Master-Thesis Presentation by Özcan Karaca

Testbed-Development for lectureStudio

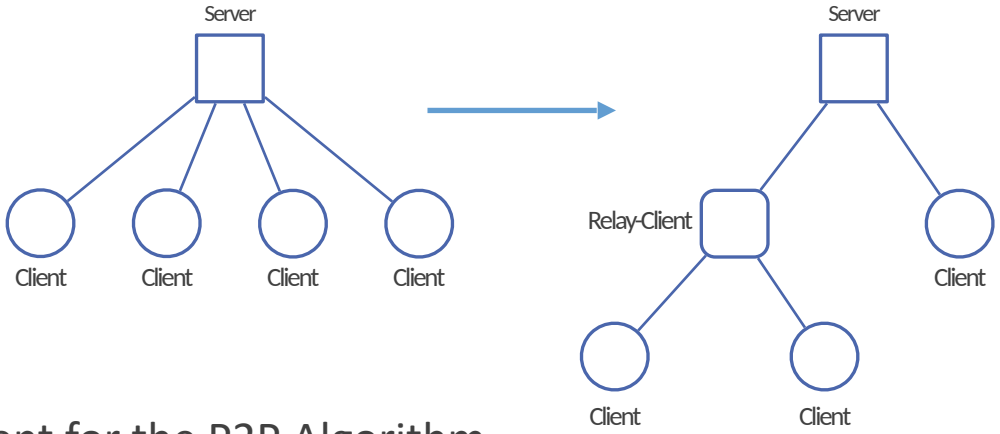
TESTBED-ENTWICKLUNG FÜR LECTURESTUDIO

Fachgebiet Echtzeitsysteme
Fachbereich Elektrotechnik und Informationstechnik
Technische Universität Darmstadt

TECHNISCHE UNIVERSITÄT DARMSTADT

Motivation

- A P2P Algorithm for lectureStudio
  - Direct Distribution Among Clients
  - Reducing the Central Server's Load
  - Optimizing Bandwidth



- Developing A Container-Based Testbed Environment for the P2P Algorithm
  - Simulation of Real Network Data
  - Configuration and Validation of the Network Characteristics
  - Communication and Data Transfer between Nodes

CONTAINERlab

- Performance Evaluation of the P2P Algorithm
  - Analysis of Resource Consumption
  - Analysis of Total Duration

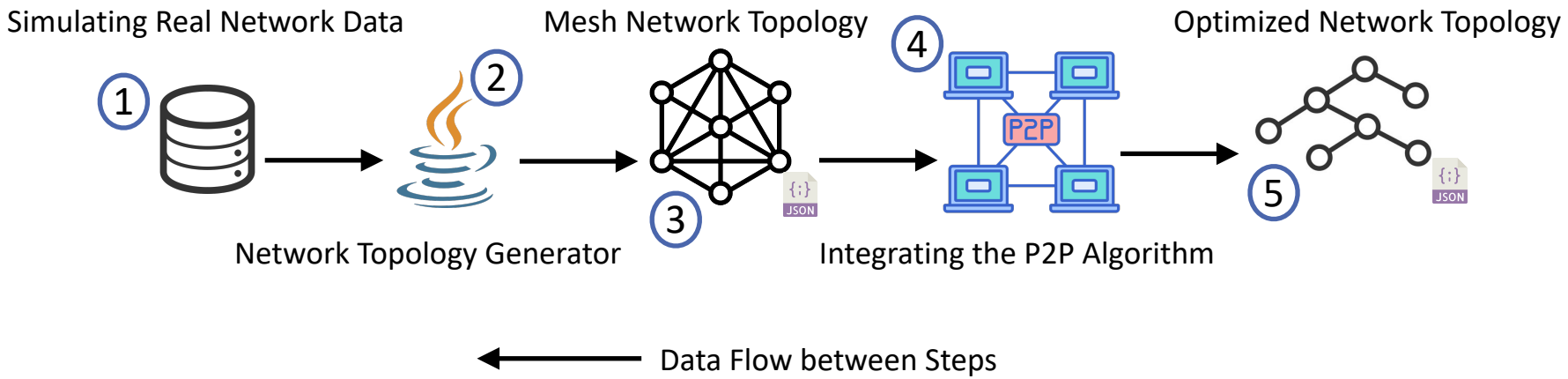
cAdvisor Prometheus Grafana

Task Description

- Finding Real Network Data
  - Maximum Upload Speed, Maximum Download Speed, Latency, and Packet Loss
- Simulating Real Network Data Using Normal Distribution
- Generating Network Topology
- Integrating the P2P Algorithm
- Configuring the Components of the P2P Algorithm in the Testbed
  - Creating a Containerlab File and Configuring Network Management, Nodes and Links
- Implementing and Validating the Network Characteristics of the Connections
  - Bandwidth, Latency, and Packet Loss
- Managing Communication and Data Transfer
  - Tracking and Validating the Data Transfer Process
- Evaluating the Testbed and the P2P Algorithm Performance

Initial Steps (Not Repeated) of the Testbed

- Simulating Real Network Data Using Normal Distribution
- Generating Mesh Network Topology
  - Integrating the P2P Algorithm
  - Implementing the Traditional Server-Client Based Approach
- Calculating Optimized Network Topology with the P2P Algorithm



Simulating Real Network Data

- Analyzing Real Network Dataset
  - Max Download Speed, Max Upload Speed
  - Latency
  - Packet Loss
- Reading Real Network Data from CSV File
- Generating Network Data Using Normal Distribution
  - Having Mean and Standard Deviation from UK and Germany-Based Data

Name of Configuration	Mean (μ)	Standard Deviation (σ)
First Configuration	From UK-Based Data	From UK-Based Data
Second Configuration	From Ger-Based Data	From UK-Based Data

[1], [2]

Network Topology

- Generating Network Topology by the Testbed
  - Listing Nodes with Network Characteristics
  - Detailing Connections between the LectureStudio Server and All Peers
- Optimizing Network Topology by the P2P Algorithm
  - Identifying Super Peers
  - Optimizing Connections between the Studio Server and Peers

