 된 샤드 처리하기
  - Close 된 인덱스 순차적으로 open 하기
  - Primary shard 균등하게 rebalance 하기

내일의 인덱스 생성하기

내일의 인덱스 생성하기

생성할 인덱스명 : filebeat-6.8.13-2021.02.28

INDEX_DATE=`date -d tomorrow "+%Y.%m.%d"`

MacOS : INDEX_DATE=`date -v+1d "+%Y.%m.%d"`

ELASTIC_ADDR=127.0.0.1:9200

INDEX_PREFIX=filebeat-6.8.13-

curl -XPUT "http://$ELASTIC_ADDR/$INDEX_PREFIX$INDEX_DATE"

(ex) curl -XPUT "http://127.0.0.1:9200/filebeat-6.8.13-2021.02.28"

crontab 등록

0 22 * * * /usr/local/elasticsearch/util/create_index.sh

Create index API

Creates a new index.

```
PUT /my-index-000001
```

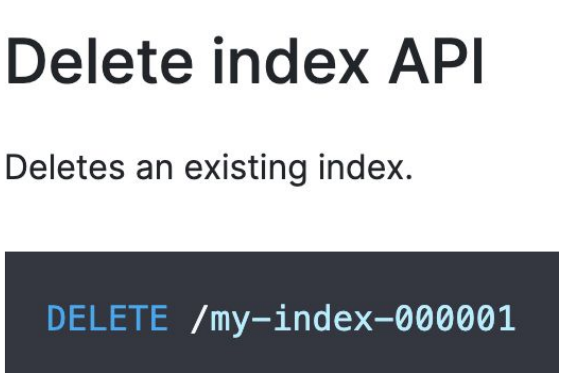
오래된 인덱스 삭제하기

오래된 인덱스 삭제하기

100일전 인덱스 삭제하기

INDEX_DATE=`date +%Y.%m.%d --date '100 day ago'`

MacOS : INDEX_DATE=`date -v-100d "+%Y.%m.%d"`



curl -XDELETE "http://$ELASTIC_ADDR/$INDEX_PREFIX$INDEX_DATE"

(ex) curl -XDELETE "http://127.0.0.1:9200/filebeat-6.8.13-2020.11.10

crontab 등록

0 23 * * * /usr/local/elasticsearch/util/delete_index.sh

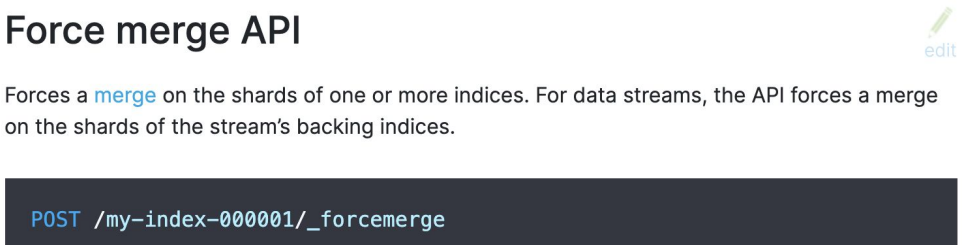
어제의 인덱스 **forcemerge** 하기

어제의 인덱스 forcemerge 하기

INDEX_DATE=`date -d yesterday "+%Y.%m.%d"`

MacOS : INDEX_DATE=`date -v-1d "+%Y.%m.%d"`

NUM_SEGMENT=10



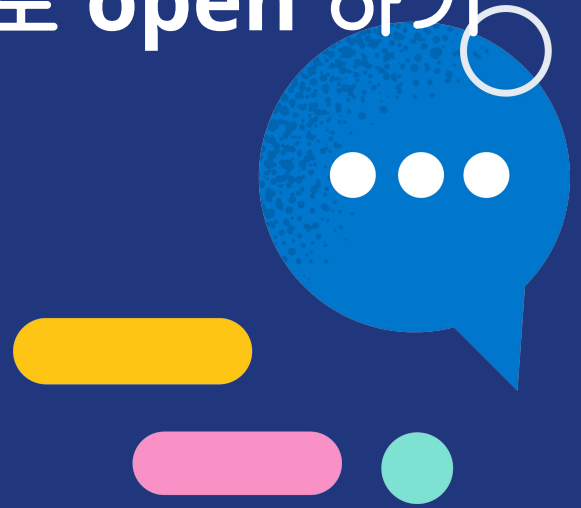
curl -XPOST "http://$ELASTIC_ADDR/$INDEX_PREFIX$INDEX_DATE/_forcemerge?max_num_segments=$NUM_SEGMENT"

(ex) curl -XPOST "http://127.0.0.1:9200/filebeat-6.8.13-2021.02.17/_forcemerge?max_num_segments=10

crontab 등록

30 0 * * * /usr/local/elasticsearch/util/forcemerge_index.sh

Close 된 인덱스 순차적으로 open 하기



Close 된 인덱스 순차적으로 open 하기

- 인자로 받은 월의 인덱스에서 close 된 인덱스를 날짜 역순으로 구한다
- 인덱스를 open 하는 중에 rebalance가 이루어지지 않도록 한다
- Close 된 날짜 역순의 리스트를 이용하여 하나씩 인덱스명을 가져온다
  - 해당 인덱스를 open 한다
  - 해당 인덱스가 open 되었는지 반복해서 확인한다
- 인덱스를 rebalance가 이루어지도록 변경한다

Close 된 인덱스 순차적으로 open 하기

- 사용하는 elasticsearch REST API
  - _cat/indices
  - _cluster/settings
  - _open
  - _cat/shards

- 사용하는 linux command
  - grep
  - awk
  - sort
  - wc
  - cut

Close 된 인덱스 순차적으로 open 하기

- 주어진 달의 인덱스에서 close 된 인덱스를 날짜 역순으로 구한다
  - INDEX_MONTH=`date +"%Y.%m"`
  - URL_SEND_CMD="curl -s"
  - INDEX_DATE_LIST=`$URL_SEND_CMD -XGET $ELASTIC_ADDR/_cat/indices | grep close | grep $INDEX_PREFIX$MONTH | awk -F "$INDEX_PREFIX" '{print $2}' | awk '{print $1}' | sort -r`

