ELK在Adxmi系统监控中的应用/Elasticsearch简介

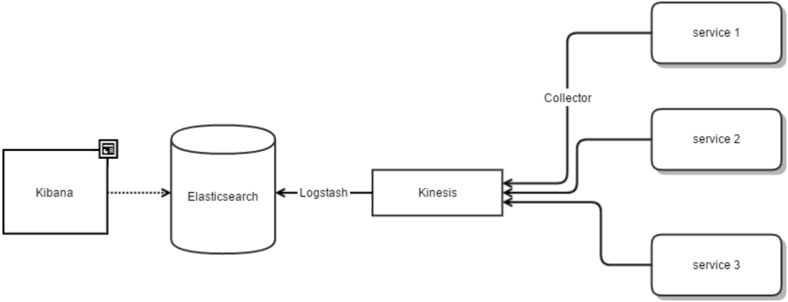
杨辰@Adxmi

2016-02-25

ELK

- **E**lasticsearch: 分布式, 实时, 全文搜索引擎
- **L**ogstash: 日志传输, ETL工具
- **K**ibana: 前端面板, 插件化

Adxmi系统监控



- 错误组件统计，响应时间统计
- 跨区域，多节点

统计数据结构

```
// Stat 监控统计信息
type Stat struct {
    name          string    // 统计组件名
    service       string    // 所在服务名
    host          string    // 所在节点名
    tstart, tstop time.Time // 统计时间段
    //
    resultOK  int            // 成功次数
    resultErr int            // 失败次数
    errors    map[string]int // 错误类型分类统计
    //
    duration time.Duration // 执行用时相关信息
}
```

Collector API

```
func MyFunc(input Input) (output Output, err error) {

    stat := collector.New("MyFunc")

    defer func() {
        stat.Report(err)
        collector.Save(stat)
    }()

    // 以下业务逻辑代码
    // ...
}
```

使用情况

Demo

问题:

- 基于监控数据的告警缺失
  - 参考elastic:watcher
- 监控代码和业务逻辑强耦合, 不够通用

可做的点:

- 数据可视化
- ELK系统扩容, 要能撑起更大的数据量

Elasticsearch

- RESTful 风
- Lucene 骨
- NoSQL 系
  - JSON document 存储
  - No Schema

和传统数据库的术语对比

RDS	Elasticsearch
database	index
table	type
row	document
column	field
schema	mapping
index	(all)
SQL	query DSL

CRUD 操作

- **Create**

```
POST /{index}/{type}
PUT /{index}/{type}/{id} {"field": "value", ...}
```

- **Read**

```
HEAD /{index}/{type}/{id}
GET /{index}/{type}/{id}
```

- **Delete**

```
DELETE /{index}/{type}/{id}
```

- **Update**: Create + Delete
  - Versioning
  - Optimistic concurrency control by last-write-wins

Query DSL

SQL	DSL (JSON format)
=	{“term”: {field: val}
IN	{“terms”: {field: [val, . . .]}
LIKE	{“wildcard:” {field: pattern}}
BETWEEN AND	{“range”: {field: {“gt”: val, “lt”: val}}}
AND / OR / NOT	{“bool”: {“must”/“should”/“must_not”: . . . }
Aggregations	{“aggs”: . . . }
JOIN	{“nestted”/“has_child”/“has_parent”: . . . }

```
// SELECT * FROM megacorp.employee
// WHERE age > 30 AND last_name = "smith"
GET /megacorp/employee/_search
{
  "query": {
    "filtered": {
      "filter": {
        "range": { "age": { "gt": 30 } }
      },
      "query": {
        "match": {
          "last_name": "smith"
        }
      }
    }
```

```
// SELECT interests, avg(age) FROM megacorp.employee
// GROUP BY interests
GET /megacorp/employee/_search
{
  "aggs": {
    "all_interests": {
      "terms": { "field": "interests" },
      "aggs": {
        "avg_age": {
          "avg": {
            "field": "age"
          }
        }
      }
    }
```