Network Topology Generated by the Testbed


```
{
  "filename": "test.pdf",
  "filesize": 5000,
  "peers": [
    {
      "name": "lectureStudioServer",
      "maxDownload": 29150,
      "maxUpload": 9209
    },
    {
      "name": "1",
      "maxDownload": 1080,
      "maxUpload": 373
    }
  ]
}
```

```
"connections": [
    {
      "sourceName": "lectureStudioServer",
      "targetName": "1",
      "bandwidth": 1080,
      "latency": 57,
      "loss": 0.0035
    },
    {
      "sourceName": "1",
      "targetName": "lectureStudioServer",
      "bandwidth": 373,
      "latency": 57.15,
      "loss": 0.0035
    }
  ]
}
```


JSON

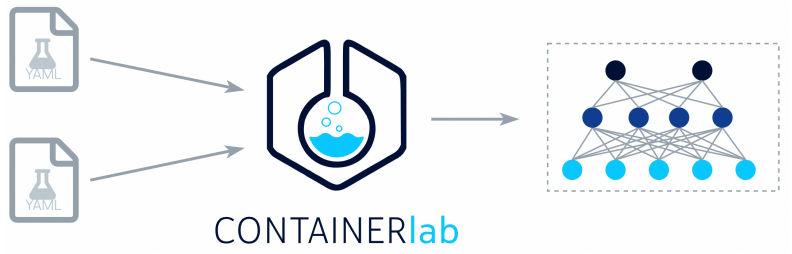
Network Topology Optimized by the P2P Algorithm


```
{
  "superpeers": [
    {
      "name": "1"
    }
  ],
  "peer2peer": [
    {
      "sourceName": "lectureStudioServer",
      "targetName": "1"
    },
    {
      "sourceName": "1",
      "targetName": "2"
    }
  ]
}
```


JSON

Container-Based Testbed Environment

- Docker and Containerlab: Efficient, Isolated Simulation Environment
- Creation and Management of User-Defined Network Topologies
- Advantages of Using Containerlab for the Testbed
  - Properties of Containerlab
    - name, image, kind, env, binds, etc.
  - Speed, Ease of Use, Repeatability
  - Creation of Complex Network Topologies

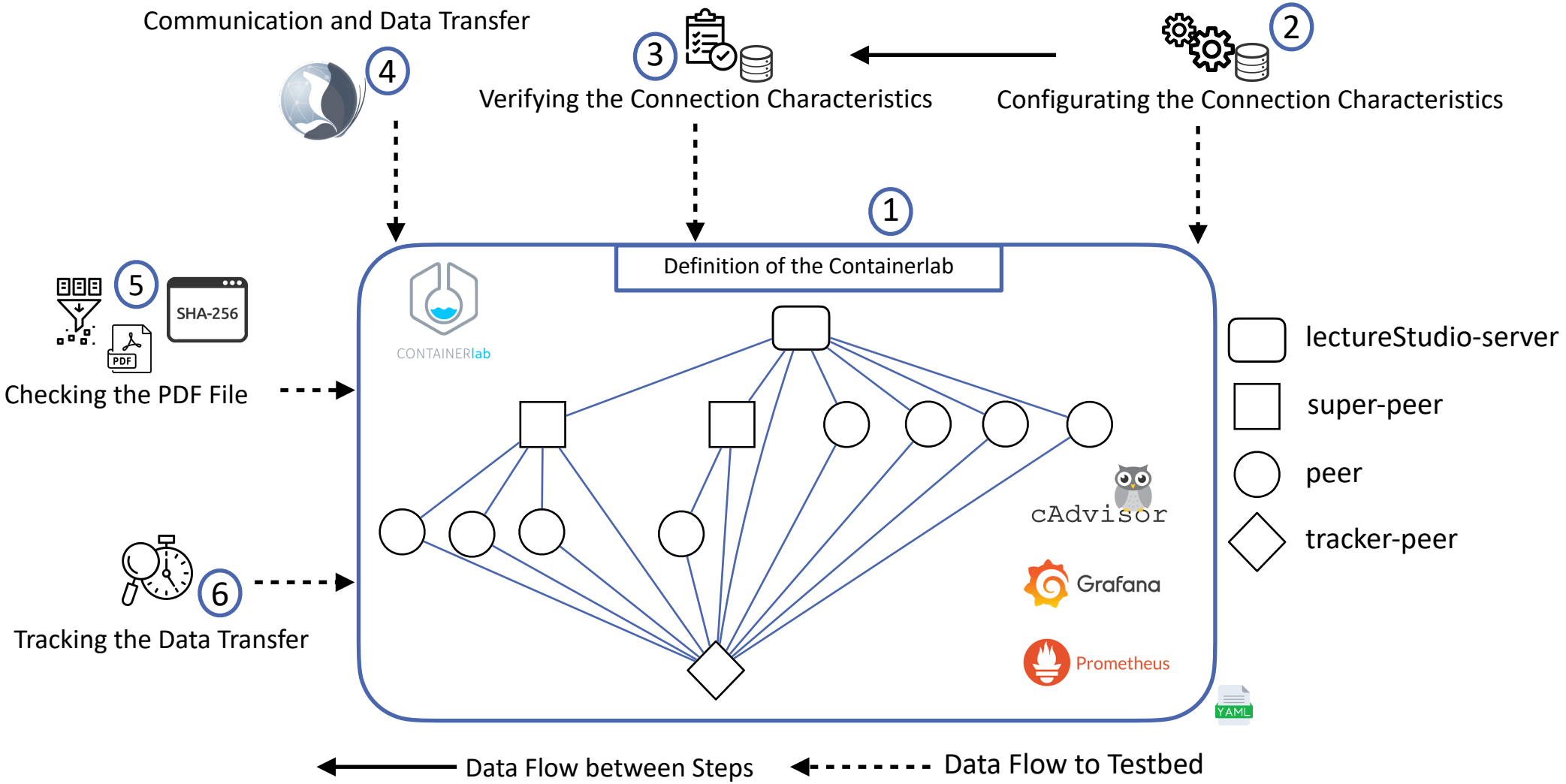


[3]

Configuration of Containerlab File (YAML)

```yaml
name: testbed

topology:
  nodes:
    peer1:
      kind: linux
      image: image-testbed
    peer2:
      kind: linux
      image: image-testbed

  links:
    - endpoints: [peer1:eth1, peer2:eth1]
```

