New option for serial gw sender dispatcher threads start

This is a draft of the RFC to be published in the Geode community

Problem

This RFC aims to solve a problem observed in serial gateway senders when the receivers they have to connect to are configured using the same host and port.

The reason for such a setup (already discussed in GEODE-7565 RFC) is deploying Geode cluster on a Kubernetes cluster where all gateway receivers are reachable from the outside world on the same VIP and port. Other kinds of configuration (different hostname and/or different port for each gateway receiver) are not cheap from operation & maintenance and resources perspective in cloud native environments and also limit some important use-cases (like scaling).

Currently, it is possible to set gateway receivers as described, but there are some problems derived from this configuration that must be solved prior to state that this configuration is supported by Geode.

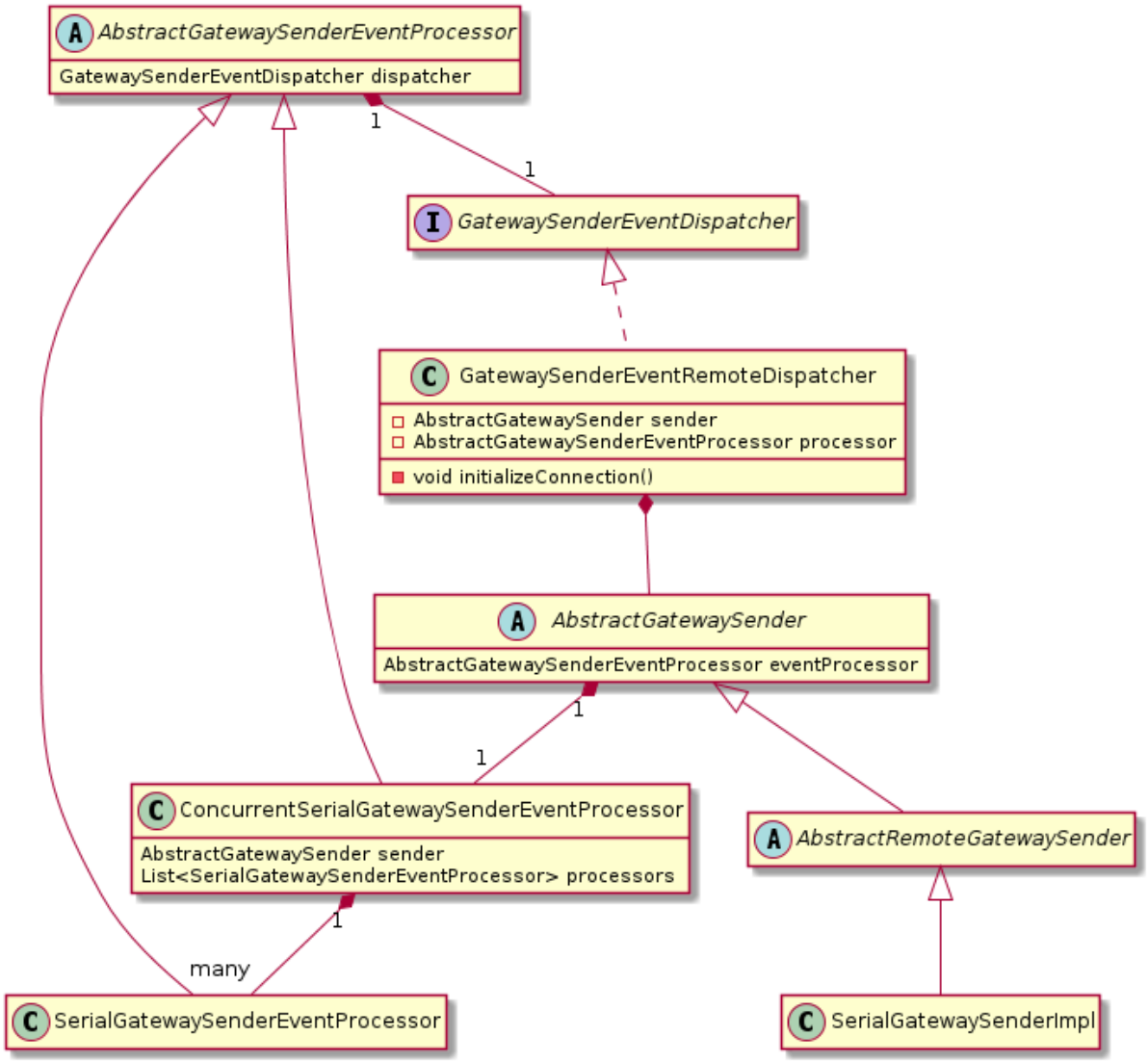
In this scenario, when serial gateway senders use more than one dispatcher thread, it is not possible to ensure that all of them are connected to the same receiver. As only host and port information is given to perform the connection, it can be pointed to any of the receivers sharing that host and port.

Anti-Goals

This RFC does not aim to solve any other issue derived from the specified configuration but the one described previously.

Solution

SerialGatewaySenderImpl class contains an AbstractGatewaySenderEventProcessor object called eventProcessor. When more than one dispatcher thread is used, eventProcessor type is RemoteConcurrentSerialGatewaySenderEventProcessor (child of ConcurrentSerialGatewaySenderEventProcessor). The eventProcessor object holds a list of SerialGatewaySenderEventProcessor objects, one per dispatcher thread.



The proposed solution impacts how `ConcurrentSerialGatewaySenderEventProcessor` initializes all its `SerialGatewaySenderEventProcessor` processors (each one in a separated thread). The number of processors is defined when the serial gateway sender is created with the `dispatcher-threads` attribute. Currently all of them are started in parallel. The change will be that one thread will be initialized first. Once it is connected, the receiver member id will be known, so it can be notified to the rest of dispatcher threads. After that, the rest of dispatcher threads can be started in parallel. While getting a connection to the receiver, they will check if the member id of the receiver they are connected to is the expected. If not, they could retry the connection to ensure they are connected to the same receiver than the first dispatcher thread.

As this change in the initialization of the threads is only needed on a very specific use case, a new boolean `--same-receiver-id` option will be added to the `create gateway-sender` command to trigger it. If it is not used, the dispatcher threads will be initialized in parallel, as it is currently done .

Changes and Additions to Public Interfaces

N/A

Performance Impact

As the new way to initialize the threads will be optional and false by default, it will not have an impact on most of the cases. Only in the scenario where this change applies (using several receivers with same host and port), serial gateway sender initialization could take longer due to the first thread is started in first place and due to the rest of threads could do some retries when connecting to the right server.

Prior Art

GEODE-7565 already introduced changes to allow the usage of same host and port in several gateway receivers, in that case the changes allowed locators to know they have several servers with that configuration, so if one goes down, replication is not considered down.

The change proposed in this RFC allows the usage of more than one dispatcher thread, increasing performance.

FAQ