Close 된 인덱스 순차적으로 open 하기

- 주어진 달의 인덱스에서 close 된 인덱스를 날짜 역순으로 구한다

```
gangyoncBookPro:util root# curl -s -XGET 127.0.0.1:9200/_cat/in
dices | grep close | grep $INDEX_PREFIX$MONTH
 close filebeat-6.8.13-2021.02.14 ey_VGogzQES9hKl2DcOHLw
 close filebeat-6.8.13-2021.02.08 D2Axf6XOQO6VelBaGX0p-A
 close filebeat-6.8.13-2021.02.10 vZgGVA9IQ86tVuAfu0UZxg
 close filebeat-6.8.13-2021.02.12 YQW_VWedTWeWzaecQ6gpvA
 close filebeat-6.8.13-2021.02.13 dSnKUwPnRYOn_FXXTihHSg
 close filebeat-6.8.13-2021.02.02 X4CpdEBgQkyAIuh9oboZwQ
 close filebeat-6.8.13-2021.02.09 MFbfxf6fTyGPdrQrH3sIQg
 close filebeat-6.8.13-2021.02.07 Qu9S5YpYS1W-bEhNV6bdWA
gangyoncBookPro:util root# curl -s -XGET 127.0.0.1:9200/_cat/in
dices | grep close | grep $INDEX_PREFIX$MONTH | awk -F "$INDEX_
PREFIX" '{print $2}'
2021.02.14 ey_VGogzQES9hKl2DcOHLw
2021.02.08 D2Axf6XOQO6VelBaGX0p-A
2021.02.10 vZgGVA9IQ86tVuAfu0UZxg
2021.02.12 YQW_VWedTWeWzaecQ6gpvA
2021.02.13 dSnKUwPnRYOn_FXXTihHSg
2021.02.02 X4CpdEBgQkyAIuh9oboZwQ
2021.02.09 MFbfxf6fTyGPdrQrH3sIQg
2021.02.07 Qu9S5YpYS1W-bEhNV6bdWA
```

```
gangyoncBookPro:util root# curl -s -XGET 127.0.0.1:9200/_cat/indi
ces | grep close | grep $INDEX_PREFIX$MONTH | awk -F "$INDEX_PREF
IX" '{print $2}' | awk '{print $1}'
2021.02.14
2021.02.08
2021.02.10
2021.02.12
2021.02.13
2021.02.02
2021.02.09
2021.02.07
gangyoncBookPro:util root# curl -s -XGET 127.0.0.1:9200/_cat/indi
ces | grep close | grep $INDEX_PREFIX$MONTH | awk -F "$INDEX_PREF
IX" '{print $2}' | awk '{print $1}' | sort -r
2021.02.14
2021.02.13
2021.02.12
2021.02.10
2021.02.09
2021.02.08
2021.02.07
2021.02.02
gangyoncBookPro:util root# #
```

Close 된 인덱스 순차적으로 open 하기

- 인덱스를 open 하는 중에 rebalance가 이루어지지 않도록 한다
  - DATA_FILE=cmd.dat
  - echo "{ \"transient\" : { \"cluster.routing.rebalance.enable\" : \"none\" }}" > $DATA_FILE
  - $URL_SEND_CMD -XPUT -H 'Content-Type: application/json' $ELASTIC_ADDR/_cluster/settings --data-binary @$DATA_FILE
- 인덱스를 rebalance가 이루어지도록 변경한다
  - echo "{ \"transient\" : { \"cluster.routing.rebalance.enable\" : \"all\" } }" > $DATA_FILE

Close 된 인덱스 순차적으로 open 하기

- Close 된 날짜 역순의 리스트를 이용하여 하나씩 인덱스명을 가져온다
  - 해당 인덱스를 open 한다
  - 해당 인덱스가 open 되었는지 반복해서 확인한다

```
for INDEX_DATE in $INDEX_DATE_LIST
do
    INDEX_NAME="$INDEX_PREFIX$INDEX_DATE"
    DATE=`date +"%Y%m%d_%H%M%S"`
    echo "$DATE [$INDEX_NAME] index is opening"
    $URL_SEND_CMD -XPOST http://$ELASTIC_ADDR/$INDEX_NAME/_open
    echo ""
    while [ 1 ]
    do
        CNT=`$URL_SEND_CMD -s -XGET $ELASTIC_ADDR/_cat/shards/$INDEX_NAME | grep -v STARTED | wc | cut -c7-8`
        DATE=`date +"%Y%m%d_%H%M%S"`
        echo "$DATE CNT : $CNT"
        if [ $CNT -eq 0 ]
        then
            DATE=`date +"%Y%m%d_%H%M%S"`
            echo "$DATE [$INDEX_NAME] index is opened"
            break
        fi
        sleep 5
    done
done
```

Close 된 인덱스 순차적으로 open 하기

- Close 된 날짜 역순의 리스트를 이용하여 하나씩 인덱스명을 가져온다
  - 해당 인덱스가 open 되었는지 반복해서 확인한다

```
gangyoncBookPro:util root# INDEX_NAME=filebeat-6.8.13-2021.02.09
gangyoncBookPro:util root# $URL_SEND_CMD -XGET $ELASTIC_ADDR/_cat/shards/$INDEX_NAME
filebeat-6.8.13-2021.02.09 6 r STARTED 33435 12.4mb 127.0.0.1 node-3
filebeat-6.8.13-2021.02.09 6 p STARTED 33435 12.4mb 127.0.0.1 node-2
filebeat-6.8.13-2021.02.09 2 p STARTED 33652 12.5mb 127.0.0.1 node-3
filebeat-6.8.13-2021.02.09 2 r STARTED 33652 12.5mb 127.0.0.1 node-2
filebeat-6.8.13-2021.02.09 5 p STARTED 33699 12.6mb 127.0.0.1 node-1
filebeat-6.8.13-2021.02.09 5 r STARTED 33699 12.6mb 127.0.0.1 node-4
filebeat-6.8.13-2021.02.09 3 p STARTED 33814 12.7mb 127.0.0.1 node-1
filebeat-6.8.13-2021.02.09 3 r STARTED 33814 12.5mb 127.0.0.1 node-2
filebeat-6.8.13-2021.02.09 4 r STARTED 33783 12.5mb 127.0.0.1 node-3
filebeat-6.8.13-2021.02.09 4 p STARTED 33783 12.5mb 127.0.0.1 node-2
filebeat-6.8.13-2021.02.09 1 p STARTED 33775 12.6mb 127.0.0.1 node-1
filebeat-6.8.13-2021.02.09 1 r STARTED 33775 12.6mb 127.0.0.1 node-4
filebeat-6.8.13-2021.02.09 7 p STARTED 34123 12.3mb 127.0.0.1 node-1
filebeat-6.8.13-2021.02.09 7 r STARTED 34123 12.3mb 127.0.0.1 node-4
filebeat-6.8.13-2021.02.09 0 p STARTED 33719 12.5mb 127.0.0.1 node-3
filebeat-6.8.13-2021.02.09 0 r STARTED 33719 12.5mb 127.0.0.1 node-4
gangyoncBookPro:util root# $URL_SEND_CMD -XGET $ELASTIC_ADDR/_cat/shards/$INDEX_NAME | grep -v STARTED
gangyoncBookPro:util root# $URL_SEND_CMD -XGET $ELASTIC_ADDR/_cat/shards/$INDEX_NAME | grep -v STARTED | wc
       0       0       0
gangyoncBookPro:util root# $URL_SEND_CMD -XGET $ELASTIC_ADDR/_cat/shards/$INDEX_NAME | grep -v STARTED | wc | cut -c7-8
 0
gangyoncBookPro:util root#
```

