semisynchronous replica acknowledges that it has received all events for the transaction, or until a timeout occurs.

- The replica acknowledges receipt of a transaction's events only after the events have been written to its relay log and flushed to disk.
- If a timeout occurs without any replica having acknowledged the transaction, the source reverts to asynchronous replication. When at least one semisynchronous replica catches up, the source returns to semisynchronous replication.
- Semisynchronous replication must be enabled on both the source and replica sides. If semisynchronous replication is disabled on the source, or enabled on the source but on no replicas, the source uses asynchronous replication.

While the source is blocking (waiting for acknowledgment from a replica), it does not return to the session that performed the transaction. When the block ends, the source returns to the session, which then can proceed to execute other statements. At this point, the transaction has committed on the source side, and receipt of its events has been acknowledged by at least one replica. The number of replica acknowledgments the source must receive per transaction before returning to the session is configurable using the `rpl_semi_sync_master_wait_for_slave_count` system variable, for which the default value is 1.

Blocking also occurs after rollbacks that are written to the binary log, which occurs when a transaction that modifies nontransactional tables is rolled back. The rolled-back transaction is logged even though it has no effect for transactional tables because the modifications to the nontransactional tables cannot be rolled back and must be sent to replicas.

For statements that do not occur in transactional context (that is, when no transaction has been started with `START TRANSACTION` or `SET autocommit = 0`), autocommit is enabled and each statement commits implicitly. With semisynchronous replication, the source blocks for each such statement, just as it does for explicit transaction commits.

The `rpl_semi_sync_master_wait_point` system variable controls the point at which a semisynchronous source server waits for replica acknowledgment of transaction receipt before returning a status to the client that committed the transaction. These values are permitted:

- `AFTER_SYNC` (the default): The source writes each transaction to its binary log and the replica, and syncs the binary log to disk. The source waits for replica acknowledgment of transaction receipt after the sync. Upon receiving acknowledgment, the source commits the transaction to the storage engine and returns a result to the client, which then can proceed.
- `AFTER_COMMIT`: The source writes each transaction to its binary log and the replica, syncs the binary log, and commits the transaction to the storage engine. The source waits for replica acknowledgment of transaction receipt after the commit. Upon receiving acknowledgment, the source returns a result to the client, which then can proceed.

The replication characteristics of these settings differ as follows:

- With `AFTER_SYNC`, all clients see the committed transaction at the same time, which is after it has been acknowledged by the replica and committed to the storage engine on the source. Thus, all clients see the same data on the source.

  In the event of source failure, all transactions committed on the source have been replicated to the replica (saved to its relay log). An unexpected exit of the source and failover to the replica is lossless because the replica is up to date. As noted above, the source should not be reused after the failover.