Maintaining a Quorum throughout the lifecycle of your Elasticsearch cluster

Konrad Beiske, konrad@found.no
@foundsays
About me

• Developer at Found
• At Found we got hundreds of Elasticsearch clusters; big and small.
• Growing and shrinking them becomes daily routine
Agenda

- A little theory
  - Split brain
  - Quorum
- Managing an Elasticsearch cluster
  - In theory (How you should do it)
  - In practice (How you end up doing it)
  - At Found (How we do it)
What does a Split Brain look like?

![Diagram showing a split brain example](image-url)

1) PUT /doc1
2) OK
3) GET /doc1
4) Not found

Node A
master

Node B
master

Load Balancer
Why does it happen?
Quorum

• A quorum is any set of nodes larger than the quorum limit.
• The solution against split brains
• floor(n/2) + 1
Quorum
What about a two node cluster?

• $Q = \text{Floor}(N/2)+1$
• $N=2 \rightarrow Q=2$
• Consequence: Two Elasticsearch nodes does not make a highly available cluster.
• Solution: Add a third master only node.
Quorum limit in Elasticsearch.

- `discovery.zen.minimum_master_nodes`
  - `elasticsearch.yml`
  - `localhost:9200/_cluster/settings`

- Used in master election
Example: Add two nodes

- Start: N=5, Q=3
- Target: N=7, Q=4
- Solution:
  - Set minimum master nodes = 4
  - Add two nodes
Example: Replace nodes

- Start: N=3, Q=2 Slow boxes
- Target: N=3, Q=2 Fast boxes
- Growing and shrinking:
  - Set minimum master nodes = 3
  - Add two nodes (N=5)
  - Set minimum master nodes = 4
  - Add last node (N=6)
  - Migrate data
  - Take one old node down (N=5)
  - Set minimum master nodes = 3
  - Take two old nodes down (N=3)
  - Set minimum master nodes = 2
Example: Replace nodes

• Start: N=3, Q=2 Slow boxes
• Target: N=3, Q=2 Fast boxes
• Growing and shrinking:
  – Set minimum master nodes = 3
  – Add two nodes (N=5)
  – Set minimum master nodes = 4
  – Add last node (N=6)
  – Migrate data
  – Take one old node down (N=5)
  – Set minimum master nodes = 3
  – Take two old nodes down (N=3)
  – Set minimum master nodes = 2
Example: Replace nodes

- **Start:** N=3, Q=2 Slow boxes
- **Target:** N=3, Q=2 Fast boxes
- **Growing and shrinking:**
  - Set minimum master nodes = 3
  - **Add two nodes (N=5)**
  - Set minimum master nodes = 4
  - Add last node (N=6)
  - Migrate data
  - Take one old node down (N=5)
  - Set minimum master nodes = 3
  - Take two old nodes down (N=3)
  - Set minimum master nodes = 2
Example: Replace nodes

- Start: N=3, Q=2 Slow boxes
- Target: N=3, Q=2 Fast boxes
- Growing and shrinking:
  - Set minimum master nodes = 3
  - Add two nodes (N=5)
  - Set minimum master nodes = 4
  - Add last node (N=6)
  - Migrate data
  - Take one old node down (N=5)
  - Set minimum master nodes = 3
  - Take two old nodes down (N=3)
  - Set minimum master nodes = 2
Example: Replace nodes

- Start: N=3, Q=2 Slow boxes
- Target: N=3, Q=2 Fast boxes
- Growing and shrinking:
  - Set minimum master nodes = 3
  - Add two nodes (N=5)
  - Set minimum master nodes = 4
  - Add last node (N=6)
  - Migrate data
  - Take one old node down (N=5)
  - Set minimum master nodes = 3
  - Take two old nodes down (N=3)
  - Set minimum master nodes = 2
Example: Replace nodes

• Start: $N=3$, $Q=2$ Slow boxes
• Target: $N=3$, $Q=2$ Fast boxes
• Growing and shrinking:
  – Set minimum master nodes = 3
  – Add two nodes ($N=5$)
  – Set minimum master nodes = 4
  – Add last node ($N=6$)
  – **Migrate data**
  – Take one old node down ($N=5$)
  – Set minimum master nodes = 3
  – Take two old nodes down ($N=3$)
  – Set minimum master nodes = 2
Example: Replace nodes

- Start: N=3, Q=2 Slow boxes
- Target: N=3, Q=2 Fast boxes
- Growing and shrinking:
  - Set minimum master nodes = 3
  - Add two nodes (N=5)
  - Set minimum master nodes = 4
  - Add last node (N=6)
  - Migrate data
  - Take one old node down (N=5)
  - Set minimum master nodes = 3
  - Take two old nodes down (N=3)
  - Set minimum master nodes = 2
Example: Replace nodes