Cluster 간 인덱스 백업하기

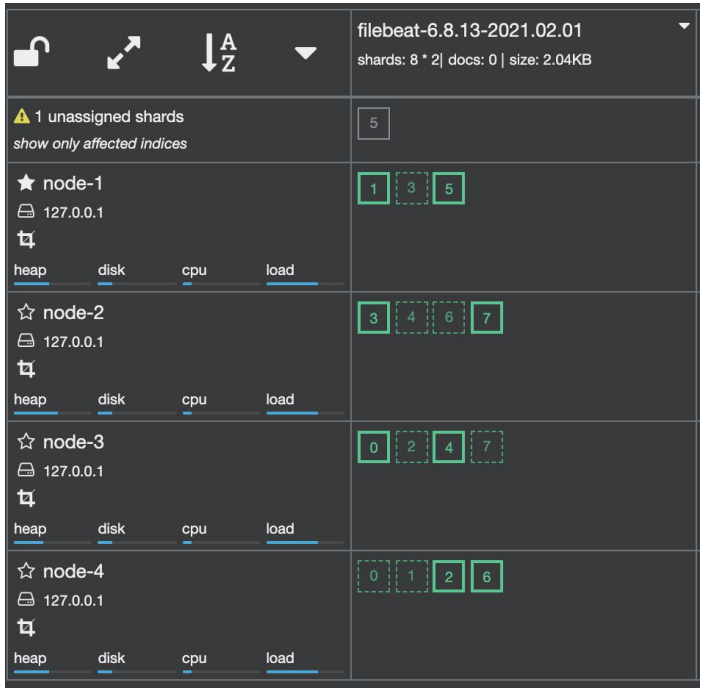
Cluster 간 인덱스 백업하기

- Warm Cluster 에 백업할 인덱스가 이미 존재하는지 확인
- Hot Cluster 에 백업할 인덱스가 존재하는지 확인
- Repository 에 백업할 인덱스와 동일한 명으로 백업할 인덱스를 포함하는 snapshot 생성
  - snapshot 생성 결과 확인 : IN_PROGRESS, SUCCESS, PARTIAL, FAILED
  - 결과가 PARTIAL 인 경우 생성했던 snapshot 삭제하고 백업 종료
  - 결과가 FAILED 인 경우 백업 종료
- snapshot 생성에 성공하였으면 Warm Cluster 에 Restore 시작
  - Warm Cluster 에 Restore 가 완료되었는지 확인
  - 진행중이면 대기
- Warm Cluster 에 Restore 가 성공적으로 이루어졌으면
  - 생성했던 snapshot 삭제
  - Hot Cluster 에서 백업된 인덱스 삭제(옵션)

Unassigned 된 샤드 처리하기

Unassigned 된 샤드 처리하기

- Unassigned 된 샤드가 존재하는지 확인한다.
  - unassigned 된 shard 가 존재하고 나머지 모든 shard 가 정상적으로 구동중인지 확인한다. (인덱스가 recovery 중이거나 하는 경우에는 처리하지 않기 위한 확인)
- 노드당 균등하게 가질 수 있는 shard 개수를 구한다
- shard 개수를 부족하게 가지고 있는 노드를 찾는다
  - 해당 노드가 가지고 있는 shard 리스트를 구한다.
  - 구한 shard 리스트에 존재하지 않는 shard 를 임의로 하나 선정한다.
  - 위에서 찾은 임의의 shard 의 replica 를 소유한 노드를 찾는다.
  - 위에서 찾은 replica shard 를 shard 개수가 부족한 노드로 reroute 시킨다