Execution Steps (Repeated) of the Testbed

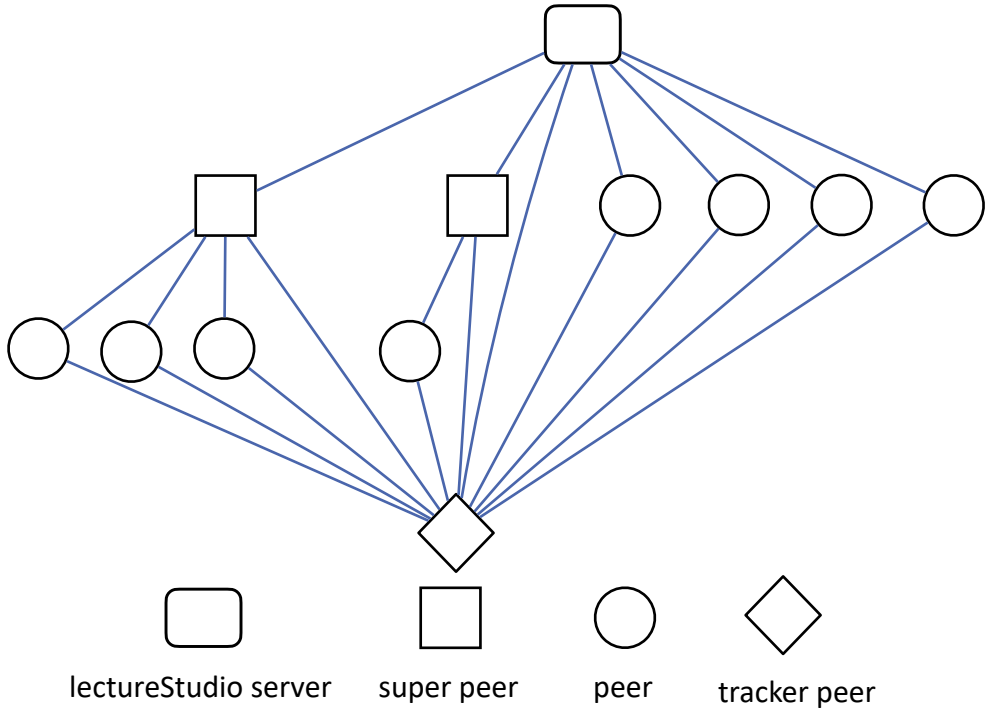


Configuring the Components of the P2P Algorithm

Configuration of Containerlab File (Network Management, Nodes and Links)

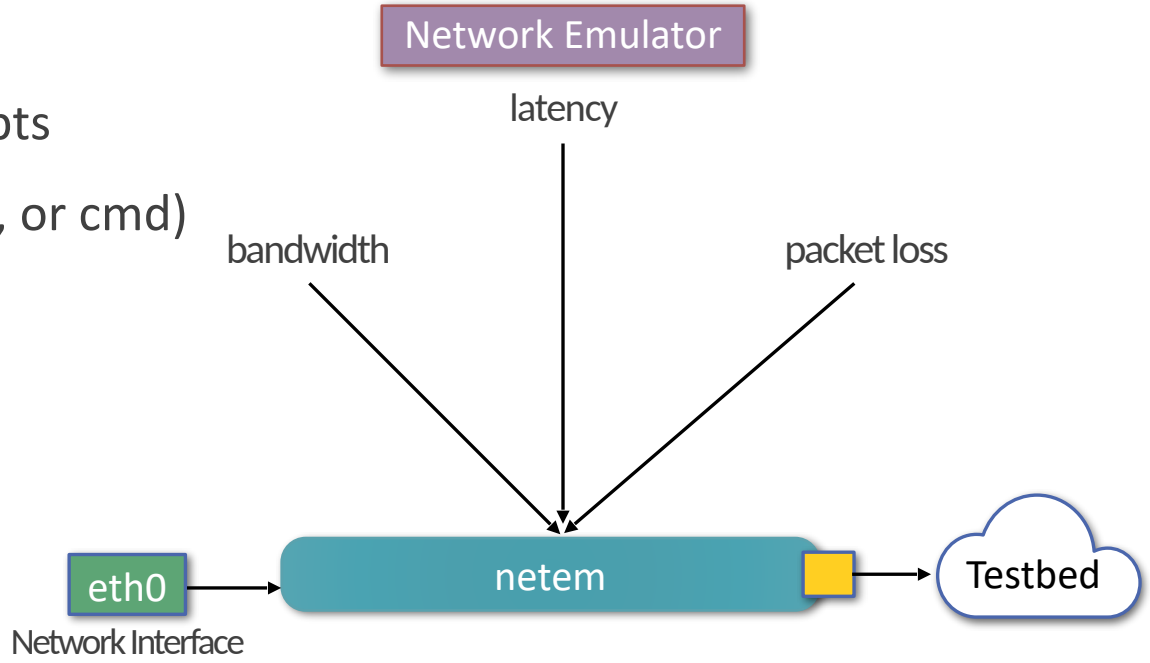
```yaml
name: testbed

mgmt:
  network: fixedips
  ipv4-subnet: 172.100.100.0/24

topology:
   nodes: ...

   links:
     - endpoints: [lectureStudioserver:eth1, 1:eth1]
     - endpoints: [lectureStudioserver:eth2, 2:eth1]
     - endpoints: [lectureStudioserver:eth3, 3:eth1]
     - endpoints: [lectureStudioserver:eth4, 4:eth1]
     - endpoints: [lectureStudioserver:eth5, 5:eth1]
     - endpoints: [lectureStudioserver:eth6, 6:eth1]
     - endpoints: [5:eth2, 7:eth1]
     - endpoints: [6:eth2, 8:eth1]
     - endpoints: [6:eth3, 9:eth1]
     - endpoints: [6:eth4, 10:eth1]
```