• Start: N=3, Q=2 Slow boxes
• Target: N=3, Q=2 Fast boxes
• Growing and shrinking:
  – Set minimum master nodes = 3
  – Add two nodes (N=5)
  – Set minimum master nodes = 4
  – Add last node (N=6)
  – Migrate data
  – Take one old node down (N=5)
  – **Set minimum master nodes = 3**
  – Take two old nodes down (N=3)
  – Set minimum master nodes = 2
Example: Replace nodes

- Start: $N=3$, $Q=2$ Slow boxes
- Target: $N=3$, $Q=2$ Fast boxes
- Growing and shrinking:
  - Set minimum master nodes = 3
  - Add two nodes ($N=5$)
  - Set minimum master nodes = 4
  - Add last node ($N=6$)
  - Migrate data
  - Take one old node down ($N=5$)
  - Set minimum master nodes = 3
  - **Take two old nodes down** ($N=3$)
  - Set minimum master nodes = 2

Node D
Node E
Node F
Example: Replace nodes

• Start: N=3, Q=2 Slow boxes
• Target: N=3, Q=2 Fast boxes
• Growing and shrinking:
  – Set minimum master nodes = 3
  – Add two nodes (N=5)
  – Set minimum master nodes = 4
  – Add last node (N=6)
  – Migrate data
  – Take one old node down (N=5)
  – Set minimum master nodes = 3
  – Take two old nodes down (N=3)
  – Set minimum master nodes = 2
Example: Replace nodes

- Start: N=3, Q=2 Slow boxes
- Target: N=3, Q=2 Fast boxes
- Growing and shrinking:
  - Set minimum master nodes = 3
  - Add two nodes (N=5)
  - Set minimum master nodes = 4
  - Add last node (N=6)
  - Migrate data
  - Take one old node down (N=5)
  - Set minimum master nodes = 3
  - Take two old nodes down (N=3)
  - Set minimum master nodes = 2
Example: Replace nodes

- Start: $N=3$, $Q=2$ Slow boxes
- Target: $N=3$, $Q=2$ Fast boxes
- Growing and shrinking:
  - Set minimum master nodes = 3
  - Add two nodes ($N=5$)
  - Set minimum master nodes = 4
  - Add last node ($N=6$)
  - Migrate data
  - Take one old node down ($N=5$)
  - Set minimum master nodes = 3
  - Take two old nodes down ($N=3$)
  - Set minimum master nodes = 2
Example: Replace nodes

• Start: N=3, Q=2 Slow boxes
• Target: N=3, Q=2 Fast boxes
• Rolling upgrade:
  – Stop one old node (N=2)
  – Add a new node (N=3)
  – Wait for replicas to restore and repeat for each old node
Example: Replace nodes

• Start: N=3, Q=2 Slow boxes
• Target: N=3, Q=2 Fast boxes
• Rolling upgrade:
  – Stop one old node (N=2)
  – Add a new node (N=3)
  – Wait for replicas to restore and repeat for each old node
Example: Replace nodes

- **Start:** $N=3$, $Q=2$ Slow boxes
- **Target:** $N=3$, $Q=2$ Fast boxes
- **Rolling upgrade:**
  - Stop one old node ($N=2$)
  - **Add a new node ($N=3$)**
  - Wait for replicas to restore and repeat for each old node
Example: Replace nodes

• Rolling upgrade is simpler, but:
  – Looses HA for a larger period
  – Reduces capacity in cluster
Failures while migrating

- Network errors
- OutOfMemoryErrors
- Version mismatches
- Bad configuration on new nodes
Node crashes during migration

- Number of nodes larger than Quorum limit:
  - Migrations involving node will fail and restart from other replicas.
- Number of nodes less than Quorum limit:
  - Master is demoted and cluster is left disconnected.
Example: Migration failure

- New nodes continuously run out of memory
  - New plugin requires too much memory
  - Customer tries to downscale
Example: Migration failure

- No quorum
- Cluster disconnected
- No master
- How do you recover?
Automated recovery

• Create timeout depending on the size of the cluster
• Rollback to last known good state
Migration at Found

- Automated
- Need automated rollback too
- A modified grow and shrink
  - Don’t temporarily increase quorum limit while migrating.
Migration at Found

• Start: N=3, Q=2 Slow boxes
• Target: N=3, Q=2 Fast boxes
• Automated with rollback
  – Add all new nodes (N=6)
  – Monitor master
  – Send queries via new nodes
  – Migrate data
  – Take down old nodes
Migration at Found

N=6
Q=2

ZooKeeper

Master is A

Node A
Node B
Node C
Node D
Node E
Node F

Proxy

Master is A
Master is A
Master is A
Future: Snapshot & Restore

- Take snapshot
- Stop indexing
- Take snapshot (incremental)
- Recover last snapshot to new cluster
- Start indexing to new cluster
Other tips and tricks

• Use dedicated nodes for master and data
  – Reduces the risk of a master node running out of memory
• Monitor memory pressure
• Upgrade before it’s too late
• Use persistent connections
• Use a client capable of discovering new nodes
Summary

- Correct quorum limit is really important
- Split brain (multiple masters) is bad
- Make a plan for failover/rollback
- Automated approaches and manual approaches can be different.
Questions?