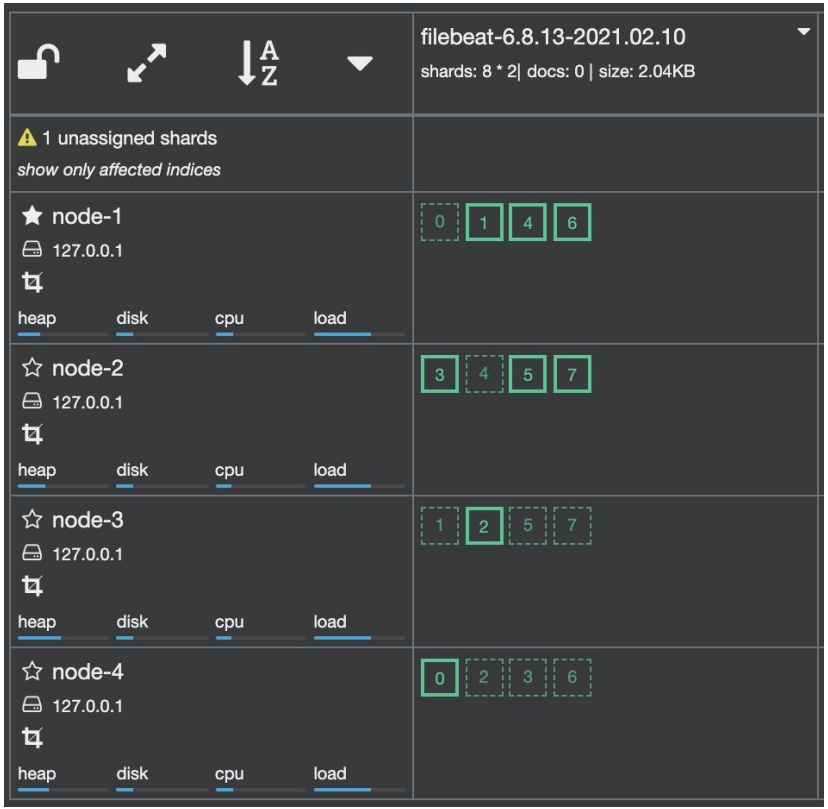
Primary shard 균등하게 rebalance 하기

Primary shard 균등하게 rebalance 하기

```
_cluster/reroute
{
   "commands": [
      {
         "cancel": {
            "allow_primary": true,
            "index": "filebeat-6.8.13-2021.02.10",
            "node": "node-1",
            "shard": 0
         }
      }
   ]
}
```

Primary shard 균등하게 rebalance 하기

- Node 에 primary shard 개수가 replica shard 개수보다 많은가?
  - Primary shard 의 replica shard 를 가진 node 의 primary shard 개수가 replica shard 개수보다 적은가?
- Node 에 primary shard 개수가 replica shard 개수보다 적은가?
  - Replica shard 의 primary shard 를 가진 node 의 primary shard 개수가 replica shard 개수보다 많은가?



Example code

https://github.com/yongkyun/elasticsearch-operation-utils

내일의 인덱스 생성하기

```
[gangyoncBookPro:util root# cat create_index.sh
#!/bin/bash
INDEX_DATE=`date -v+1d +"%Y.%m.%d"`
ELASTIC_ADDR=127.0.0.1:9200
INDEX_PREFIX=filebeat-6.8.13-

if [ "$#" -eq 1 ]; then
  INDEX_DATE="$1"
fi

curl -XPUT "http://$ELASTIC_ADDR/$INDEX_PREFIX$INDEX_DATE"
echo ""
[gangyoncBookPro:util root# ./create_index.sh
{"acknowledged":true,"shards_acknowledged":true,"index":"filebeat-6.8.13-2021.02.21"}
```

오래된 인덱스 삭제하기

```
[gangyoncBookPro:util root# cat delete_index.sh
#!/bin/bash
INDEX_DATE=`date -v-100d +"%Y.%m.%d"`
ELASTIC_ADDR=127.0.0.1:9200
INDEX_PREFIX=filebeat-6.8.13-

if [ "$#" -eq 1 ]; then
  INDEX_DATE="$1"
fi

echo "$INDEX_PREFIX$INDEX_DATE will be deleted"
curl -XDELETE "http://$ELASTIC_ADDR/$INDEX_PREFIX$INDEX_DATE" | python -m json.tool
echo ""
[gangyoncBookPro:util root# ./delete_index.sh
filebeat-6.8.13-2020.11.12 will be deleted
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    21  100    21    0     0     61      0 --:--:-- --:--:-- --:--:--    61
{
    "acknowledged": true
}
```

어제의 인덱스 forcemerge 하기

```
[gangyoncBookPro:util root# cat forcemerge_index.sh
#!/bin/bash
INDEX_DATE=`date -v-1d +"%Y.%m.%d"`
ELASTIC_ADDR=127.0.0.1:9200
INDEX_PREFIX=filebeat-6.8.13-
NUM_SEGMENT=10

if [ "$#" -eq 1 ]; then
  INDEX_DATE="$1"
fi

echo "$INDEX_PREFIX$INDEX_DATE will be forcemerged"
curl -XPOST "http://$ELASTIC_ADDR/$INDEX_PREFIX$INDEX_DATE/_forcemerge?max_num_segments=$NUM_SEGMENT" | python -m json.tool
echo ""
[gangyoncBookPro:util root# ./forcemerge_index.sh
filebeat-6.8.13-2021.02.19 will be forcemerged
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    51  100    51    0     0   3400      0 --:--:-- --:--:-- --:--:--  3400
{
    "_shards": {
        "failed": 0,
        "successful": 16,
        "total": 16
    }
}
```

Cluster 간 인덱스 백업하기

- 사용하는 elasticsearch REST API
  - Indices exist : HEAD index_name
  - _cat/indices
  - snapshot and restore : _snapshot
  - _cat/shards
  - Update Indices Settings : index_name/_settings
  - Delete index : DELETE index_name
- 사용하는 linux command
  - grep
  - awk
  - sort
  - wc
  - cut

Cluster 간 인덱스 백업하기

- Warm Cluster 에 백업할 인덱스가 이미 존재하는지 확인

```
########## check index at backup cluster ##########
INDEX_CHECK_EXIST=`$URL_SEND_CMD -I -XHEAD http://$BACKUP_ELASTIC/$INDEX_NAME | grep HTTP | awk {'print $2'}`
if [ "$INDEX_CHECK_EXIST" == "200" ]
then
        DATE=`date +"%Y%m%d_%H%M%S"`
        echo "$DATE $INDEX_NAME is already exist at backup elasticsearch server($BACKUP_ELASTIC)" >> $LOGFILE
        exit
fi
```

```
gangyoncBookPro:util root# $URL_SEND_CMD -I -XHEAD http://$BACKUP_ELASTIC/$INDEX_NAME
HTTP/1.1 404 Not Found
content-type: application/json; charset=UTF-8
content-length: 427

gangyoncBookPro:util root# $URL_SEND_CMD -I -XHEAD http://$BACKUP_ELASTIC/$INDEX_NAME | grep HTTP
HTTP/1.1 404 Not Found
gangyoncBookPro:util root# $URL_SEND_CMD -I -XHEAD http://$BACKUP_ELASTIC/$INDEX_NAME | grep HTTP | awk {'print $2'}
404
```