Configuration of Nodes



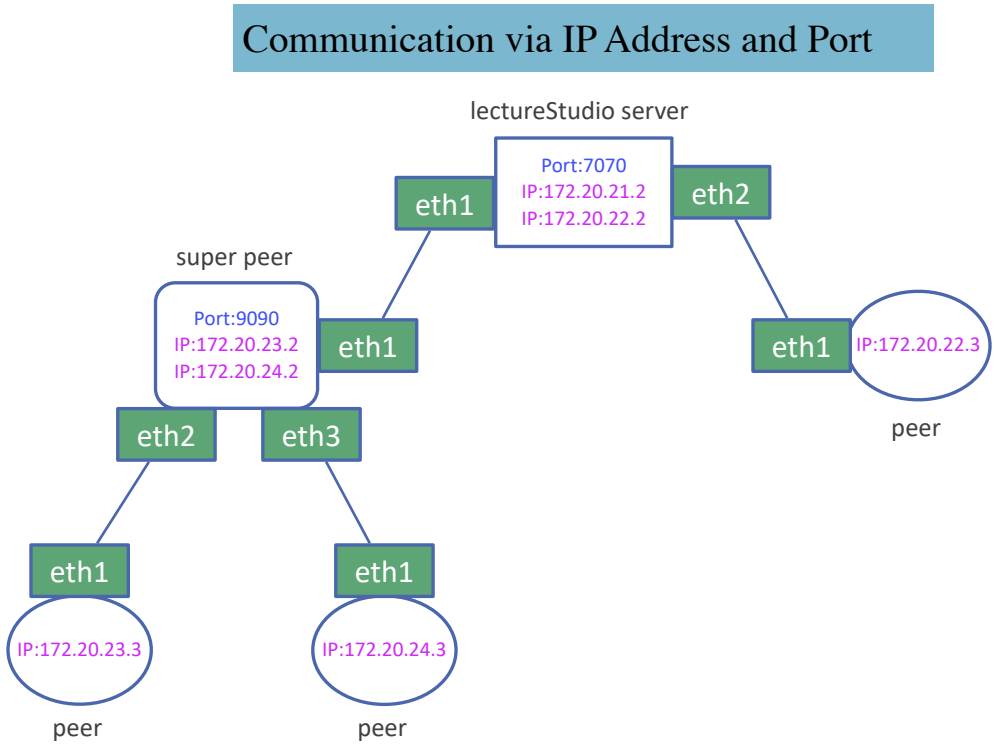
Configuring and Verifying the Network Characteristics

- Configuring the Connection Characteristics Using Traffic Control Commands
  - Bandwidth Limitation
  - Latency Addition
  - Packet Loss Simulation
- Configuring these Characteristics with Scripts
  - Properties of Containerlab (exec, binds, or cmd)
- Verifying the Connection Characteristics
  - Using Tools like ping, iperf3



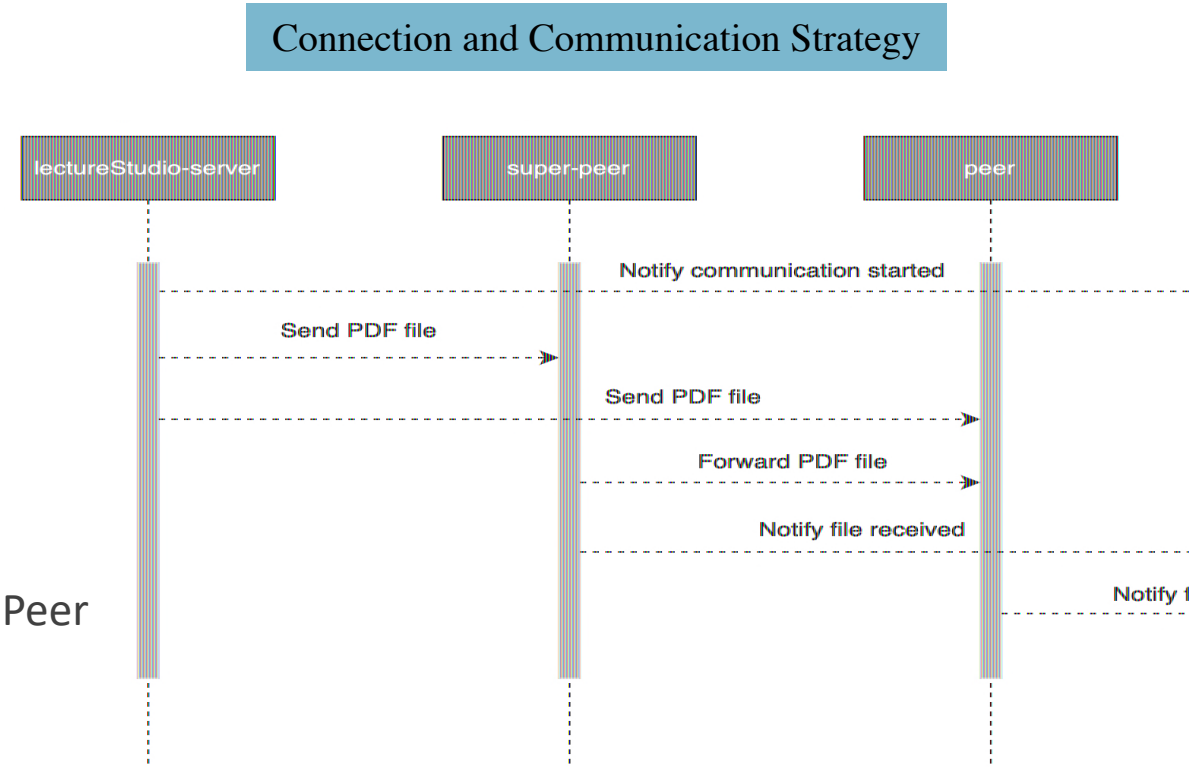
Connection Strategy Among Peers

- Add Node Info in the Testbed Setup
  - Port Number
  - IP Address
- Use Netty Framework for Server
  - Handshaking
  - Authenticating Peers
  - Establishing Connections
  - Preparing the Network for Data Transfer
- Monitor Process and Performance



Communication and Data Transfer Processes

- Initial Notification by the LectureStudio Server
  - Notification to Tracker Peer
  - Start of Data Transfer Process
- Role of Super Peers
  - Reception and Forwarding of PDF File
  - Transition from Receiver to Sender
- Confirmation Messages
  - From Peers and Super Peers
- Calculation of Data Transfer Duration by Tracker Peer
  - Counting Received Confirmations
  - Total Duration Calculation
- Integrity Checks of PDFs with Hash Value Calculation



Evaluation Research Questions

Testbed

RQ1.1 How Accurately Does the Testbed Measure the Configured Bandwidth, Latency, and Packet Loss?

