If one of your Elasticsearch data nodes is full while the other still has space, it suggests that your shard allocation settings might not be distributing data evenly. Here’s how you can ensure balanced data distribution across your two data nodes:

**1. Check Current Disk Usage & Shard Allocation**

Run the following command to check the disk usage and shard allocation status:

sh

CopyEdit

GET \_cat/allocation?v

This will show how shards are distributed across the nodes.

**2. Check and Adjust Disk-Based Shard Allocation**

Elasticsearch uses **watermarks** to prevent nodes from getting overloaded:

* **Low watermark (cluster.routing.allocation.disk.watermark.low)** – Default: 85% (starts avoiding placing new shards)
* **High watermark (cluster.routing.allocation.disk.watermark.high)** – Default: 90% (tries to relocate shards)
* **Flood stage watermark (cluster.routing.allocation.disk.watermark.flood\_stage)** – Default: 95% (makes index read-only)

Check your settings:

sh

CopyEdit

GET \_cluster/settings?include\_defaults=true

If the high watermark is too low, Elasticsearch might stop placing shards on the full node too soon.

**3. Modify Watermark Settings (If Needed)**

Increase the limits (use caution):

sh

CopyEdit

PUT \_cluster/settings

{

"persistent": {

"cluster.routing.allocation.disk.watermark.low": "90%",

"cluster.routing.allocation.disk.watermark.high": "95%",

"cluster.routing.allocation.disk.watermark.flood\_stage": "98%"

}

}

This allows Elasticsearch to use more of the available disk before stopping allocations.

**4. Enable Shard Balancing**

Check if automatic rebalancing is enabled:

sh

CopyEdit

GET \_cluster/settings

If cluster.routing.rebalance.enable is set to none, shards won't be redistributed. Enable it with:

sh

CopyEdit

PUT \_cluster/settings

{

"persistent": {

"cluster.routing.rebalance.enable": "all"

}

}

**5. Manually Reroute Shards**

If automatic rebalancing is not happening, you can manually move shards to the less-used node:

sh

CopyEdit

POST \_cluster/reroute

{

"commands": [

{

"move": {

"index": "your-index-name",

"shard": 0,

"from\_node": "node-1",

"to\_node": "node-2"

}

}

]

}

**6. Ensure Evenly Distributed Index Shards**

If your indices have too few primary shards, all data might be allocated to one node. Check the shard count:

sh

CopyEdit

GET \_cat/indices?v

If needed, increase the number of primary shards when creating an index:

sh

CopyEdit

PUT my\_index

{

"settings": {

"number\_of\_shards": 2,

"number\_of\_replicas": 1

}

}

This ensures both nodes share the data.

**7. Check Node Attributes & Allocation Awareness**

If your cluster has node.attr settings, check if data is being allocated based on specific attributes:

sh

CopyEdit

GET \_nodes/\_all/settings

If needed, adjust allocation rules.

**Summary:**

* Check current allocation (\_cat/allocation)
* Modify disk watermark settings if needed
* Enable automatic shard rebalancing (cluster.routing.rebalance.enable)
* Manually reroute shards if required
* Ensure indices have a balanced number of shards
* Verify no custom allocation rules (node.attr settings)