Data Synchronization in a Docker Swarm Cluster with MQTTS

Michele Montesi
E-Mail: michele.montesi3@studio.unibo.it

June 18, 2023

1 Introduction

This document presents the project of data synchronization in a Docker Swarm cluster that uses the MQTTS protocol. It describes the application context, the specific problem to solve, and the implemented solution, including the technologies and configurations adopted.

2 Application Context

The project starts from a Docker Swarm environment, which is a container orchestration system that allows managing a cluster of Docker nodes. Additionally, it is assumed that an MQTT (Message Queuing Telemetry Transport) messaging service has been configured on all cluster nodes, known as MQTTS (MQTT over TLS/SSL). The use of MQTTS provides secure and encrypted communication between the Swarm cluster nodes.

The project's goal is to achieve a Docker Swarm cluster where the MQTTS service is running on all cluster nodes using the same data. Furthermore, it is necessary to ensure data persistence and synchronization between nodes using the lsyncd utility.

3 Specific Problem

The problem at hand is the need to synchronize data between Docker Swarm cluster nodes. Due to the distributed nature of the cluster with its independent nodes, situations may arise where data is inconsistent or out-of-sync across the nodes. Resolving this problem is essential to ensure the integrity and consistency of data in the Docker Swarm environment.

4 Implemented Solution

To address the data synchronization problem between Docker Swarm cluster nodes, the following solution has been adopted:

4.1 lsyncd

To ensure data synchronization between cluster nodes, the lsyncd (Live Syncing Daemon) utility has been used. lsyncd is a tool that monitors file changes in a directory and automatically propagates them to one or more remote destinations.

A dedicated synchronization rule has been configured in lsyncd to monitor the data directories within each Docker Swarm node and automatically propagate changes to the remaining nodes. This way, any modification made on one node will be automatically and transparently synchronized across all other cluster nodes.

4.2 Ansible

To automate the installation of lsyncd on all Docker Swarm cluster nodes, Ansible has been used. Ansible is an IT automation framework that simplifies application and infrastructure deployment, configuration, and management. By utilizing Ansible playbooks, it was possible to declaratively define and execute the installation and configuration operations of lsyncd on all cluster nodes.

5 Development of Ansible Scripts

The following steps were followed for the development of Ansible scripts used in this project:

5.1 Project Structure

The Ansible project was organized following a standard structure, consisting of the following main elements:

- `inventory.yml`: An inventory file listing all Docker Swarm cluster nodes where lsyncd installation will be performed.
- `lsyncdInstall.yml`: The playbook that handles the installation of the lsyncd package on all Docker Swarm cluster nodes.
- `lsyncdConf.yml`: The main playbook that defines the operations to be executed for lsyncd configuration.
- `roles/`: A directory containing Ansible roles. Each role represents a specific functionality or configuration to be applied on cluster nodes.
- `roles/setup-lsyncd/`: The role that defines lsyncd configuration on each Docker Swarm cluster node.
- `roles/start-lsyncd/`: The role that defines the start of the lsyncd service on each Docker Swarm cluster node.

5.2 Defining Roles

To define Ansible roles for lsyncd configuration and startup, the following activities were performed:

- Creation of directories:
  - `roles/setup-lsyncd/`
  - `roles/start-lsyncd/`
- Definition of a `tasks/main.yml` file within each role. This file contains a list of tasks to install and configure lsyncd.
- Definition of lsyncd configuration files within the `setup-lsyncd` role: `templates/lsyncd.conf.j2`.
- Definition of variable files within `setup-lsyncd` to define the paths to be synchronized.

5.3 Using Playbooks

To perform the installation and configuration of lsyncd on all Docker Swarm cluster nodes, you can execute the following commands:

```
ansible-playbook -i inventory.ini -vv lsyncdInstall.yml
ansible-playbook -i inventory.ini -vv lsyncdConf.yml
```

6 Data Persistence

To ensure data persistence in the Docker Swarm cluster, a Docker volume has been configured for data directories. Docker volumes allow separating data from the container, enabling them to persist even in the event of container or node restart. This guarantees that the data remains available even in failure or maintenance situations. This data will be synchronized using the Ansible script described earlier.

7 Conclusion

This report presented the project of data synchronization in a Docker Swarm cluster with MQTTS using lsyncd. The project was motivated by the data synchronization problem between independent nodes in the Swarm cluster. The implemented solution, the lsyncd utility for automatic data synchronization between nodes, was introduced. The importance of data synchronization in a distributed environment like Docker Swarm was emphasized to ensure data integrity and consistency across cluster nodes.