RQ1.2 How Well Does the Testbed Scale with More Nodes and Complex Topologies Affect The Host in Terms of Resource Utilization?

P2P Algorithm

RQ2.1 How Do CPU and Memory Usage Change of the Participants (the LectureStudio Server and Peers) in Tests with and without the P2P Algorithm?

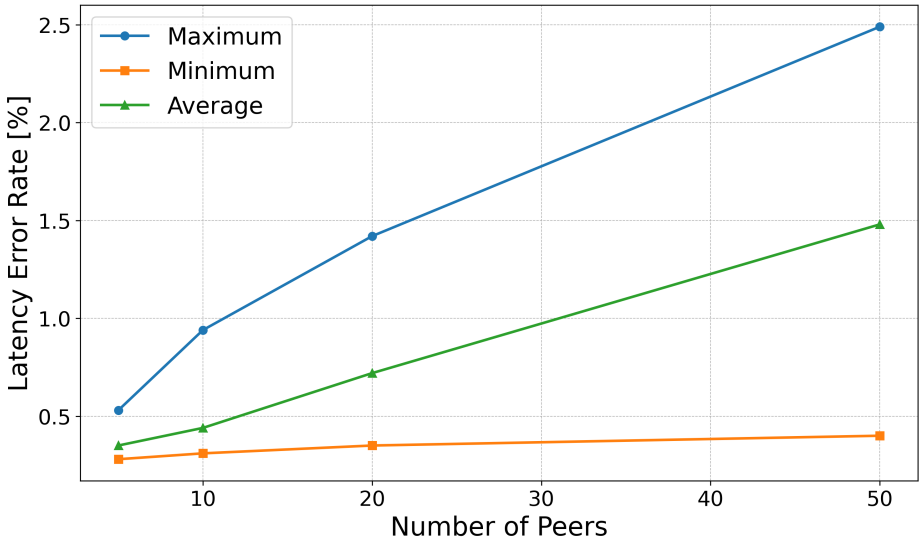
RQ2.2 How Does the P2P Algorithm React to Changing Network Characteristics (Bandwidth, Latency, Packet Loss)?

RQ2.3 Overall, is the P2P Algorithm Efficient for Data Transfer? How Does the Total Duration Obtained by the P2P Algorithm Respond to the Changing Number of Peers and Data Size?

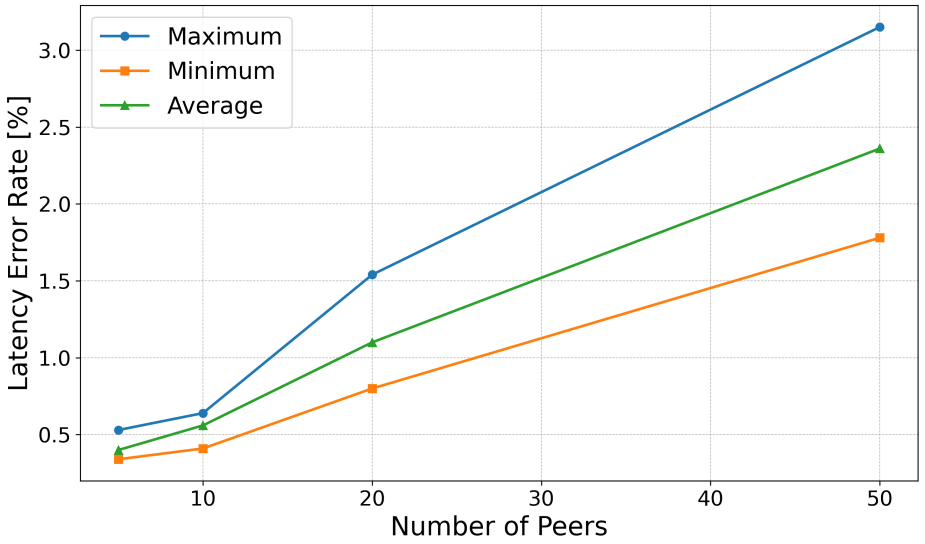
Accuracy of the Testbed, RQ1.1 (1)

- Discrepancy between Configured and Measured Values
- More Nodes Result in Reduced Bandwidth Allocation, Leading to Increased Latency
- Containerization Technologies Increase Latency, Reduce Overheads
- High CPU and Memory Usage, Affecting Network Performance and Latency

Error Rate for Latency with the P2P Algorithm



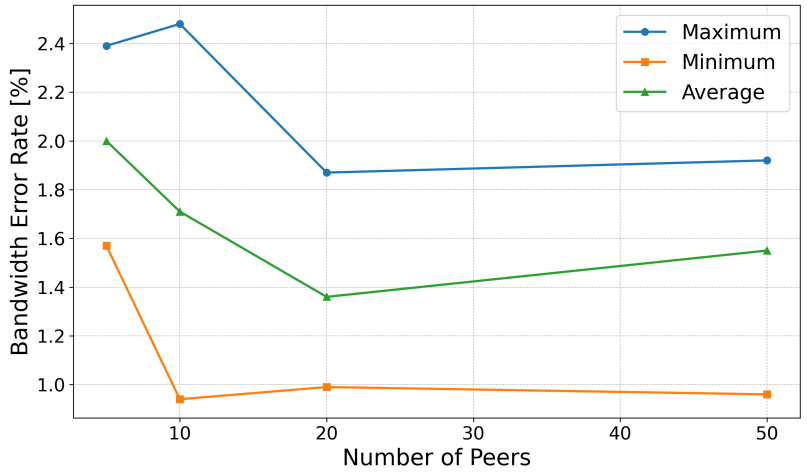
Error Rate for Latency without the P2P Algorithm



