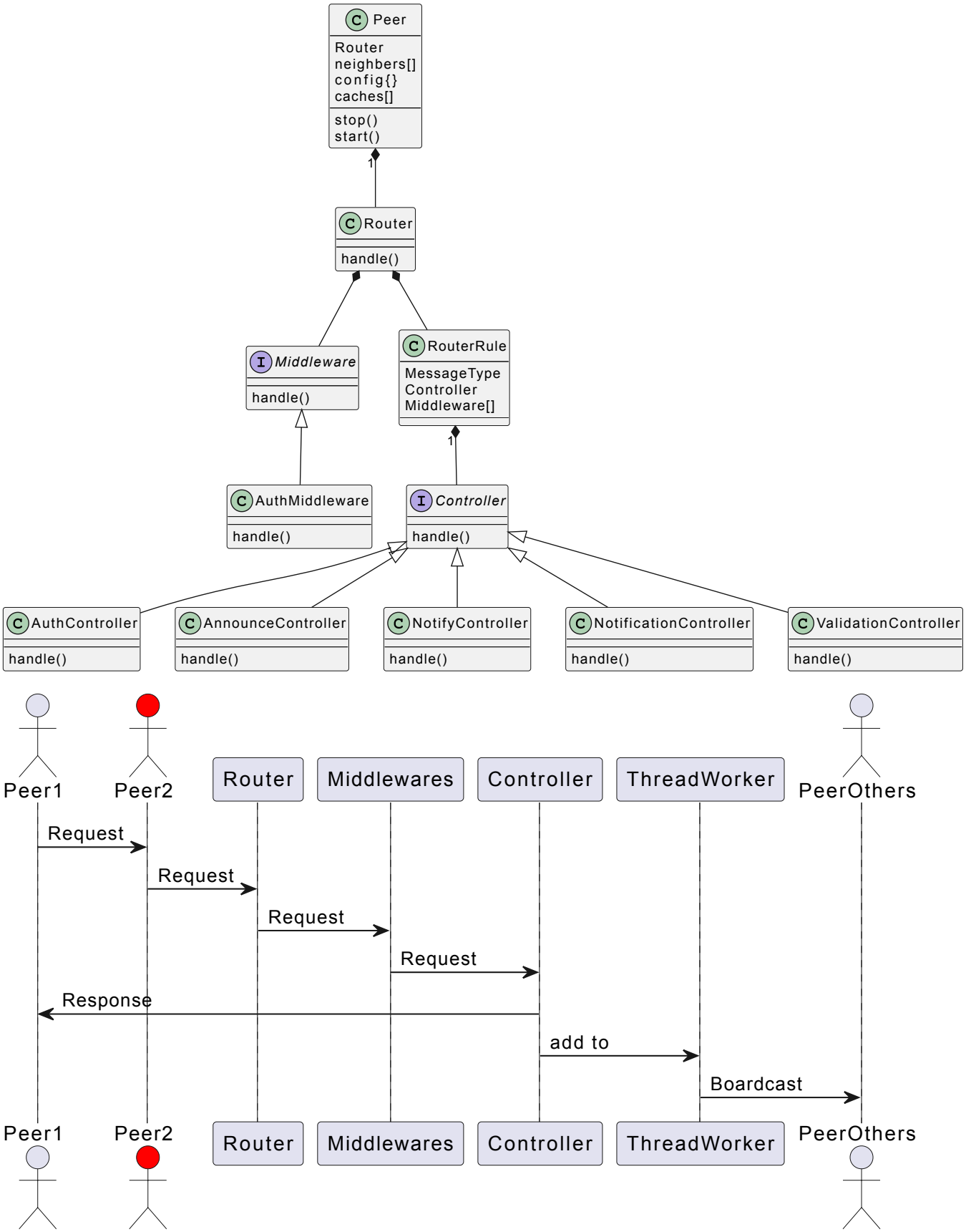
Midterm Report

Changes

No changes.

