

Performance: Distribution Agent improve network performance

Last updated by | Holger Linke | Dec 21, 2022 at 7:13 AM PST

Contents

- [Issue](#)
- [Investigation / Analysis](#)
- [Mitigation](#)
 - [Parameter SubscriptionStreams](#)
 - [Parameter CommitBatchSize](#)
 - [Parameter CommitBatchThreshold](#)
 - [How to configure the parameters](#)
- [More Information](#)
- [Public Doc Reference](#)

Improve performance and throughput for the Distribution Agent

Issue

The customer has configured Transactional Replication and it is working as expected, without any errors or failures. After the initial snapshot has been applied, they notice that the Distribution Agent is unable to keep up with the workload. The latency is increasing although there is no apparent bottleneck at the Distributor or Subscriber.

The symptoms are pointing to the Distribution Agent itself being the bottleneck. The customer is looking for ways to improve the Distribution Agent throughput.

Investigation / Analysis

To confirm the customer findings, you should still consider going through the [Performance Troubleshooting Initial Steps](#), especially the questions around blocking. The symptoms as described above could also result from the Distribution Agent reader or writer thread being blocked in the Distribution and/or Subscriber databases.

See the Mitigation section below for options to optimize the Distribution Agent configuration for performance.

Mitigation


The best configuration for the replication agents depends highly on the customer environment, that is the overall workload pattern and the number of replication agents that are running in the same databases. The optimal values are subject to the actual results seen on the replication environment and the customer needs to test them in their environment. There isn't a single value adjustment that improves performance for every

situation. If the customer insists on a "one size fits all" solution, then they should keep the default values, as these were chosen to match the most common scenarios.

Parameter SubscriptionStreams

Increasing the `-SubscriptionStreams` value can greatly improve the overall throughput of the Distribution Agent. It opens multiple connections to the Subscriber for applying batches in parallel while maintaining many of the transactional characteristics present when using a single thread. However, depending on the number of Managed Instance vCores and other metadata conditions (such as primary key, foreign keys, unique constraints, and indexes), the higher value of `SubscriptionStreams` might actually have an adverse effect. Additionally, if a stream fails to execute or commit, the Distribution Agent falls back to using a single stream to retry the failed batches. After the failed batches are successfully committed, it switches back to use parallel streams.

A good start value for customer tests is setting `SubscriptionStreams` to 4 or 8.

A value for this agent parameter can be specified either as a job step parameter, or by using the `@subscriptionstreams` parameter of [sp_addsubscription \(Transact-SQL\)](#)  when creating the subscription.

Parameter CommitBatchSize

`CommitBatchSize` configures the number of transactions to be sent to the Subscriber before the batch is committed. The default is 100 and the maximum is 10000. This parameter is ignored when the snapshot is being applied on the Subscriber by the Distribution Agent.

This parameter can improve overall throughput by increasing the writer thread batch size, thus reducing network roundtrips and improving the transaction log performance at the Subscriber. Committing a set of transactions has a fixed overhead; by committing a larger number of transactions less frequently, the overhead is spread across a larger volume of data (especially important for PaaS subscribers).

The trade-off is that, if a failure occurs, the Distribution Agent must roll back and start over to reapply a larger number of transactions. For unreliable networks, like possibly in on-premise <--> MI scenarios, a lower value can result in fewer failures and thus a smaller number of transactions to roll back and reapply if a failure occurs.

A good start value for customer tests in well-connected environments is setting `CommitBatchSize` to 200.

Parameter CommitBatchThreshold

`CommitBatchThreshold` is the number of replication commands to be sent to the Subscriber before the batch is committed. The default is 1000 and the maximum is 10000. This parameter is ignored when the snapshot is being applied on the Subscriber by the Distribution Agent.

This parameter works the same way like `CommitBatchSize` but at command level. It can be used to optimize scenarios where replicated transactions consist of very few or a lot of individual commands. The batch will be committed when either `CommitBatchSize` OR `CommitBatchThreshold` has been reached.

A good start value for customer tests in well-connected environments is setting `CommitBatchThreshold` to the default of 1000.

How to configure the parameters

The Distribution Agent parameters described above can be configured by adding them to the "Run Agent" job step.

Example: `-SubscriptionStreams 8 -CommitBatchSize 200 -CommitBatchThreshold 1000`







- Identify the Log Reader Agent job in the SSMS Job Activity Monitor.
- Open the job and go to "Steps".
- Edit the "Run agent" step and add the parameters to the end of the parameter command line.
- Save the change.
- Restart the agent job and monitor the performance.

You can also [Work with Replication Agent Profiles](#)  to configure agent parameters.

More Information

See the article [Optimizing Replication Agent profile parameters for better performance](#)  for a discussion about optimizing different scenarios.

Public Doc Reference

- [Enhance Transactional Replication Performance](#) 
- [Optimizing Replication Agent profile parameters for better performance](#) 
- [Enhance General Replication Performance](#) 
- [Best Practices for Replication Administration](#) 
- [Publishing Stored Procedure Execution in Transactional Replication](#) 
- [Replication Distribution Agent](#) 

How good have you found this content?



-