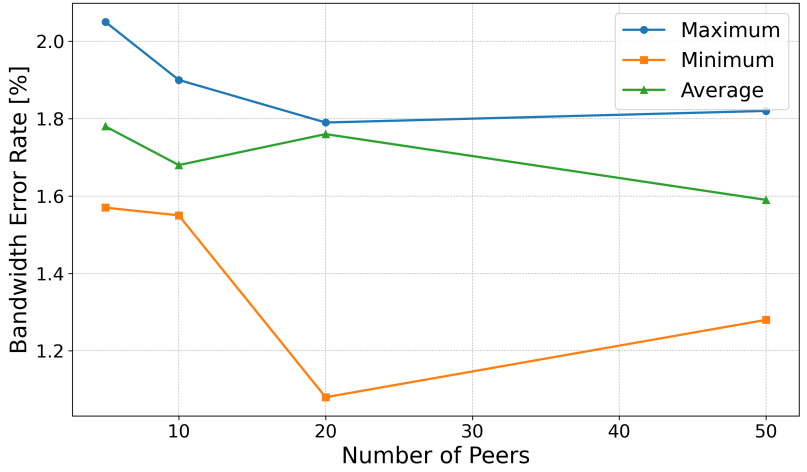
Accuracy of the Testbed, RQ1.1 (2)

- Accuracy of Measuring Tools
  - Iperf Accuracy Results Variation from Actual Performance
  - Variables Like Network Conditions, Configuration, System Overhead

Error Rate for Bandwidth with the P2P Algorithm



Error Rate for Bandwidth without the P2P Algorithm

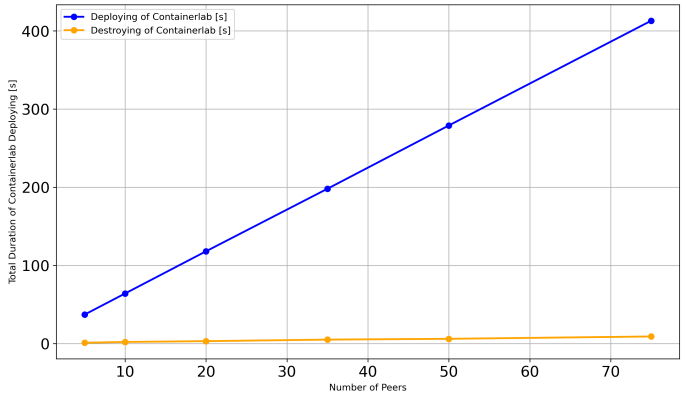


- Testbed Configured with Packet Loss Values from Real Network Data (%0.001)
  - Noted Limitation in Accurately Measuring Packet Loss
  - Increase of Latency with Observed Packet Losses

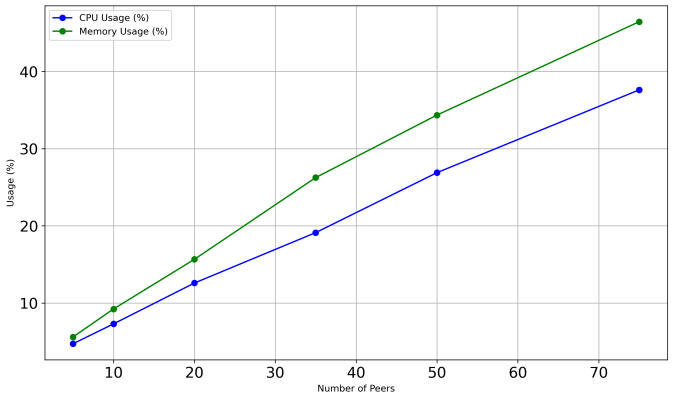
Testbed Scaling and Resource Utilization, RQ1.2

- Scaling Peers with Containerlab for Simplified Topology Deployment and Destroying
  - Deployment Time Increase of 5.37s per Additional Node
  - Destroying Process Time Low, Approximately 9 Seconds for 75 Nodes

Duration of Deploying and Destroying of Containerlab



CPU and Memory Usage of the Host

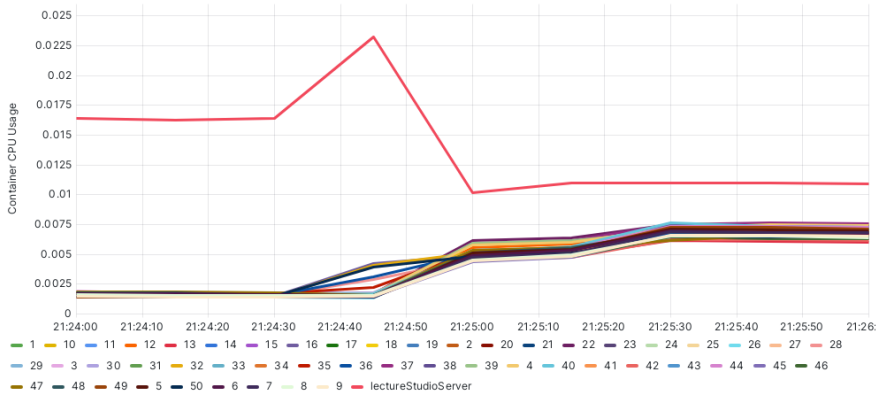


- CPU and Memory Usage Rise with More Peers, Indicating Increased Host Load
  - CPU and Memory Usage Increase Linearly, Showing Scalable Performance
  - Memory Usage Growth Suggests Potential Bottleneck with More Peers

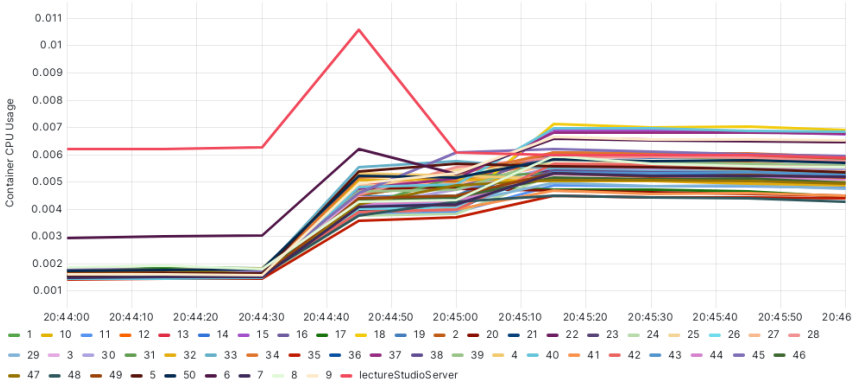
CPU and Memory Usage Analysis of Nodes, RQ2.1 (1)