Architecture of your module

In the architecture we created, each peer is both server and client. After the information entered, the Router handles it with size.



Security measures

We employ three security measures:

Proof-of-work: Our proof of work mechanism builds upon our previous group registration project, incorporating a streamlined approach that focuses solely on a 64-bit challenge with an RSA public key.

Authentication and authorization: To generate peer identities based on network addresses, we can enhance the process by utilizing RSA public keys as unique identifiers. This approach ensures the uniqueness and security of peer identities, aligning with the principles of cryptographic protocols.

Specific for Sybil and Eclipse attack: To enhance the resilience against Eclipse and Sybil attacks, we can implement the following measures:

1. Increased connectivity: Instead of relying on a fixed set of nodes, we can enlarge the list to include a larger number of nodes. By increasing the connectivity of the network, the chances of being isolated by a malicious entity in an Eclipse attack are reduced. Additionally, establishing connections with a diverse range of nodes enhances network robustness and decreases the impact of potential attacks.

2. Random node removal: To further mitigate the risk of Eclipse attacks, we can introduce a mechanism to randomly remove nodes from the connection list. By periodically and randomly removing nodes, we prevent attackers from predicting or targeting specific nodes for isolation. This approach adds an additional layer of randomness and makes it more challenging for malicious actors to execute successful Eclipse attacks.

By enlarging the node list and incorporating random node removal, we introduce greater diversity and unpredictability in the network connections. These measures make it significantly more difficult for attackers to isolate a specific node or control the connectivity of the network. As a result, the network becomes more resilient against Eclipse attacks and ensures a more secure and robust environment for decentralized operations.

Because our specification requires us to make every effort to propagate the announcement messages, guarding against DDoS attacks becomes extremely important. Otherwise, any node's transmission of a large amount of data will be replicated and disseminated by the announces, resulting in an amplification of data by at least n times, where n represents the number of nodes in the network. Therefore, we need a method to prevent this. We adopt a combination of blacklist and a signature chain to prevent nodes from pushing excessive announcement messages within a short period. When we detect a node sending a large number of messages, the network will temporarily remove the node to ensure network stability. Additionally, implementing proof-of-work during network joining can hinder attackers from rapidly changing identities, thereby ensuring network robustness.

P2P Protocal

ENROLL INIT

This message is sent by the server after you have connected to it. It will contain a 64 bit challenge which is to be used in calculating a response for the subsequent ENROLL REGISTER messages sent by the client.

size (16 bits)	ENROLL INIT (16 bits)
challenge(64 bits)	

In the message, the "size" field indicates the length of the entire message, while the "ENROLL INIT" field represents the message type. The "challenge" field contains a randomly generated number used for verification purposes.

ENROLL REGISTER

size (16 bits)	ENROLL REGISTER
challenge(64 bits)	
nonce(64 bits)	
encryption type (8 bits)	
public key	

In the message, the "size" field indicates the length of the entire message, while the "ENROLL REGISTER" field represents the message type. The "challenge" field contains a number sent by server. Encryption type is the asymmetric encryption process method, i.e. RSA or ESA, and by which we create a public key, filled in the field "public key".

ENROLL SUCCESS

size (16 bits)	ENROLL SUCCESS (16 bits)
size of neighbers (32 bits)	
neighbers information (4 MB)	

If connection is succeed, server will send to the client a ENROLL SUCCESS message with the neighbers information.

size (16 bits)	ENROLL FAILURE (16 bits)
error number (8 bits)	error description

If connection is failured, server will send to the client a ENROLL FAILURE message with the neighbers information.

