This assignment consists of two main tasks aimed at exploring different communication paradigms within the Robot Operating System (ROS): asynchronous and synchronous communication.

Task 1: Asynchronous Communication

Definition: Asynchronous communication refers to a communication method where the sender and receiver operate independently of each other, allowing for concurrent execution without the need to wait for each other's response.

In this task, students will implement asynchronous communication between two ROS nodes in Python:

- One node acts as a publisher.
- The other node acts as a subscriber.

Publisher Node:

- Generates random integers.
- Encrypts each integer using the equation: `encrypted_num = num ** 2 + 10`
- Publishes the encrypted data to a specific topic.

Subscriber Node:

- Receives the encrypted data sent by the publisher.
- Decrypts each encrypted integer.
- Logs both the received encrypted data and the final decrypted data to the console.

Task 2: Synchronous Communication

Definition: Synchronous communication refers to a communication method where the sender and receiver coordinate their actions, typically involving waiting for responses or acknowledgments before proceeding.

In this task, students will implement synchronous communication between the publisher and subscriber nodes:

- The publisher will initiate publication with its data but wait until the subscriber receives and responds with the decrypted data.
- Upon receiving the decrypted data from the subscriber, the publisher will compare it with the original data before encryption.
- If the decrypted data matches the original, the publisher will send another message. Otherwise, it will resend the previous encrypted data for decryption.

Submission Requirements:

- For each task, students are required to submit the code for both the publisher and subscriber nodes in Python.
- Additionally, they should provide a 20-second screen recording demonstrating the operation of each task, including encryption, decryption, and communication between the nodes.

Evaluation Criteria:

- Functionality: The code should accurately implement the specified communication paradigms, encryption/decryption logic, and error handling.
- Clarity and Readability: Code should be well-commented and organized, promoting understanding and ease of maintenance.
- Demonstration Video: The screen recording should clearly depict the execution of each task, showcasing successful communication and data processing.