Cluster 간 인덱스 백업하기

- Hot Cluster 에 백업할 인덱스가 존재하는지 확인

```
########## check index at indexing cluster ##########
INDEX_CHECK_EXIST=`$URL_SEND_CMD -I -XHEAD http://$CUR_ELASTIC/$INDEX_NAME | grep HTTP | awk {'print $2'}`
if [ "$INDEX_CHECK_EXIST" != "200" ]
then
        DATE=`date +"%Y%m%d_%H%M%S"`
        echo "$DATE $INDEX_NAME is not exist at current elasticsearch server($CUR_ELASTIC)" >> $LOGFILE
        exit
fi
```

```
[gangyoncBookPro:util root# $URL_SEND_CMD -I -XHEAD http://$CUR_ELASTIC/$INDEX_NAME
HTTP/1.1 200 OK
content-type: application/json; charset=UTF-8
content-length: 67651

[gangyoncBookPro:util root# $URL_SEND_CMD -I -XHEAD http://$CUR_ELASTIC/$INDEX_NAME | grep HTTP
HTTP/1.1 200 OK
[gangyoncBookPro:util root# $URL_SEND_CMD -I -XHEAD http://$CUR_ELASTIC/$INDEX_NAME | grep HTTP | awk {'print $2'}
200
```

Cluster 간 인덱스 백업하기

- Repository 에 백업할 인덱스와 동일한 명으로 백업할 인덱스를 포함하는

```
########## make snapshut ##########
echo "{ \"indices\": \"$INDEX_NAME\", \"ignore_unavailable\": \"true\", \"include_global_state\": \"false\" }" > $
DATA_FILE
RESULT=`$URL_SEND_CMD -XPUT -H 'Content-Type: application/json' http://$CUR_ELASTIC/_snapshot/$REPOSITORY_NAME/$IN
DEX_NAME --data-binary @$DATA_FILE`
rm -f $DATA_FILE

if [ "$RESULT" != "{\"accepted\":true}" ]
then
        DATE=`date +"%Y%m%d_%H%M%S"`
        echo "$DATE Fail to make $INDEX_NAME's snapshut : $RESULT" >> $LOGFILE
        exit
fi
```

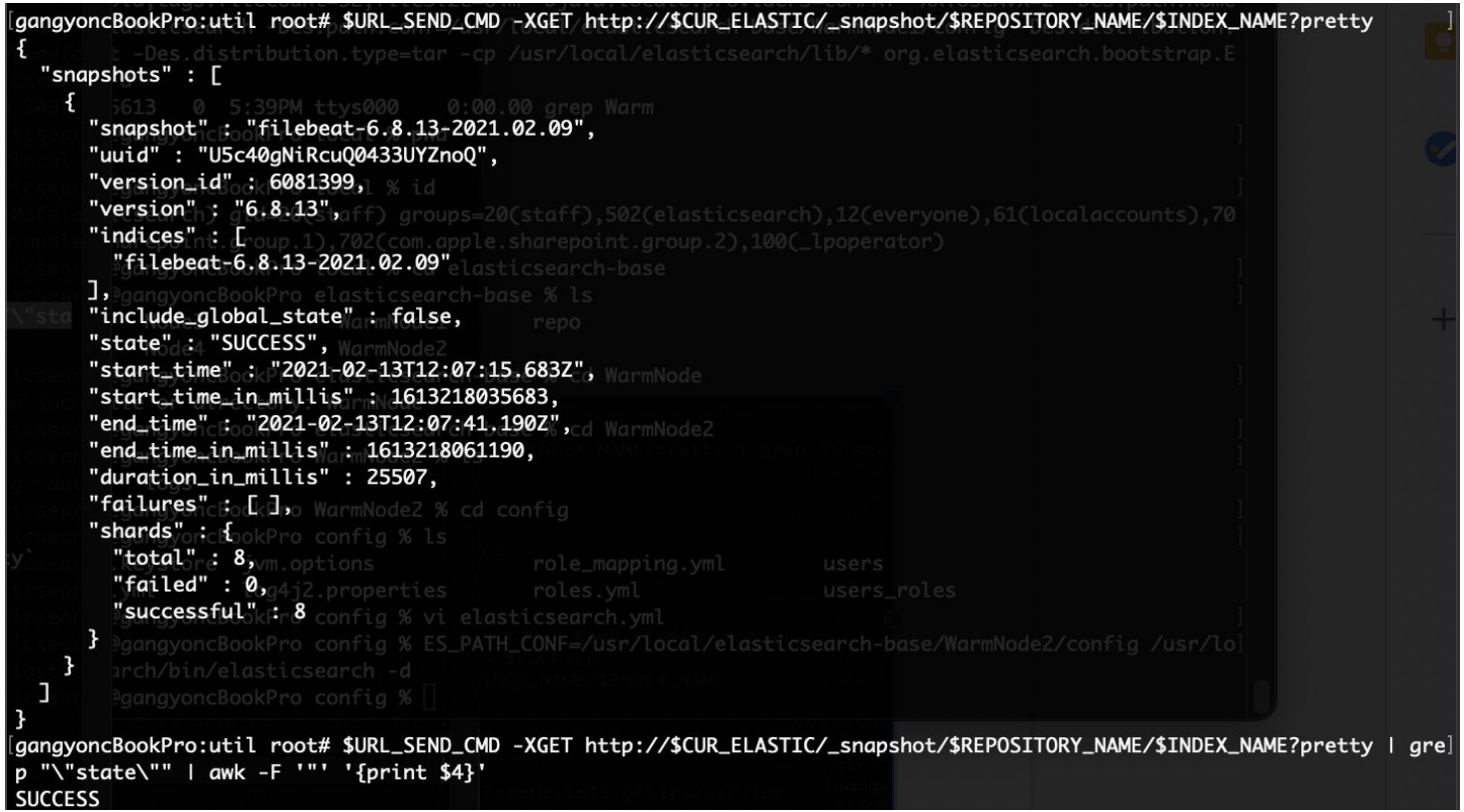
```
gangyoncBookPro:util root# echo "{ \"indices\": \"$INDEX_NAME\", \"ignore_unavailable\": \"true\", \"include_global_st
ate\": \"false\" }" > $DATA_FILE
gangyoncBookPro:util root# $URL_SEND_CMD -XPUT -H 'Content-Type: application/json' http://$CUR_ELASTIC/_snapshot/$REPO
SITORY_NAME/$INDEX_NAME --data-binary @$DATA_FILE
{"accepted":true}gangyoncBookPro:util root#
```

