Elasticsearch Watcher

엘라스틱서치 공식 문서

- https://www.elastic.co/guide/en/watcher/current/index.html

엘라스틱서치 공식 유튜브

- https://www.youtube.com/watch?v=gD7MGt5jgDY
- **https://www.youtube.com/watch?v=-UVUzGMpyoo (한 번 실습해볼 것★)**

```
PUT _watcher/watch/failed-login
{
  "trigger": {
    "schedule": {
      "interval": "10s"
    }
  },
  "input": {
    "search": {
      "request": {
        "indices": "auth_logs",
        "body": {
          "size": 0,
          "query": {
            "bool": {
              "filter": [
                {
                  "range": {
                    "@timestamp": {
                      "from": "now-5m",
                      "to": "now"
                    }
                  }
                },
                {
                  "term": {
                    "outcome": "failed password"
                  }
                }
              ]
            }
          },
          "aggs": {
            "failed_ip": {
              "terms": {
                "field": "source_ip"
              }
            }
          }
        }
      }
    }
  },
  "condition": {
    "compare": {
      "ctx.payload.aggregations.failed_ip.buckets.0.doc_count": {
        "gt": 10
      }
    }
  },
  "actions": {
    "email_admin": {
      "email": {
        "to": "[]@[]",
        "subject": "Warning: Multiple failed logins from same IP",
        "body": "error_kmh"
      }
    }
  }
}
```

```
PUT _watcher/watch/test_watcher(=index 이름)
{
  "trigger" : {

  },
  "input" : {

  },
  "condition":{

  },
  "transform":{

  },
  "actions":{

  }
}
```

총 trigger, input, condition, transform, actions 5개로 구성되어 있다.

5개를 다 쓸 필요는 없다.

1.trigger

```
"trigger" : {
    "schedule" : {
      "interval" : "1m" }
  },

"trigger" : {
    "schedule" : {
      "daily" : { "at" : "noon" }
    }
```

꼴로 쓸 수 있으며, 반복될 시간을 정의해주는 역할을 한다.

2.input

```
"input": {
    "search": {
      "request": {
        "indices": [
          ".marvel-es-1-*"
        ],
        "types" : [
          "node_stats"
        ],
        "body": {
          "size" : 0,
          "query": {
            "filtered": {
              "filter": {
                "range": {
                  "timestamp": {
                    "gte": "now-2m",
                    "lte": "now"
                  }
                }
              }
            }
          },
          "aggs": {
            "minutes": {
              "date_histogram": {
                "field": "timestamp",
                "interval": "minute"
              },
              "aggs": {
                "nodes": {
                  "terms": {
                    "field": "source_node.name",
                    "size": 10,
                    "order": {
                      "cpu": "desc"
                    }
                  },
```

```
                  "aggs": {
                    "cpu": {
                      "avg": {
                        "field": "node_stats.process.cpu.percent"
                      }
                    }
                  }
                }
              }
            }
          }
        }
      }
    }
  },
  "throttle_period": "30m",

혹은
"input" : {
      "search" : {
        "request" : {
          "indices" : [ "seats" ],
          "body" : {
            "query" : { "term": { "sold": "true"} },
            "aggs" : {
              "theatres" : {
                "terms" : { "field" : "play" },
                "aggs" : {
                  "money" : {
                    "sum": { "field" : "cost" }
                  }
                }
              }
            }
          }
        }
      },

혹은
"input" : {
    "search" : {
      "request" : {
        "indices" : [ "logs" ],
        "body" : {
          "query" : {
            "match" : { "message": "error" }
          }
        }
      }
    }
  },
```

꼴로 쓸 수 있으며, 구체적으로 에러를 감시할 데이터를 입력하는 역할을 한다.

3.condition

```
"condition": {
    "script":  "if (ctx.payload.aggregations.minutes.buckets.size() == 0) return false; def latest = ctx.payload.aggregations.minutes.
```

꼴로 쓸 수 있으며, 에러로 설정한 구체적인 조건을 설정하는 역할을 한다.

4.transform

```
"transform" : {
      "script":
      """
        return [
          'money_makers': ctx.payload.aggregations.theatres.buckets.stream()
            .filter(t -> {
                return t.money.value > 50000
            })
            .map(t -> {
                return ['play': t.key, 'total_value': t.money.value ]
            }).collect(Collectors.toList()),
          'duds' : ctx.payload.aggregations.theatres.buckets.stream()
            .filter(t -> {
                return t.money.value < 15000
            })
```

```
            .map(t -> {
                return ['play': t.key, 'total_value': t.money.value ]
            }).collect(Collectors.toList())
          ]
      """
    },
```

꼴로 쓸 수 있으며, 데이터 중 변환해야될 부분을 변환해주는 역할을 한다.

5.actions

```
"actions" : {
  "send_email" : {
    "email" : {
      "to" : "<username>@<domainname>",
      "subject" : "Watcher Notification",
      "body" : "{{ctx.payload.hits.total}} watches took more than 2.5 seconds to execute.",
      "attachments" : {
        "data_attachments" : {
          "data" : {
            "format" : "json"
          }
        }
      }
    }
  }
}
```

꼴로 쓸 수 있으며, 에러구문이 생길 시 정보를 보낼 창구를 설정하는 역할을 한다.

action이 실행될려면 elasticsearch.yml에 해당 구문을 넣어주어야 한다.

`watcher.actions.email.service.account:`

https://www.elastic.co/guide/en/watcher/current/email-services.html → 이메일

`watcher.actions.slack.service:`

https://www.elastic.co/guide/en/watcher/current/configuring-slack.html → 슬랙

G-mail의 경우 두 가지 옵션을 변경해주어야한다.

https://pdi-mz-support.zendesk.com/hc/ko/articles/360050901451--Account-%EB%B3%B4%EC%95%88[...]EC%82%AC%EC%9A%A9-%EC%84%A4%EC%A0%95-%EB%B0%A9%EB%B2%95

https://docs.3rdeyesys.com/99.etc/etc_smt_auth_to_google_gmail_account/

watcher vs alert

watcher	alerting
elasticsearch의 watcherAPI를 통해 관리	Kibana의 Task Manager에서 관리
stack management>watcher메뉴에서만 생성관리	logs, metric, APM, ML등의 app메뉴에서 생성 가능
elasticsearch 시스템 로그 (elastic)	kibana시스템로그 (kibana에서 로그 생성)