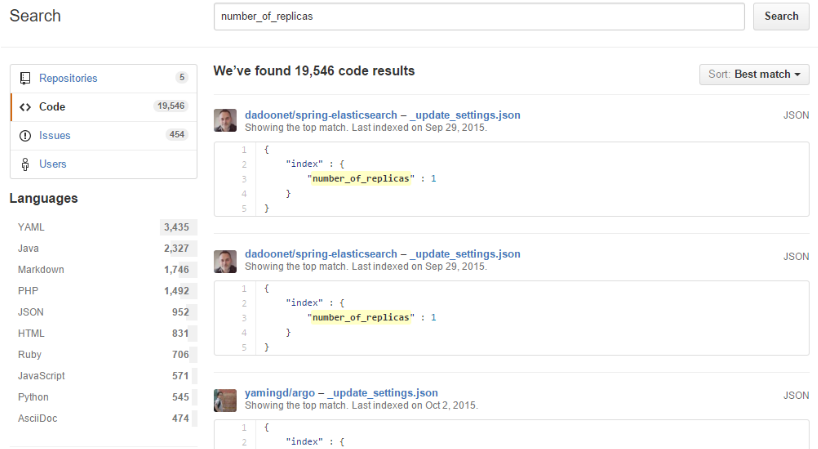
query DSL 学习难度高

JSON in / JSON out

解决办法:

- Kibana 界面化操作
- SQL to query DSL
  - github.com/NLPchina/elasticsearch-sql

全文搜索



Search

number_of_replicas

Repositories	5
Code	19,546
Issues	454
Users	

Languages

YAML	3,435
Java	2,327
Markdown	1,746
PHP	1,492
JSON	952
HTML	831
Ruby	706
JavaScript	571
Python	545
AsciiDoc	474

We've found 19,546 code results

dadoonet/spring-elasticsearch – _update_settings.json JSON

Showing the top match. Last indexed on Sep 29, 2015.

```
{
    "index" : {
        "number_of_replicas" : 1
    }
}
```

dadoonet/spring-elasticsearch – _update_settings.json JSON

Showing the top match. Last indexed on Sep 29, 2015.

```
{
    "index" : {
        "number_of_replicas" : 1
    }
}
```

yamingd/argo – _update_settings.json JSON

Showing the top match. Last indexed on Oct 2, 2015.

```
{
    "index" : {
```

- Use Cases: GitHub, WikiMedia, . . .

全文搜索原理(Lucene)

1. Document

   ```
   # doc1
   The quick brown fox jumped over the lazy dog.
   # doc2
   Quick brown foxes leap over lazy dogs in summer.
   ```

2. Token (via Tokenizer)
3. Term (via Linguistic Processor)

   ```
   foxes -> fox
   jumped -> jump
   leap -> jump
   ...
   ```

4. Inverted Index

Term	doc1	doc2	...
brown	1	1	
dog	1	1	
fox	1	1	
in	0	1	
jump	1	1	
lazy	1	1	
over	1	1	
quick	1	1	
summer	0	1	
the	2	0	
...			

Relevance Score / 相似度分析

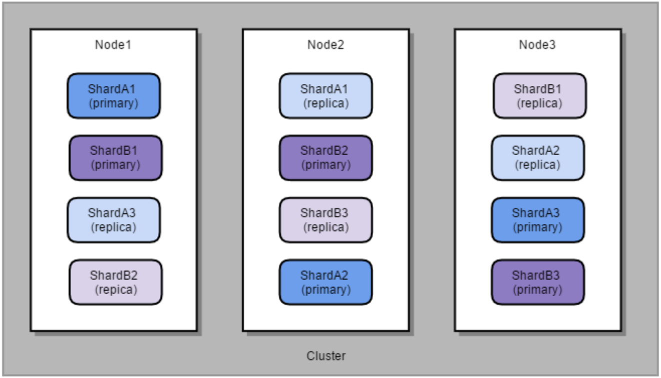
- Vector Space Model (VSM)
  - 特征向量
  - 余弦定理

