Automatic activation design - draft

Preconditions

Cluster activation feature is very handy when working together with new persistence feature introduced in Apache Ignite 2.1.

However it has its drawbacks:

- User has to activate cluster manually after each restart (or build custom automation tools for this).
- Taking out nodes from cluster for planned maintenance always triggers partition rebalancing and thus slows down system throughput.
- Specific scenarios become possible causing two histories of the same data being created.
  For example, user may start cluster of two nodes: A, B, add some data; then stop B, modify data on A only; then stop A, start B, modify the same data on B only. After that nodes A and B will have different versions of the same data impossible to be merged automatically.

Design

BaselineTopology

It is possible to address these items by introducing new concept of BaselineTopology (working name; other options: RestartNodeSet, FixedNodeSet, MinimalNodeSet). The main idea is to have a fixed set of nodes that are expected to be in the cluster.

BaselineTopology (BLT) is created and changed by command from user (or automatically in special cases, see Use Cases section). Nodes joining or leaving cluster don't affect BLT.

For example, user may create a cluster of ten nodes and active it. On activation BLT is established, partitions are assigned to nodes based on BLT and not on actual topology.
When nodes join or leave cluster, no rebalancing happens because BLT stays the same.

Integration with activation functionality

In current activation implementation new node joining cluster automatically becomes active as well. With BLT we will end up in situation when there are two kind of "active" nodes in the cluster: active in BLT (have partitions assigned to them and stores actual data) and active out of BLT (don't have any partitions).
To have "active" status consistent I propose to introduce new status with working name "active_empty" which means that node in this status has all components started and initialized, but is not visible to affinity function and thus doesn't store any data.

BaselineTopology Versioning

BLT needs to be versioned to address the issue with different histories of the same data (see last example in Preconditions section). When old node with a BLT tries to join an existing cluster, version of its BLT and current BaselineTopology are compared. If they mismatch cluster refuses to join such node to avoid data corruption and undefined behavior.

Use cases

1. User creates new BLT using WebConsole or other tool and "applies" it to brand-new cluster.

2. User starts up brand-new cluster with desired amount of nodes and activates it. At the moment of activation BLT is created with all server non-daemon nodes presented in the cluster.

3. User starts up a cluster with previously prepared BLT -> when set of nodes in the cluster matches with BLT cluster gets automatically activated.

4. User has an up-and-running active cluster and starts a few more nodes. They join the cluster but no partitions are assigned to them.
   User recreates BLT on new cluster topology -> partitions are assigned to new nodes.

5. User takes out nodes from cluster (e.g. for maintenance purposes): no rebalance happens until user recreates BLT on new cluster topology.

6. If some parameters reach critical levels (e.g. number of backups for a partition is too low) coordinator automatically recreates BLT and thus triggers rebalancing.