### **Elasticsearch Geo-Distributed Setup and Implementation**

This guide explains how to set up and replicate an Elasticsearch geo-distributed cluster with sharding, replication, and zone awareness.

### **1. Prerequisites**

#### **Software and Tools**

* **Elasticsearch (7.x+ recommended)**
* **Kibana**: For visualizing data (optional).
* **Python (3.x)**: For query testing and Flask-based UI (optional).

#### **Hardware/VMs for Deployment**

* Minimum of 4 nodes:
  + **1 Master Node**
  + **3 Data Nodes** (with shards and replicas).
* Ensure network connectivity between all nodes.

### **2. Setting Up the Elasticsearch Cluster**

#### **Step 1: Download and Configure Elasticsearch**

1. Download Elasticsearch and Kibana.
2. Copy the Elasticsearch folder three times to create multiple nodes.

#### **Step 2: Start Elasticsearch and Kibana**

1. Start Kibana and the Elasticsearch nodes.

Verify the cluster health and node setup using Kibana DevTools or cURL:  
  
GET /\_cluster/health?pretty

GET /\_cat/nodes?v

#### **Step 3: Start Elasticsearch Nodes**

**Node 1 (Master Node)**:  
  
elasticsearch-create-enrollment-token.bat -s node

**Node 2, Node 3, Node 4 (Data Nodes)**: Replace <token> with the appropriate enrollment token for each node:  
  
elasticsearch.bat --enrollment-token <token>

#### **Step 4: Verify Node Setup**

Check the nodes again:

Copy code

GET /\_cat/nodes?v

### **3. Managing Data with Python Scripts**

#### **Users Index**

1. Run the users.py script to upload data.

Verify data upload and shard distribution:  
  
GET /\_cat/shards/users?v&h=index,shard,prirep,state,docs,store,node

Check node allocation:  
  
GET /\_cat/nodes?v&h=id,ip,node.role,master,name

Verify index settings:  
  
GET /users/\_settings

Search all documents:  
  
GET /users/\_search

{

"query": {

"match\_all": {}

}

}

#### **Regional Trends Index**

1. Run the regional\_trends.py script.

Verify shard distribution and settings:  
  
GET /\_cat/shards/regional\_trends?v&h=index,shard,prirep,state,docs,store,node

GET /regional\_trends/\_settings

Search all documents:  
  
GET /regional\_trends/\_search

{

"query": {

"match\_all": {}

}

}

#### **Content Index**

1. Run the content.py script.

Verify shard distribution and settings:  
  
GET /\_cat/shards/content?v&h=index,shard,prirep,state,docs,store,node

GET /content/\_settings

Search all documents:  
  
GET /content/\_search

{

"query": {

"match\_all": {}

}

}

#### **Interaction History Index**

1. Run the interaction\_history.py script.

Verify shard distribution and settings:  
  
GET /\_cat/shards/interaction\_history?v&h=index,shard,prirep,state,docs,store,node

GET /interaction\_history/\_settings

Search all documents:  
  
GET /interaction\_history/\_search

{

"query": {

"match\_all": {}

}

}

### **Notes**

* Replace <token> with the actual enrollment token provided during Elasticsearch setup.
* Ensure Elasticsearch and Kibana are running before performing any operations.
* Use Kibana DevTools for quick queries and debugging.

### **Conclusion**

This setup demonstrates how to build and manage a scalable Elasticsearch cluster with multiple nodes, upload data using Python scripts, and verify shard and node configurations.