ANNOUNCE

When we receive announce messages from other modules, this protocol is designed to make every effort to propagate these announcements. The announce protocol utilizes the BJSON format for data exchange. The data format is defined as follows:

```
{
  TYPE: "ANNOUNCE",
  ID: String (hash value of the original message),
  MESSAGE: String (encrypted message),
  KEYLIST: publicKey[] (array of public keys),
  NEIGHBERSSIZE: number,
  TTL: number
}
```

The "announce" type indicates that it's an announcement message. The "ID" field represents the hash value of the original message, which allows for verification and identification of the message's integrity. The actual content of the message is stored in the "MESSAGE" field, encrypted to ensure data security. The "KEYLIST" field contains an array of public keys, which are essential for the decryption process to retrieve the original message content. This field is also used to indicate which nodes have forwarded the data.

To enhance the propagation efficiency, we employ various optimization techniques. First, we prioritize the forwarding of announcement messages to nodes with a higher reliability score, which ensures that critical information reaches trustworthy destinations promptly. Additionally, we implement a caching mechanism that stores previously processed announcements, avoiding redundant processing and transmission of identical messages.

To prevent potential network overload and minimize unnecessary resource consumption, we introduce a Time-to-Live (TTL) parameter. This parameter defines the maximum duration for which the announce message remains valid and relevant for propagation. Once the TTL expires, the message is no longer forwarded to avoid outdated or obsolete data circulating within the network.

Other APIs

While the remaining APIs adhere to the Voidphone Project Specification.

Exception handling

Churn: Increased connectivity: When a node decides to leave the network, it is important to ensure the connectivity of the remaining nodes. To achieve this, a connectivity check is performed. If the connectivity of the departing node is below a certain threshold, indicating insufficient connections, a process is initiated to connect its neighboring nodes to existing nodes in the network. By establishing these additional connections, the network's overall connectivity is maintained and strengthened, reducing the impact of the departing node on the network's structure and functionality. This proactive approach helps to mitigate any potential disruptions caused by node departures and fosters a more resilient and interconnected peer-to-peer network.

Connection breaks: To handle connection breaks in a more robust manner, we can implement a retry mechanism with a limited number of attempts. Here's an improved version:

- Retry Mechanism: When a connection break occurs, the system will attempt to reconnect to the destination node. It will make up to three retry attempts to establish a successful connection.
- Limited Retry Attempts: The retry attempts are limited to three times, ensuring that the system does not get stuck indefinitely in attempting to reconnect.
- Throw Exception: If the connection attempts are unsuccessful after three retries, an exception is thrown. This informs the system or application that the connection could not be established, allowing appropriate error handling or fallback mechanisms to be triggered.

Additionally, each node is responsible for maintaining its own online status. It calculates its Local Clustering Coefficient based on the number of neighbors' neighbors. If the node discovers that its coefficient is too low, it will attempt to randomly connect with other nodes.

By incorporating this retry mechanism, we provide a more resilient approach to handling connection breaks. It allows the system to make multiple attempts to restore the connection while avoiding excessive retries that could impact overall performance or cause unnecessary delays.

corrupted data: The integrity of the data is already verified by the data link layer, if it happens, we will consider it as an attack and we will add it to the block list for 24 hours.

Future Work:

Security

By incorporating middleware into the system, we can further enhance the protection and security measures.

In Another Branch

In the future, our peer-to-peer (P2P) project has the potential to expand into the financial sector.

Workload:

Yueqi Zhang: Documents and research. Boning Li: Framework building.

Effort

Our collaborative project involves thorough discussions where both Yueqi Zhang and Boning Li contribute to refining the concepts. Yueqi Zhang takes the lead in conducting research, querying information, and documenting our findings. On the other hand, Boning Li concentrates on building the framework and coding to bring our project to life.

Over the course of three weeks, we dedicate our time to determining the project's objectives and tasks. This includes regular communication with our tutor to seek clarification and guidance. During this process, we encountered some confusion regarding the scope of our work. For instance, we deliberated whether to implement the Enroll section and how to address security concerns effectively.

To ensure effective collaboration, we schedule weekly meetings, allocating a 5-hour time slot. These meetings serve as an opportunity for us to synchronize our progress, discuss challenges, and make informed decisions collectively. By leveraging our collaborative efforts and leveraging our individual strengths, we aim to deliver a successful project.