1. ELK Stack 소개

ELK Stack?

- 각 서버에 저장된 로그성 데이터를 한곳에 모아서 시각화 하는 Opensource
- Elasticsearch / Logstash / Kibana를 줄여 ELK라고 명칭하고 있었고 , fileBeat가 추가되면서 ELK Stack으로 명칭 .

Component별 역활

- **filebeat** : 시스템에 기록된 로그데이터를 logstash로 보내기 위한 역활 (logstash보다 경량화되었고 , 로그데이터가 json으로 파싱되어 있는 경우 logstash를 거치지 않고 elasticsearch바로 전송할 수 있다 함)
- **Logstash** : filebeat에서 받은 로그 , 시스템에 기록된 로그데이터를 Elasticsearch로 보내는 역활
- **Elasticsearch** : logstash에서 전달받은 데이터를 DB화 수행
- **Kibana** : Elasticsearch에서 정제된 데이터를 시각화 수행

하드웨어 요구사항

Component	Hardware	설명	비고

Elasticsearch	cpu	8Core	
Elasticsearch	mem	□□:16GB□□:64GB□□	8GB□□□□□□□□□□□□□□□□□

Elasticsearch	disk	SSD□□	* s s d □ □ i o □ □ □ □ □ d e a d l i n e □ □ n o o p □ □ □ □ □ c f q : r / r , d e a d l i n e :

Kibana	cpu	8 Core	
Kibana	mem	□□1GB□□:4GB□□	
Kibana	disk	□□□□	
Logstash	cpu	2 Core	
Logstash	mem	2GB	

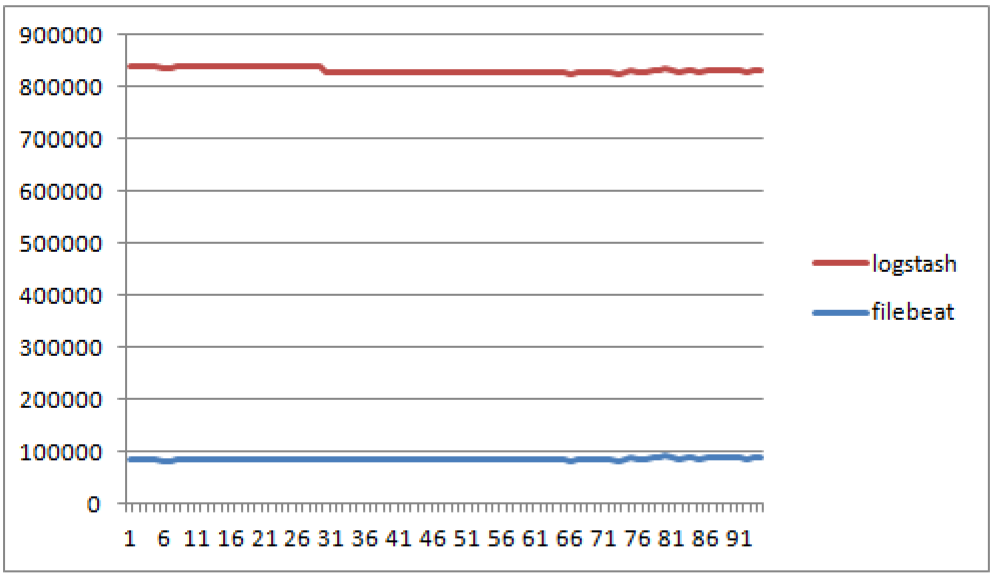
Logstash	disk	제약없음	

* Elasticsearch에서 사용할 디스크 용량 계산 방법

(예상 로그양 * 보관일) * 데이터 노드수가 기본적인 용량 계산방법

ELK Data Flow

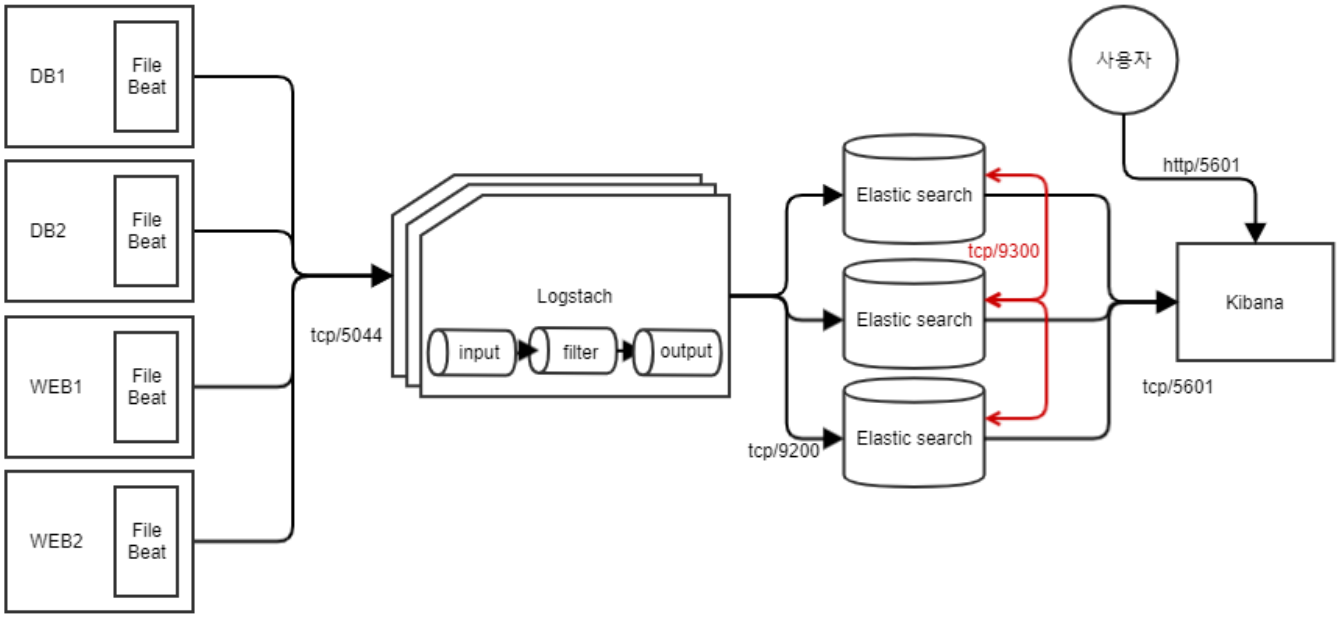
1. Node에 설치되는 filebeat대신에 logstash를 설치해서 ElasticSearch로 바로 전송할수 있음 .
2. filebeat / logstash의 RSS메모리 사용량 비교
   - 로그 데이터는 초당 1Mbyte씩 전송할수 있게 로그파일 생성 , 총 100초동안 초단위로 수집 진행