- Analysis of Resource Consumption (CPU and Memory Usage) of the P2P Algorithm Components (the LectureStudio Server and Peers)
- CPU Usage Evaluation with and without the P2P Algorithm in the Testbed
  - Direct Data Transfer, Increased CPU Usage on the LectureStudio Server
  - P2P Algorithm, Data Distribution Leading to Decreased CPU Usage on the LectureStudio Server
  - Peak CPU Usage Reduction of 52% by the P2P Algorithm

CPU Usage without the P2P Algorithm



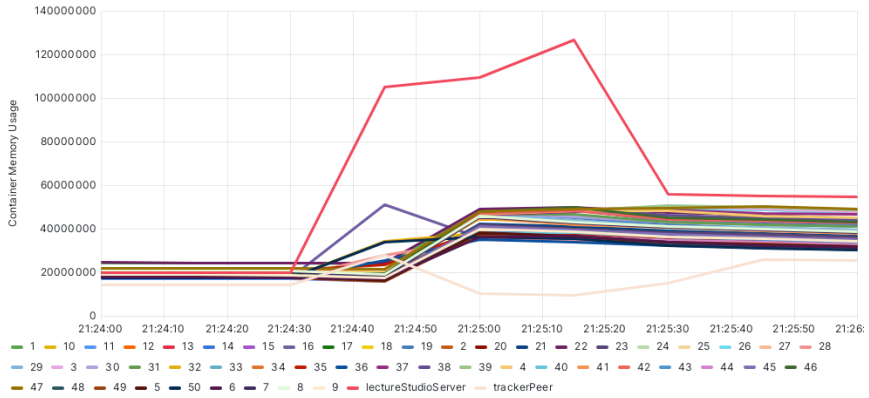
CPU Usage with the P2P Algorithm



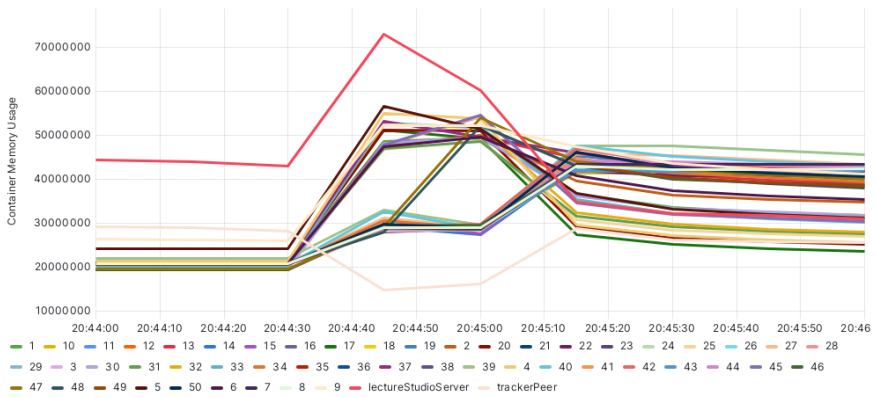
CPU and Memory Usage Analysis of Nodes, RQ2.1 (2)

- Memory Usage Evaluation with and without the P2P Algorithm in the Testbed
  - Direct Data Transfer, Increased Memory Usage on the LectureStudio Server
  - P2P Algorithm, Data Distribution Leading to Decreased Memory Usage on the LectureStudio Server
  - Memory Usage Reduction of 41% by the P2P Algorithm

Memory Usage without the P2P Algorithm



Memory Usage with the P2P Algorithm



- Manageable Increase in CPU and Memory Usage of Super Peers Handling Data Flow

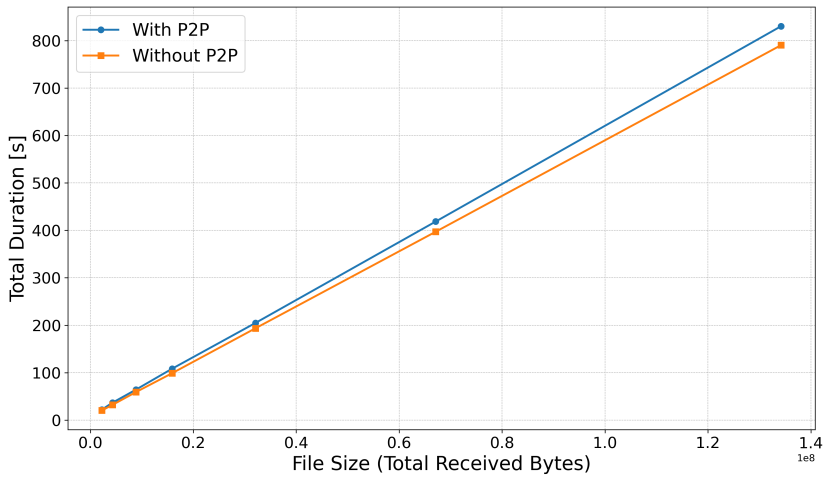
Performance Evaluation of the P2P Algorithm, RQ2.2 and RQ2.3 (1)

- Test Duration Measurement from First to Last Acknowledgment Message
  - Over 1000 Tests Performed
  - Variation in Data Size or Number of Peers
- First Configuration:
  - Simulation of Real Network Data Using Normal Distribution
  - Minimal Difference between the P2P Algorithm and Server-Client Based Approach
  - Optimization by the P2P Algorithm not Significant Advantageous with Small File Sizes
- Second Configuration:
  - Simulation Using Normal Distribution with Different Source for Mean Value (e.g., Germany)
  - Combination of German Mean Values with Real Network Data's Standard Deviation
  - Results with the P2P Algorithm Quite Good in this Configuration
  - Efficiency Increases as Number of Peers and Data Size Grow

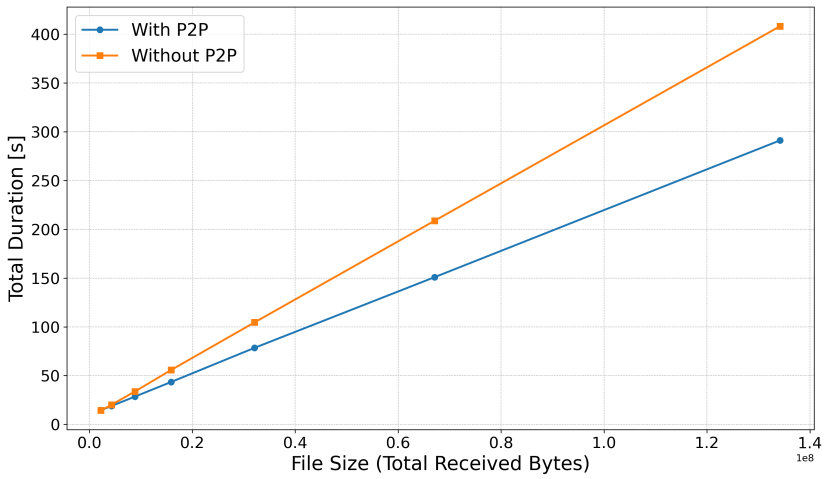
Performance Evaluation of the P2P Algorithm, RQ2.2 and RQ2.3 (2)