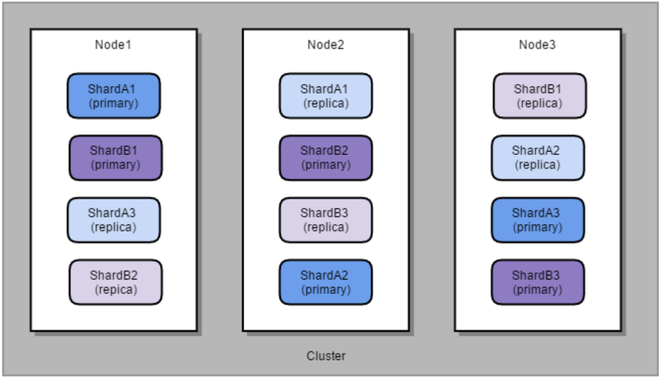
Cluster / 集群

- **node**: a single instance of Elasticsearch
- **index**:

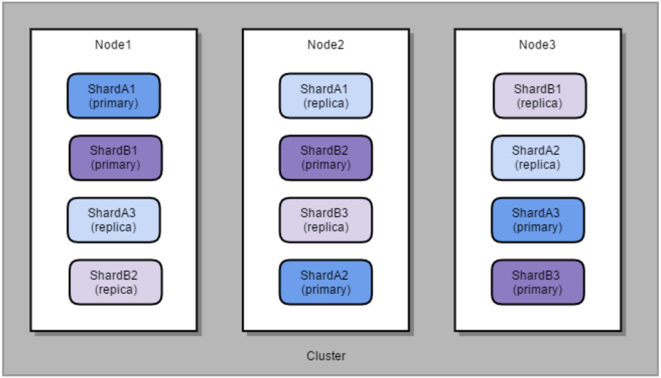
  > *. . . an index is just a logical namespace that points to one or more physical shards.*

- **shard**: a single instance of Lucene
  - $\sim$ partition
- **replica**: duplicated shard
  - primary shard push write to replica
  - $\sim$ HA


Node1
ShardA1 (primary)
ShardB1 (primary)
ShardA3 (replica)
ShardB2 (repica)
Node2
ShardA1 (replica)
ShardB2 (primary)
ShardB3 (replica)
ShardA2 (primary)
Node3
ShardB1 (replica)
ShardA2 (replica)
ShardA3 (primary)
ShardB3 (primary)
Cluster

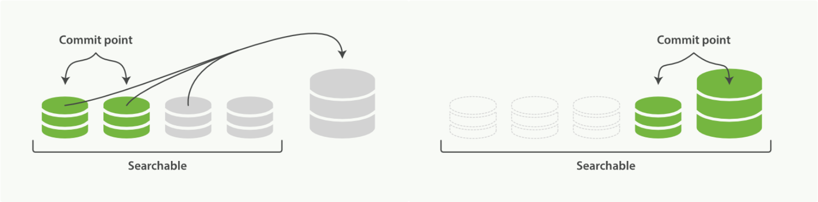


- *more shards per index: faster indexing, more scale*



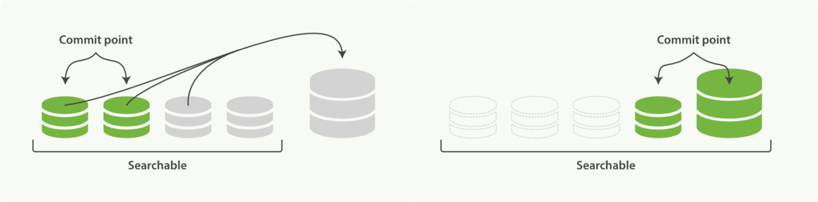
- *more shards per index: faster indexing, more scale*
- *more replica per shard: faster searching, more failover*

Segment



- **segment**: inverted index
  - multiple segments per shard
  - immutable up to delete
  - auto-merged by flushing
  - $\sim$ WAL
  - shard merge query result from segments

Segment



- **segment**: inverted index
  - multiple segments per shard
  - immutable up to delete
  - auto-merged by flushing
  - $\sim$ WAL
  - shard merge query result from segments
- *more segments per shard: longer search time*

时间序列数据管理策略

- 按天索引
- 定期清理旧索引
- 优化昨天之前的数据

  curator delete / close / optimize / snapshot / . . .

参考

- Elasticsearch: The Definitive Guide
- Elasticsearch 實戰介紹