**ES学习笔记之-Translog实现机制的理解**

ES作为一个NoSQL，典型的应用场景就是存储数据。即用户可以通过api添加数据到es中。由于Lucene内部的实现， 每次添加的数据并不是实时落盘的。而是在内存中维护着索引信息，直到缓冲区满了或者显式的commit, 数据才会落盘，形成一个segement，保存在文件中。

那么假如由于某种原因，ES的进程突然挂了，那些在内存中的数据就会丢失。而实际上，用户调用api, 返回结果确认用户数据已经添加到索引中。这种数据丢失是无法被接受的。怎么解决这个问题呢？

ES实现了Translog， 即数据索引前，会先写入到日志文件中。假如节点挂了，重启节点时就会重放日志，这样相当于把用户的操作模拟了一遍。保证了数据的不丢失。

通过ES的源码，了解一下实现的细节。 首先关注Translog类。

Translog类是一个索引分片层级的组件，即每个index shard一个Translog类。它的作用是: 将没有提交的索引操作以持久化的方式记录起来(其实就是写到文件中)。

InternalEngine 在commit metadata中记录了当前最新的translog generation。 通过这个 generation，可以关联到所有没有commit的操作记录。

每个Translog实例在任何时候都只会有一个处于open状态的translog file. 这个translog file跟translog generation ID是一一映射的关系。

出于性能的考虑，灾后重建并不是回放所有的translog, 而是最新没有提交索引的那一部分。所以必须有一个checkpoint, 即translog.ckp文件。

综上，从文件的视角看待translog机制其实是两个文件:

$ tree translog

translog

├── translog-11.tlog

└── translog.ckp

translog记录日志的格式如下:|记录size|操作的唯一id|操作的内容|checksum| 每次add操作返回的location会记录到versionMap中，这样就能实现realtime get的功能了。

了解了这一点，在配置es的时候，有两种途径可以提升ES索引的性能。

a. 将translog日子和索引配置到不同的盘片。

b. 将translog的flush间隔设置长一些。比如如下的参数:

index.translog.sync\_interval : 30s

index.translog.durability : “async”

index.translog.flush\_threshold\_size: 4g

index.translog.flush\_threshold\_ops: 50000

了解了translog的机制，会发现，即使是translog机制，也并不能完全能避免数据的丢失。在性能和数据丢失容忍度上，还是需要做一些平衡。