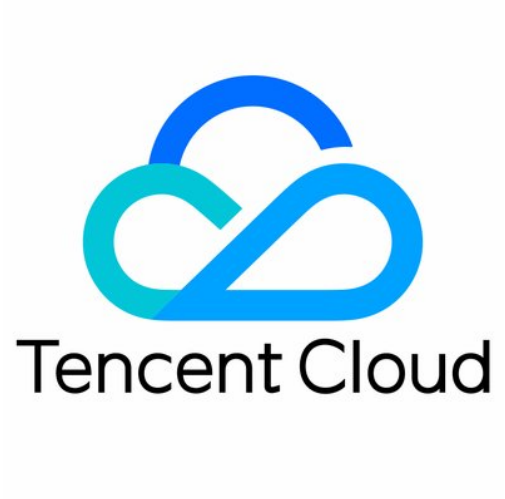
Elasticsearch Service

ES Kernel Enhancement

Product Documentation



ES Kernel Enhancement

Last updated : 2021-03-12 14:58:58

While maintaining full compatibility with the open-source Elasticsearch kernel, Tencent Cloud ES team has been continuously and thoroughly exploring and optimizing its kernel based on its rich experience in large-scale applications in various use cases, so as to enhance cluster performance, improve stability, and reduce costs. In addition, it has been maintaining close communications with the open-source community. This document describes kernel optimizations.

The following table summarizes the key kernel optimizations that the ES team has made by **July 2020** since the start of its kernel research:

Optimization Dimension	Optimization Category	Optimization Policy	Supported Versions
Performance	Write performance	The translog lock mechanism is optimized, increasing the overall write performance by 20%. Write deduplication and segment file cropping are optimized, increasing the performance of writes with primary keys by over 50%.	7.5.1
	Query performance	• The aggregation performance is optimized, making query pruning more efficient and improving the composite aggregation performance by 3-7 times in sorting scenarios. • The query cache is optimized by canceling data caches with high overheads and low hit rates, reducing query glitches from 750 ms to 50 ms in actual use cases. • The merge policies are optimized by developing proprietary merge policies based on time series and size similarity and auto warm shard merge policy, improving the query performance by over 40% in search scenarios. • Sequence capture in the query fetch phase is optimized, increasing the cache hit rate and improving the performance by over 10% in scenarios where the result set is large.	6.4.3, 6.8.2, 7.5.1
Stability	Availability	• Traffic can be limited through a smooth line curve at the access layer. • The coordinator node performs memory bloat estimation after receiving results returned by the	6.4.3, 6.8.2, 7.5.1

	data node to check whether the estimated memory will exceed the limit. • Result sets of large aggregated queries are checked in a streaming manner, and requests will be canceled if the used memory reaches the threshold. • The proprietary single request circuit breaker can prevent a large query from occupying excessive resources and thus affecting other queries. • Node crashes and cluster avalanches caused by high-concurrence writes and large queries are significantly reduced, and the overall availability is increased to 99.99%.	
Balancing policy	• Balancing policies based on index and node distribution are introduced, alleviating the serious uneven allocation of shards caused by new nodes added to the cluster. • The uneven allocation of shards among multiple disks (multiple data directories) is alleviated. • The balance of shards of newly created indices in cluster scale-out scenarios and multiple-disk scenarios is improved, reducing OPS costs.	5.6.4, 6.4.3, 6.8.2, 7.5.1
Rolling restart speed	• The logic of reusing local data for shards in case of node restart is optimized. • The restoration of shard copies within a scheduled delay time period can be precisely controlled. The time to restart one single node in a large cluster is reduced from over 10 minutes to 1 minute.	6.4.3, 6.8.2, 7.5.1
Online master switch	The proprietary online master switch feature allows you to switch the master online in seconds by specifying the preferred master through APIs. Typical use cases include: • You can switch online from the current heavily loaded master to a node with a higher specification and a lower load during manual OPS. • During rolling restart, you can restart the master node last and quickly switch the master role to another node before the restart, which	6.4.3, 6.8.2, 7.5.1

		helps reduce the service interruption from minutes to seconds.	
Costs	Memory	• The proprietary off-heap cache helps achieve FST off-heap optimization. • The off-heap cache ensures that the FST reclaim policy is controllable. • The precise eviction policy improves the cache hit rate. • Zero-copy and multi-level caches guarantee high access performance. • The heap memory overheads are significantly reduced, the GC time is decreased by over 10%, and the disk capacity of a single node can reach 50 TB, with read/write performance generally unaffected.	6.8.2, 7.5.1
	Storage	• The proprietary ID field-based row storage cropping algorithm reduces storage overheads by over 20% in time series scenarios. • A new compression algorithm is introduced, increasing the compression ratio by 30%-50% and the compression performance by 30%.	5.6.4, 6.4.3, 6.8.2, 7.5.1