Group Project 1: A Bite of Distributed Communication

CECS 327 – Intro to Networks and Distributed Computing

You should submit the required deliverable materials on BeachBoard by **11:55pm, March 09th (Sunday), 2025**.

1. Project Overview

Objective:

This project will introduce you to distributed communication by requiring you to design and implement a small-scale distributed system using Docker. You will develop and analyze communication protocols to optimize message delivery across nodes in a simulated network.

Key Learning Outcomes:

- Gain hands-on experience in building a distributed system.
- Understand and implement broadcast, anycast, and multicast protocols.
- Use Docker to containerize and deploy distributed nodes.
- Monitor and analyze network traffic.

2. Project Tasks

1. **Docker Containers**:
   - Create 16 Docker containers, divided into two clusters (Cluster A and Cluster B), each with 8 containers.
   - Use Docker Compose or custom scripts to automate the setup and configuration of the clusters.
   - Assign unique IP addresses to each container (e.g., 172.17.0.2 to 172.17.0.17).

2. **Cluster Masters:**
   - Designate one container in each cluster as the cluster master (e.g., 172.17.0.2 for Cluster A and 172.17.0.10 for Cluster B).
   - The cluster master will handle intra-cluster communication and act as a gateway for inter-cluster communication.

3. **Intra-Cluster Communication**

   Implement any **TWO** of following protocols within each cluster:

- Broadcast: Send a message to all containers in the same cluster.
- Anycast: Send a message to the nearest container in the same cluster.
- Multicast: Send a message to a specific group of containers in the same cluster

4. **Inter-Cluster Communication**

   Routing Mechanism:

   - Design a routing algorithm to allow messages to be sent between clusters.
   - Use the cluster masters as gateways for inter-cluster communication.
   - For example:

  - If a container in Cluster A wants to send a message to a container in Cluster B, it will first send the message to its cluster master (172.17.0.2).
  - The cluster master in Cluster A will forward the message to the cluster master in Cluster B (172.17.0.10).
  - The cluster master in Cluster B will deliver the message to the target container.

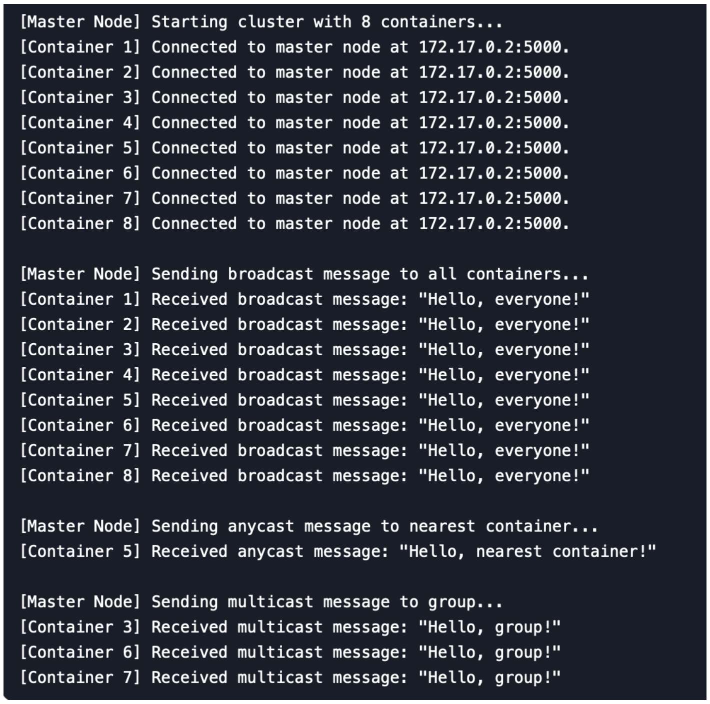
5. Network Monitoring

- Implement **network traffic monitoring** to log and analyze communications between nodes.
- Log all communication activities, including:
  - **Intra-Cluster Communication**: Messages sent within a cluster.
  - **Inter-Cluster Communication**: Messages sent between clusters.
- Use the following format for the log:

Type	**Time (s)**	**Source Cluster**	**Destination Cluster**	**Source IP**	**Destination IP**	**Protocol**	**Length (bytes)**	**Flags (hex)**
Intra-Broadcast	0.0000000	Cluster A	Cluster A	172.17.0.2	172.17.0.3	TCP	128	0x010
Inter-Anycast	0.1234567	Cluster A	Cluster B	172.17.0.2	172.17.0.10	UDP	256	0x011
Intra-Multicast	0.2345678	Cluster B	Cluster B	172.17.0.10	172.17.0.11	TCP	512	0x012

- Use a tool such as **tcpdump, Wireshark, or Python's Scapy** for packet monitoring.
- Store logs in a **CSV or TXT file**

Intel-cluster sample outputs:



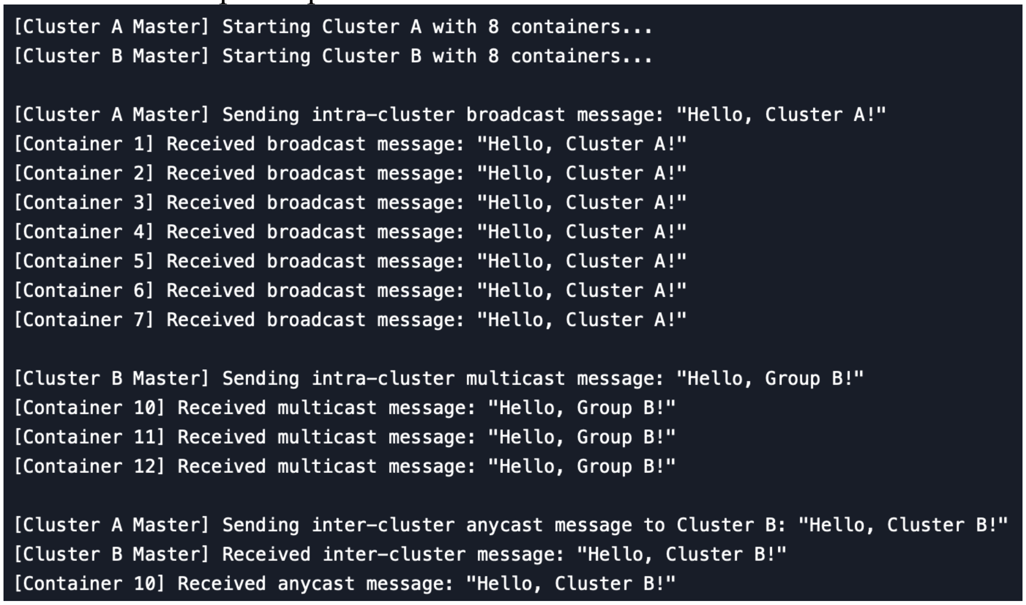
```
[Master Node] Starting cluster with 8 containers...
[Container 1] Connected to master node at 172.17.0.2:5000.
[Container 2] Connected to master node at 172.17.0.2:5000.
[Container 3] Connected to master node at 172.17.0.2:5000.
[Container 4] Connected to master node at 172.17.0.2:5000.
[Container 5] Connected to master node at 172.17.0.2:5000.
[Container 6] Connected to master node at 172.17.0.2:5000.
[Container 7] Connected to master node at 172.17.0.2:5000.
[Container 8] Connected to master node at 172.17.0.2:5000.

[Master Node] Sending broadcast message to all containers...
[Container 1] Received broadcast message: "Hello, everyone!"
[Container 2] Received broadcast message: "Hello, everyone!"
[Container 3] Received broadcast message: "Hello, everyone!"
[Container 4] Received broadcast message: "Hello, everyone!"
[Container 5] Received broadcast message: "Hello, everyone!"
[Container 6] Received broadcast message: "Hello, everyone!"
[Container 7] Received broadcast message: "Hello, everyone!"
[Container 8] Received broadcast message: "Hello, everyone!"

[Master Node] Sending anycast message to nearest container...
[Container 5] Received anycast message: "Hello, nearest container!"

[Master Node] Sending multicast message to group...
[Container 3] Received multicast message: "Hello, group!"
[Container 6] Received multicast message: "Hello, group!"
[Container 7] Received multicast message: "Hello, group!"
```

Intra-cluster sample outputs:



```
[Cluster A Master] Starting Cluster A with 8 containers...
[Cluster B Master] Starting Cluster B with 8 containers...

[Cluster A Master] Sending intra-cluster broadcast message: "Hello, Cluster A!"
[Container 1] Received broadcast message: "Hello, Cluster A!"
[Container 2] Received broadcast message: "Hello, Cluster A!"
[Container 3] Received broadcast message: "Hello, Cluster A!"
[Container 4] Received broadcast message: "Hello, Cluster A!"
[Container 5] Received broadcast message: "Hello, Cluster A!"
[Container 6] Received broadcast message: "Hello, Cluster A!"
[Container 7] Received broadcast message: "Hello, Cluster A!"

[Cluster B Master] Sending intra-cluster multicast message: "Hello, Group B!"
[Container 10] Received multicast message: "Hello, Group B!"
[Container 11] Received multicast message: "Hello, Group B!"
[Container 12] Received multicast message: "Hello, Group B!"

[Cluster A Master] Sending inter-cluster anycast message to Cluster B: "Hello, Cluster B!"
[Cluster B Master] Received inter-cluster message: "Hello, Cluster B!"
[Container 10] Received anycast message: "Hello, Cluster B!"
```

- Discuss findings in your report, including network performance under different test cases.

3. Required Deliverables

1. **README File:** Instructions on how to build, run, and test your Dockerized system.
2. **Source Code:** Include a Makefile (if applicable) and ensure the submission is in the correct format.
3. **Project Report (PDF or Word):**
    - Explanation of the system design.
    - Justification of the chosen protocols.
    - Network performance analysis.
    - Clearly stated contributions of each team member.
4. **Execution Demonstration Video:**
    - Record a video showing your code execution and outputs.
    - The video should display **your name and date** as identification.
    - Upload to **YouTube** (or another platform) and provide a link in your report.

4. Submission Guidelines

- Submit a **single .zip/.rar file** containing all required files.
- Only **one submission per group** is required.
- Ensure your code compiles and runs; otherwise, a zero grade will be assigned.
- If your code is incomplete, specify missing parts in your report for partial credit consideration.
- Provide sufficient **comments in your code** to explain the logic.

5. Grading Criteria

Details	Points
Have a README file shows how to compile and test your submission	5 pts
Submitted code has proper comments to show the design	15 pts
Screen a ***video*** to record code execution and outputs	20 pts
Have a **report** (pdf or word) file explains the details of your entire design	20 pts
Report contains clearly individual contributions of your group mates	5 pts
Code can be compiled and shows correct outputs	35 pts

6. Policies

1. **Late Submissions:** Will be penalized as per course syllabus.
2. **Plagiarism:** Code-level discussions are **prohibited**. Anti-plagiarism tools will be used.
3. **Use of AI Tools: ChatGPT, GPT-4, and similar AI tools are prohibited** for both code and written content.

Final Notes:

- This project requires independent research and problem-solving skills.
- Properly cite any resources you reference.
- Have fun experimenting with distributed systems and networking!

Good luck! 🚀