Cluster 간 인덱스 백업하기

- Repository 에 백업할 인덱스와 동일한 명으로 백업할 인덱스를 포함하는 snapshot 생성
  - snapshot 생성 결과 확인 : SUCCESS, PARTIAL, FAILED, IN_PROGRESS
  - 결과가 PARTIAL 인 경우 생성했던 snapshot 삭제하고 백업 종료

```
while [ 1 ]
do
        STATE=`$URL_SEND_CMD -XGET http://$CUR_ELASTIC/_snapshot/$REPOSITORY_NAME/$INDEX_NAME?pretty | grep "\"sta
te\"" | awk -F '"' '{print $4}'`
        DATE=`date +"%Y%m%d_%H%M%S"`
        if [ "$STATE" == "SUCCESS" ]
        then
                echo "$DATE $INDEX_NAME's snapshot making is succeed" >> $LOGFILE
                break;
        elif [ "$STATE" == "PARTIAL" ] || [ "$STATE" == "FAILED" ]
        then
                ERR_MSG=`$URL_SEND_CMD -XGET http://$CUR_ELASTIC/_snapshot/$REPOSITORY_NAME/$INDEX_NAME?pretty`
                echo "$DATE $INDEX_NAME's snapshot making is fail : $ERR_MSG" >> $LOGFILE
                $URL_SEND_CMD -XDELETE http://$BACKUP_ELASTIC/_snapshot/$REPOSITORY_NAME/$INDEX_NAME
                DATE=`date +"%Y%m%d_%H%M%S"`
                echo "$DATE $INDEX_NAME's snapshot is deleted" >> $LOGFILE
                exit
        fi
        echo "$DATE $INDEX_NAME $STATE"
        sleep 5
done
```

Cluster 간 인덱스 백업하기



Cluster 간 인덱스 백업하기

- snapshot 생성에 성공하였으면 Warm Cluster 에 Restore 시작

```
echo "{ \"indices\": \"$INDEX_NAME\", \"index_settings\": { \"index.number_of_replicas\": $BACKUP_ELASTIC_NUM_REPL
ICA }, \"ignore_index_settings\": [ \"index.refresh_interval\" ] }" > $DATA_FILE
RESULT=`$URL_SEND_CMD -XPOST -H 'Content-Type: application/json' http://$BACKUP_ELASTIC/_snapshot/$REPOSITORY_NAME
/$INDEX_NAME/_restore --data-binary @$DATA_FILE`
echo ""
rm -f $DATA_FILE

if [ "$RESULT" != "{\"accepted\":true}" ]
then
        DATE=`date +"%Y%m%d_%H%M%S"`
        echo "$DATE Fail to restore $INDEX_NAME's snapshut : $RESULT" >> $LOGFILE
        exit
fi

########## If you want to set additional configuration, use this part  ##########
echo "{\"index.routing.allocation.total_shards_per_node\" : 8 }" > $DATA_FILE
$URL_SEND_CMD -XPUT -H 'Content-Type: application/json' http://$BACKUP_ELASTIC/$INDEX_NAME/_settings --data-binary
 @$DATA_FILE
echo ""
rm -f $DATA_FILE
```

Cluster 간 인덱스 백업하기

- snapshot 생성에 성공하였으면 Warm Cluster 에 Restore 시작
  - Warm Cluster 에 Restore 가 완료되었는지 확인, 진행중이면 대기
- Warm Cluster 에 Restore 가 성공적으로 이루어졌으면
  - 생성했던 snapshot 삭제, Hot Cluster 에서 백업된 인덱스 삭제

```
while [ 1 ]
do
        CNT=`$URL_SEND_CMD -s http://$BACKUP_ELASTIC/_cat/shards | grep $INDEX_NAME | grep -v STARTED | wc | cut -
c7-8`
        DATE=`date +"%Y%m%d_%H%M%S"`
        echo "$DATE [$INDEX_NAME] CNT : $CNT"
        if [ $CNT -eq 0 ]
        then
                DATE=`date +"%Y%m%d_%H%M%S"`
                echo "$DATE index is opened : $INDEX_NAME" >> $LOGFILE
                ########## delete snapshut ##########
                $URL_SEND_CMD -XDELETE http://$BACKUP_ELASTIC/_snapshot/$REPOSITORY_NAME/$INDEX_NAME
                DATE=`date +"%Y%m%d_%H%M%S"`
                echo "$DATE $INDEX_NAME's snapshot is deleted" >> $LOGFILE
                ########## delete source index ##########
                #$URL_SEND_CMD -XDELETE http://$CUR_ELASTIC/$INDEX_NAME
                #DATE=`date +"%Y%m%d_%H%M%S"`
                #echo "$DATE $INDEX_NAME is deleted at Cluster($CUR_ELASTIC)" >> $LOGFILE
                break
        fi
        sleep 10
done
```

Cluster 간 인덱스 백업하기

- snapshot 생성에 성공하였으면 Warm Cluster 에 Restore 시작
  - Warm Cluster 에 Restore 가 완료되었는지 확인, 진행중이면 대기

```
gangyoncBookPro:util root# echo "{ \"indices\": \"$INDEX_NAME\", \"index_settings\": { \"index.number_of_replicas\": $]
BACKUP_ELASTIC_NUM_REPLICA }, \"ignore_index_settings\": [ \"index.refresh_interval\" ] }" > $DATA_FILE
gangyoncBookPro:util root# $URL_SEND_CMD -XPOST -H 'Content-Type: application/json' http://$BACKUP_ELASTIC/_snapshot/$]
REPOSITORY_NAME/$INDEX_NAME/_restore --data-binary @$DATA_FILE
{"accepted":true}gangyoncBookPro:util root#
gangyoncBookPro:util root# $URL_SEND_CMD -s http://$BACKUP_ELASTIC/_cat/shards | grep $INDEX_NAME
filebeat-6.8.13-2021.02.09 3 p INITIALIZING        127.0.0.1 warmnode-2
filebeat-6.8.13-2021.02.09 3 r UNASSIGNED
filebeat-6.8.13-2021.02.09 2 p INITIALIZING        127.0.0.1 warmnode-1
filebeat-6.8.13-2021.02.09 2 r UNASSIGNED
filebeat-6.8.13-2021.02.09 7 p INITIALIZING        127.0.0.1 warmnode-2
filebeat-6.8.13-2021.02.09 7 r UNASSIGNED
filebeat-6.8.13-2021.02.09 1 p INITIALIZING        127.0.0.1 warmnode-2
filebeat-6.8.13-2021.02.09 1 r UNASSIGNED
filebeat-6.8.13-2021.02.09 4 p INITIALIZING        127.0.0.1 warmnode-1
filebeat-6.8.13-2021.02.09 4 r UNASSIGNED
filebeat-6.8.13-2021.02.09 6 p INITIALIZING        127.0.0.1 warmnode-1
filebeat-6.8.13-2021.02.09 6 r UNASSIGNED
filebeat-6.8.13-2021.02.09 5 p INITIALIZING        127.0.0.1 warmnode-2
filebeat-6.8.13-2021.02.09 5 r UNASSIGNED
filebeat-6.8.13-2021.02.09 0 p INITIALIZING        127.0.0.1 warmnode-1
filebeat-6.8.13-2021.02.09 0 r UNASSIGNED
```