- First Configuration with the LectureStudio Server and 20 Peers
  - The P2P Algorithm Performance Little Worse to the Traditional Approach
- Second Configuration with the LectureStudio Server and 20 Peers
  - Transferring 2MB Data Size, the P2P Algorithm Performance Nearly Identical to the Traditional Approach
  - Transferring 128MB Data Size, the P2P Algorithm Performance 27% Faster than the Traditional Approach

Time of Whole Data Transfer with the First Configuration



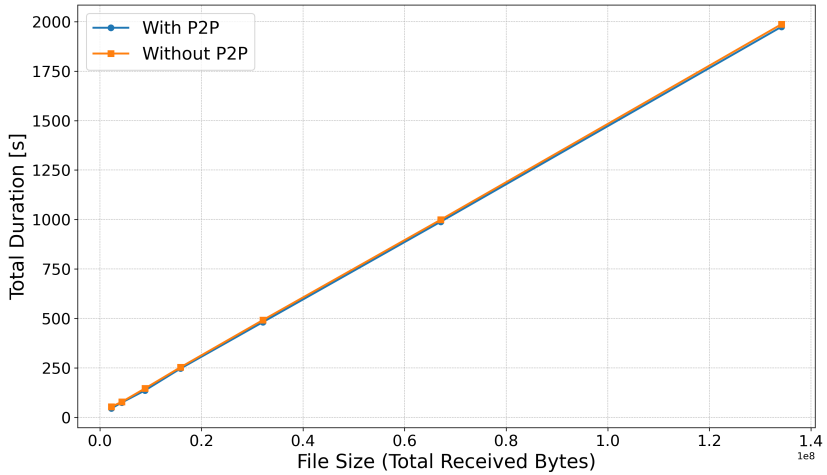
Time of Whole Data Transfer with the Second Configuration



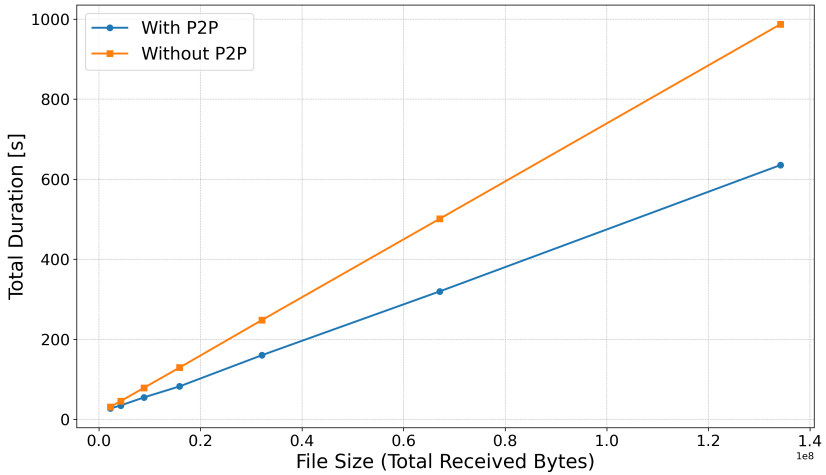
Performance Evaluation of the P2P Algorithm, RQ2.2 and RQ2.3 (3)

- First Configuration with the LectureStudio Server and 50 Peers
  - Transferring 2MB Data Size, the P2P Algorithm Performance Nearly Identical to the Traditional Approach
  - Transferring 128MB Data Size, the P2P Algorithm Performance 5% Faster than the Traditional Approach
- Second Configuration with the LectureStudio Server and 50 Peers
  - Transferring 2MB Data Size, the P2P Algorithm Performance 12% Faster than the Traditional Approach
  - Transferring 128MB Data Size, the P2P Algorithm Performance 35% Faster than the Traditional Approach

Time of Whole Data Transfer with the First Configuration



Time of Whole Data Transfer with the Second Configuration



Conclusion and Future Work

- Goal: Develop a Testbed for the P2P Data Distribution Algorithm
  - Utilization of Docker and Containerlab for An Efficient, Isolated Testing Environment
- Simulation of Real Network Environments and Complex Network Topologies in the Testbed
  - High Replication Accuracy of Bandwidth Limitation and Latency Addition
  - Effective Scalability with Increasing Nodes
- Resource Consumption Performance of the P2P Algorithm
  - High Resource Usage Demand without the P2P Algorithm on the LectureStudio Server
  - Significant Reduction Achieved with the P2P Algorithm on the LectureStudio Server
- Total Duration Performance of the P2P Algorithm
  - Limited Benefits Observed for Small Files
  - Increased Robustness of the P2P Algorithm with Larger Numbers of Peers and Data Sizes
- Future Works
  - Enhancement of Packet Loss Simulation for Accurate Network Behavior
  - Automatic Integration between the Testbed and the P2P Algorithm Optimization
  - Integration of Additional P2P Algorithms into the Testbed
  - Development of A Graphical Testbed Interface for Easier Configuration and Real-Time Analysis

Thank you for your attention!

ANY QUESTIONS?

References

[1] https://www.data.gov.uk/dataset/dfe843da-06ca-4680-9ba0-fbb27319e402/uk-fixed-line-broadband-performance

[2] https://www.speedtest.net/

[3] https://containerlab.dev/manual/topo-def-file/

Challenges

- Finding Real Network Dataset
- Configuration and Validation of the Network Characteristics for Connections
  - Bandwidth Limitation
  - Latency Addition
  - Packet Loss Simulation
- Data Transmission between the LectureStudio Server and Peers (or Super Peers)
- Synchronisation Problem of Total Time
- Monitoring by Grafana, Prometheus, and cAdvisor