Elasticsearch 구성

- Elasticsearch 노드 종류
  - **master node** : Elasticsearch의 인덱스 메타데이터 , 샤드 , 클러스터 상태 정보를 관리하는 역할 .. 서버 수량이 많을때 모든 노드가 master역할을 수행할 경우 성능상 부담이 되기 때문에 통상적으로 10대이상 구성될 경우 master/data 노드 분리해서 운영하는것이 best practice

    - **data node** : 실제 데이터를 저장하는 노드
- Cluster기반의 통신정책
    - 클라이언트와 Elasticsearch와의 통신을 위한 포트 : tcp/9200
    - Elasticsearch노드간의 통신을 위한 포트 : tcp/9300
- 용어 확인 / 비교



DBMS (like, mysql)	Elasticsearch
database	index
table	type
row	document
column	field
schema	mapping
index	index
sql	Query DSL
select	GET (Rest API사용)

DBMS (like, mysql)	Elasticsearch
update	PUT (Rest API사용)
insert	POST (Rest API사용)
delete	DELETE (Rest API사용)

Index 수명주기에 따른 노드 관리 : 6.7이후부터 공식릴리즈에 포함된 기능

- hot : 가장많이 검색되는 인덱스
- warm : 검색되긴하지만, 자주 검색되지 않은 인덱스
- cold : 자주 검색되지 않으나, 만약을 위해 유지하는 인덱스

Index 관리 규칙

- Create : 인덱스 생성
- close : 인덱스는 유지하되, write는 불가
- delete : 인덱스 삭제

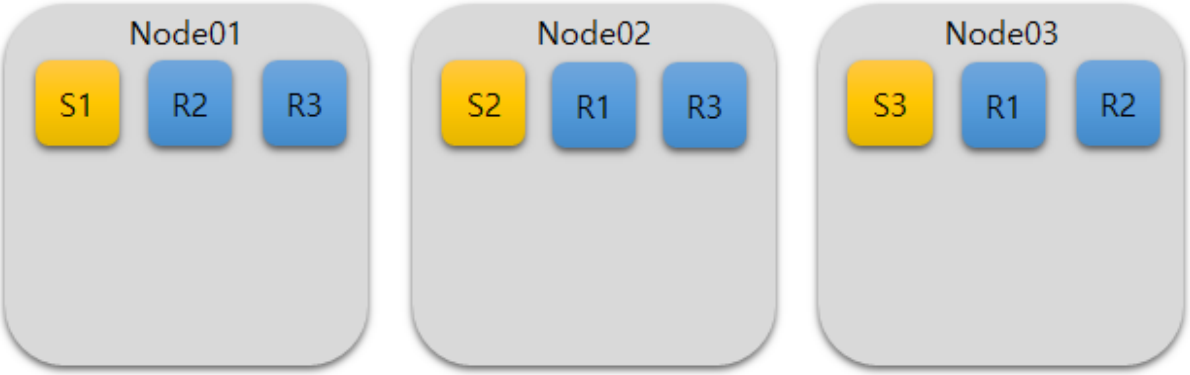
Index 상태 확인

- green : 샤드, 리플리케이션 모두 정상인 상태
- yellow : 일부 인덱스가 비정상적으로 구동되고 있는 상태
- red : 모든 인덱스의 샤드가 비정상적으로 구동되고 있는 상태, 인덱스 데이터의 read/write가 불가능

데이터 관리방안 (인덱스 생성시 정해야 하고, 데이터가 저장되어 있는 경우 re-index 작업해야 반영됨)

- shard :
    - Document를 노드단위로 분산저장하는 방식 (7.x부터는 Default가 1로 설정, 6.x 이하는 Default가 5)

    - 데이터를 분산 저장하는 것뿐만 아니라 분산 검색도 수행함. shard의 갯수가 증가할수록 쿼리 속도도 증가 (분산 쿼리)
- replica :
    - Primary shard(원본데이터)갯수만큼 복제 shard(2개), replica(1개)로 구성한 경우 2x1 = 2개의 replica가 생성, replica의 데이터는 primary shard가 없는 노드에 각각 저장
    - node = 3, shard = 2, replica = 1로 구성된 경우 노드별 데이터 저장 방식



Revision #6

Created 15 July 2022 16:28:00 by artop0420

Updated 30 March 2024 00:34:41 by artop0420