Unassigned 된 샤드 처리하기

- 사용하는 elasticsearch REST API
  - _cat/shards
  - _cat/nodes
  - _cluster/reroute
- 사용하는 linux command
  - grep, egrep
  - awk
  - sort
  - wc
  - cut
  - tail
  - jot (MacOS), shuf(Linux)

Unassigned 된 샤드 처리하기

- Unassigned 된 샤드가 존재하는지 확인한다.
  - unassigned 된 shard 가 존재하고 나머지 모든 shard 가 정상적으로 구동중인지 확인한다. (인덱스가 recovery 중이거나 하는 경우에는 처리하지 않기 위한 확인)
- 노드당 균등하게 가질 수 있는 shard 개수를 구한다

```
TOTAL_SHARD_COUNT=`$URL_SEND_CMD $ELASTIC_ADDR/_cat/shards | grep $INDEX_NAME  | wc | cut -c7-8`
TOTAL_STARTED_SHARD_COUNT=`$URL_SEND_CMD $ELASTIC_ADDR/_cat/shards | grep $INDEX_NAME  | grep STARTED | wc | cut -c7-8`
TOTAL_UNSIGNED_SHARD_COUNT=`$URL_SEND_CMD $ELASTIC_ADDR/_cat/shards | grep $INDEX_NAME  | grep UNASSIGNED | wc | cut -c7-8`
TOTAL_NODE_COUNT=`$URL_SEND_CMD $ELASTIC_ADDR/_cat/nodes?h=node.role | grep d | wc | awk '{print $1}'`
SHARD_PER_NODE=$((TOTAL_SHARD_COUNT/TOTAL_NODE_COUNT))
MAX_SHARD_NUMBER=`$URL_SEND_CMD $ELASTIC_ADDR/_cat/shards | grep $INDEX_NAME | awk '{print $2}' | sort | tail -n 1`

TOTAL_SUM_SHARD_COUNT=$((TOTAL_STARTED_SHARD_COUNT+TOTAL_UNSIGNED_SHARD_COUNT))
if [ $TOTAL_UNSIGNED_SHARD_COUNT -gt 0 ] && [ $TOTAL_SHARD_COUNT -eq $TOTAL_SUM_SHARD_COUNT ]
then
```

Unassigned 된 샤드 처리하기

```
gangyoncBookPro:util root# $URL_SEND_CMD $ELASTIC_ADDR/_cat/shards | grep $INDEX_NAME
filebeat-6.8.13-2021.02.01 1 p STARTED        0   261b 127.0.0.1 node-1
filebeat-6.8.13-2021.02.01 1 r STARTED        0   261b 127.0.0.1 node-4
filebeat-6.8.13-2021.02.01 3 r STARTED        0   261b 127.0.0.1 node-1
filebeat-6.8.13-2021.02.01 3 p STARTED        0   261b 127.0.0.1 node-2
filebeat-6.8.13-2021.02.01 6 p STARTED        0   261b 127.0.0.1 node-4
filebeat-6.8.13-2021.02.01 6 r STARTED        0   261b 127.0.0.1 node-2
filebeat-6.8.13-2021.02.01 7 r STARTED        0   261b 127.0.0.1 node-3
filebeat-6.8.13-2021.02.01 7 p STARTED        0   261b 127.0.0.1 node-2
filebeat-6.8.13-2021.02.01 4 p STARTED        0   261b 127.0.0.1 node-3
filebeat-6.8.13-2021.02.01 4 r STARTED        0   261b 127.0.0.1 node-2
filebeat-6.8.13-2021.02.01 5 p STARTED        0   261b 127.0.0.1 node-1
filebeat-6.8.13-2021.02.01 5 r UNASSIGNED
filebeat-6.8.13-2021.02.01 2 r STARTED        0   261b 127.0.0.1 node-3
filebeat-6.8.13-2021.02.01 2 p STARTED        0   261b 127.0.0.1 node-4
filebeat-6.8.13-2021.02.01 0 p STARTED        0   261b 127.0.0.1 node-3
filebeat-6.8.13-2021.02.01 0 r STARTED        0   261b 127.0.0.1 node-4
gangyoncBookPro:util root# $URL_SEND_CMD $ELASTIC_ADDR/_cat/nodes?v
ip        heap.percent ram.percent cpu load_1m load_5m load_15m node.role master name
127.0.0.1           42         100   7    4.90                  mdi       -      node-2
127.0.0.1           62         100   9    4.90                  mdi       -      node-3
127.0.0.1           28         100   8    4.90                  mdi       *      node-1
127.0.0.1           41         100   9    4.90                  mdi       -      node-4
```

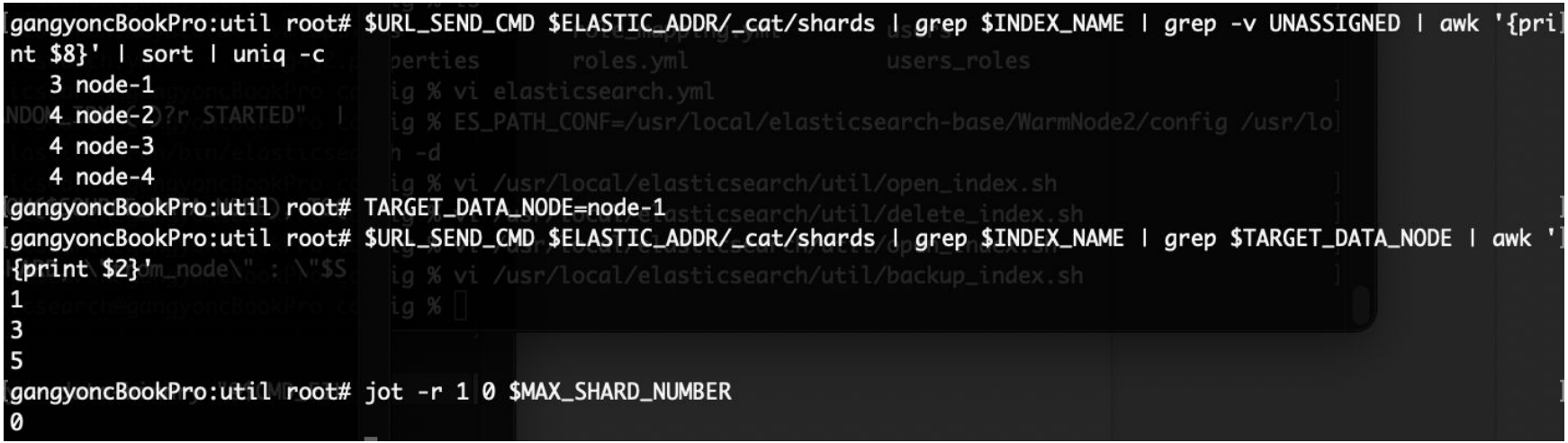
Unassigned 된 샤드 처리하기

- shard 개수를 부족하게 가지고 있는 노드를 찾는다
  - 해당 노드가 가지고 있는 shard 리스트를 구한다.
  - 구한 shard 리스트에 존재하지 않는 shard 를 임의로 하나 선정한다.

```
        TARGET_SHARD_DATA_NODES=`$URL_SEND_CMD $ELASTIC_ADDR/_cat/shards | grep $INDEX_NAME | grep -v UNASSIGNED | awk '{print $8}' | s
ort | uniq -c | awk -v shard_per_node="$SHARD_PER_NODE" '$1 < shard_per_node {print $2}'`
        for TARGET_DATA_NODE in $TARGET_SHARD_DATA_NODES
        do
                TARGET_NODE_SHARDS=`$URL_SEND_CMD $ELASTIC_ADDR/_cat/shards | grep $INDEX_NAME | grep $TARGET_DATA_NODE | awk '{print $
2}'  | sort`
                SOURCE_SHARD=-1
                while [ 1 ]
                do
                        #Linux
                        #RANDOM_IDX=`shuf -i0-$MAX_SHARD_NUMBER -n1`
                        #MacOS
                        RANDOM_IDX=`jot -r 1 0 $MAX_SHARD_NUMBER`
                        CHECK_EXIST=`echo $TARGET_NODE_SHARDS | grep $RANDOM_IDX | wc | cut -c7-8`
                        if [ $CHECK_EXIST == "0" ]
                        then
                                SOURCE_SHARD=$RANDOM_IDX
                                break
                        fi
                done
```

Unassigned 된 샤드 처리하기

- shard 개수를 부족하게 가지고 있는 노드를 찾는다
  - 해당 노드가 가지고 있는 shard 리스트를 구한다.
  - 구한 shard 리스트에 존재하지 않는 shard 를 임의로 하나 선정한다.



Unassigned 된 샤드 처리하기

- shard 개수를 부족하게 가지고 있는 노드를 찾는다
  - 위에서 찾은 임의의 shard 의 replica 를 소유한 노드를 찾는다.
  - 위에서 찾은 replica shard 를 shard 개수가 부족한 노드로 reroute 시킨다

```
            SOURCE_DATA_NODE=`$URL_SEND_CMD $ELASTIC_ADDR/_cat/shards | grep $INDEX_NAME | egrep -e " $SOURCE_SHAR
D ( )?r STARTED"  | awk '{print $8}'`
            DATE=`date +"%Y%m%d_%H%M%S"`
            echo "$DATE [$INDEX_NAME] shard is relocating for unassigned shard : SHARD($SOURCE_SHARD), FROM($SOURC
E_DATA_NODE), TO($TARGET_DATA_NODE)" >> $LOGFILE
            CMD=`echo "{\"commands\" : [ {\"move\" : { \"index\" : \"$INDEX_NAME\", \"shard\" : $SOURCE_SHARD, \"f
rom_node\" : \"$SOURCE_DATA_NODE\", \"to_node\" : \"$TARGET_DATA_NODE\" } } ]}"`
            echo "$CMD" > $CMD_FILE
            echo "" >> $CMD_RESULT_FILE
            $URL_SEND_CMD -H 'Content-Type: application/json' -XPOST $ELASTIC_ADDR/_cluster/reroute?pretty --data-
binary "@$CMD_FILE" >> $CMD_RESULT_FILE
```

Unassigned 된 샤드 처리하기

- shard 개수를 부족하게 가지고 있는 노드를 찾는다
  - 위에서 찾은 임의의 shard 의 replica 를 소유한 노드를 찾는다.
  - 위에서 찾은 replica shard 를 shard 개수가 부족한 노드로 reroute 시킨다

```
gangyoncBookPro:util root# SOURCE_SHARD=0
gangyoncBookPro:util root# $URL_SEND_CMD $ELASTIC_ADDR/_cat/shards | grep $INDEX_NAME | egrep -e " $SOURCE_SHARD ( )?r STARTED"
filebeat-6.8.13-2021.02.01 0 r STARTED       0   261b 127.0.0.1 node-4
gangyoncBookPro:util root# $URL_SEND_CMD $ELASTIC_ADDR/_cat/shards | grep $INDEX_NAME | egrep -e " $SOURCE_SHARD ( )?r STARTED"
| awk '{print $8}'
node-4
gangyoncBookPro:util root# TARGET_DATA_NODE=node-1
gangyoncBookPro:util root# SOURCE_DATA_NODE=node-4
gangyoncBookPro:util root# echo "{\"commands\" : [ {\"move\" : { \"index\" : \"$INDEX_NAME\", \"shard\" : $SOURCE_SHARD, \"from_
node\" : \"$SOURCE_DATA_NODE\", \"to_node\" : \"$TARGET_DATA_NODE\" } } ]}" | python -m json.tool
{
    "commands": [
        {
            "move": {
                "from_node": "node-4",
                "index": "filebeat-6.8.13-2021.02.01",
                "shard": 0,
                "to_node": "node-1"
            }
        }
    ]
}
```