ARIZONA STATE UNIVERSITY

CSE 434, SLN 70569 — Computer Networks — Fall 2021

Instructor: Dr. Violet R. Syrotiuk

Socket Programming Project

Group - 28

Abhishek Jindal	**1222309842**
Sai Cherish Potluri	**1223441869**

1.A Message format and Commands implemented

The server program will maintain the state information about the peers in the system, for example, when a user wants to join the system they will be initiated at a particular IP address, port number and a state. (Free, Leader, and InDHT).

The message format will contain the following commands which will be implemented:

- register
- setup-dht
- dht-complete
- query-dht
- deregister
- leave-dht
- dht-rebuilt
- teardown-dht
- teardown-complete

register :

The *register* command will be the first command that will be used in order to register the process to the server. In order to register the server will record the IP address, one or more ports information.

register sam *IPaddress p0p1p2*

The register command here will act as a client-server program, it will interact with the server and it will check whether the username has been registered before, if it is not registered before, it will return SUCCESS otherwise it will register FAILURE.

setup-dht :

From the processes that were registered, a hash table will be created and in order to do that we utilize setup-dth command. Where the contents of the hash table are distributed over three processes and the process at which the setup-dht command is used it will be the leader of the process.

setup-dht (number of processes) (leader of the process)

The server will look for free processes whose number has been specified in the command, including the leader. Once the command has been executed and returns successful, the server will return a list of processes, their IP address and port numbers. This will construct a ring overlay on top of these processes, including identifiers.

dht - complete:

This command indicates whether the leader process has completed all the steps to set up the Distributed Hash Table or not. On execution this command will give either SUCCESS or FAILURE.

dht-complete(leader-name)

query-dht:

Once the Distributed Hash Table is constructed we can use the query-dht command to query the DHT to obtain values.

query-dht (user-name)

Querying can only be done by the processes that are not part of maintaining the table. Once the command has been successfully executed, the server will at random choose a process that is in the ring and return the user-name, IP address and port as the output.

deregister:

This command will remove the state of a process that is free from the state information and it will return SUCCESS as the output if it were able to successfully terminate the process.

deregister (user-name)

leave-dht:

This command will allow the user to exit the DHT; if the user does not exist or is not participating in DHT maintenance, it will return an FAILURE.

leave-dht(user-name)

dht-rebuilt:

After initiating the leave-dht command, in order to rebuild dht we use the dht-rebuilt command. For this command we give user-name and leader-name as input parameters. The command returns a FAILURE if the user-name for dht-rebuilt is not the same as leave-dht.

dht-rebuilt(user-name)(leader-name)

teardown-dht:

This command is used to remove the DHT from the system. The input parameter for this command is user-name. If the user is not the DHT's leader, the command will return FAILURE. The server will then wait for teardown to finish before returning FAILURE to any remaining queries.

teardown-dht(user-name)

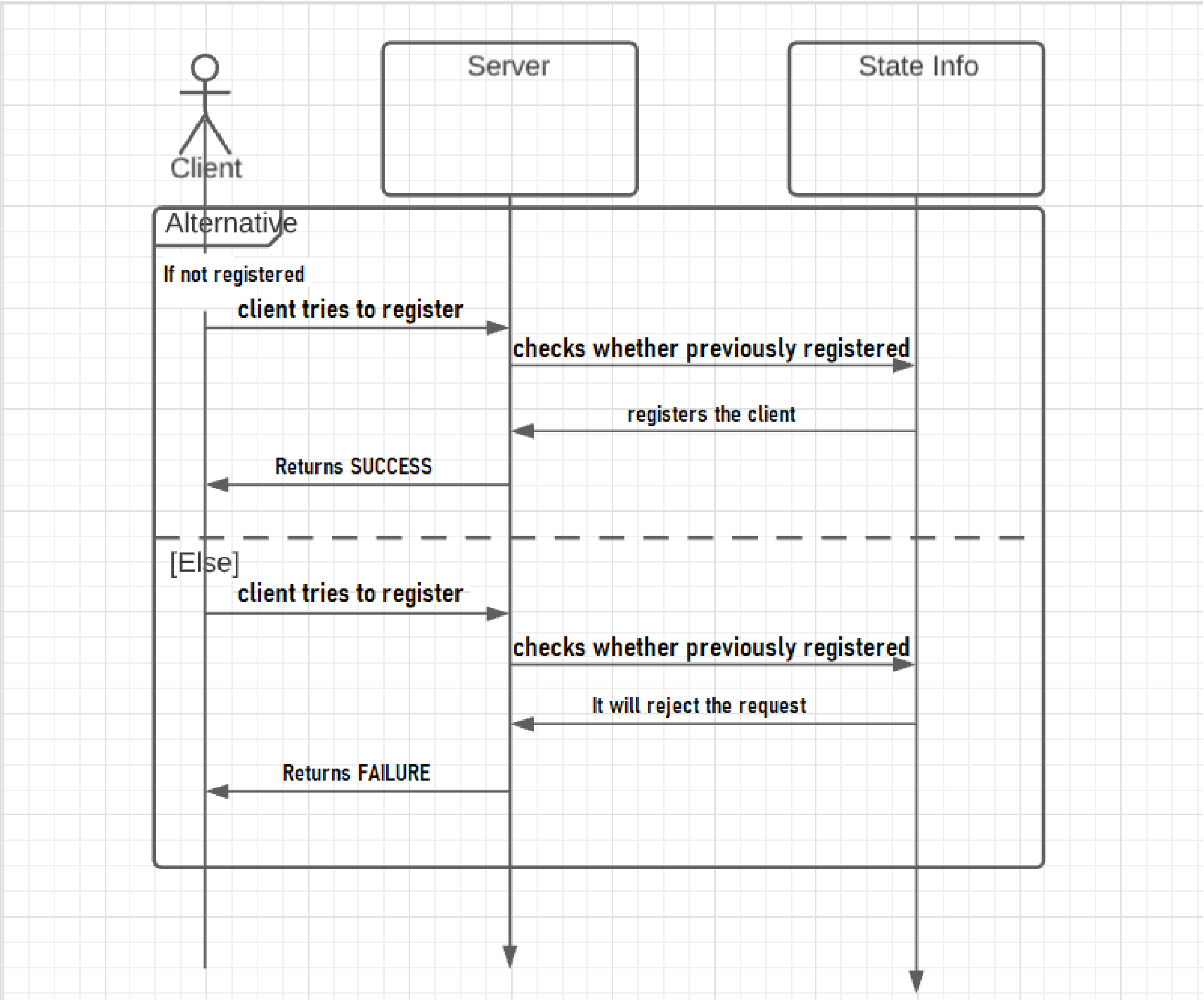
teardown-complete:

This command indicates that the DHT has been deleted. The server returns FAILURE if the user is not the leader of the DHT. Otherwise the server will change the state of each user involved in the DHT to free.

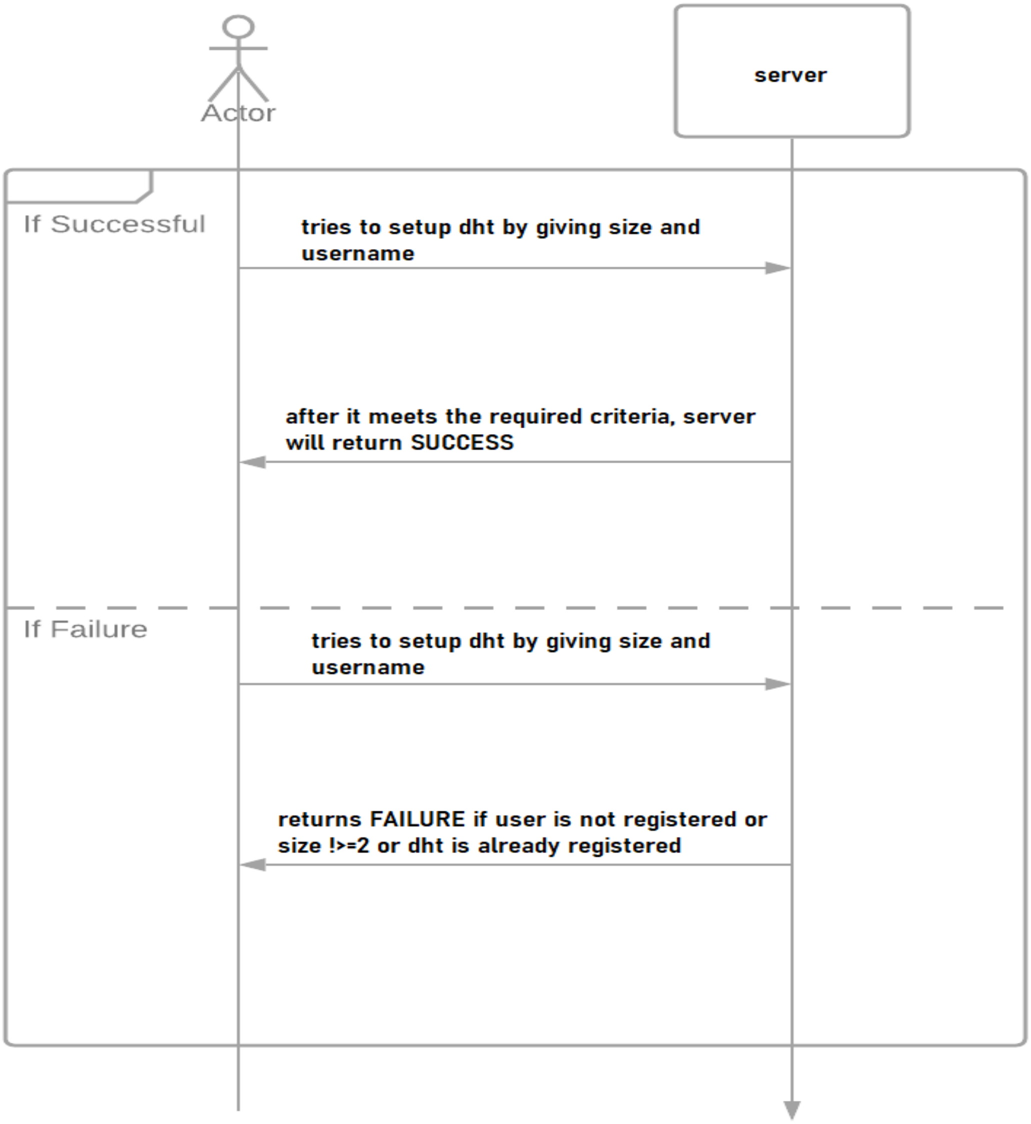
teardown-complete(user-name)

1.B Timespace Diagram

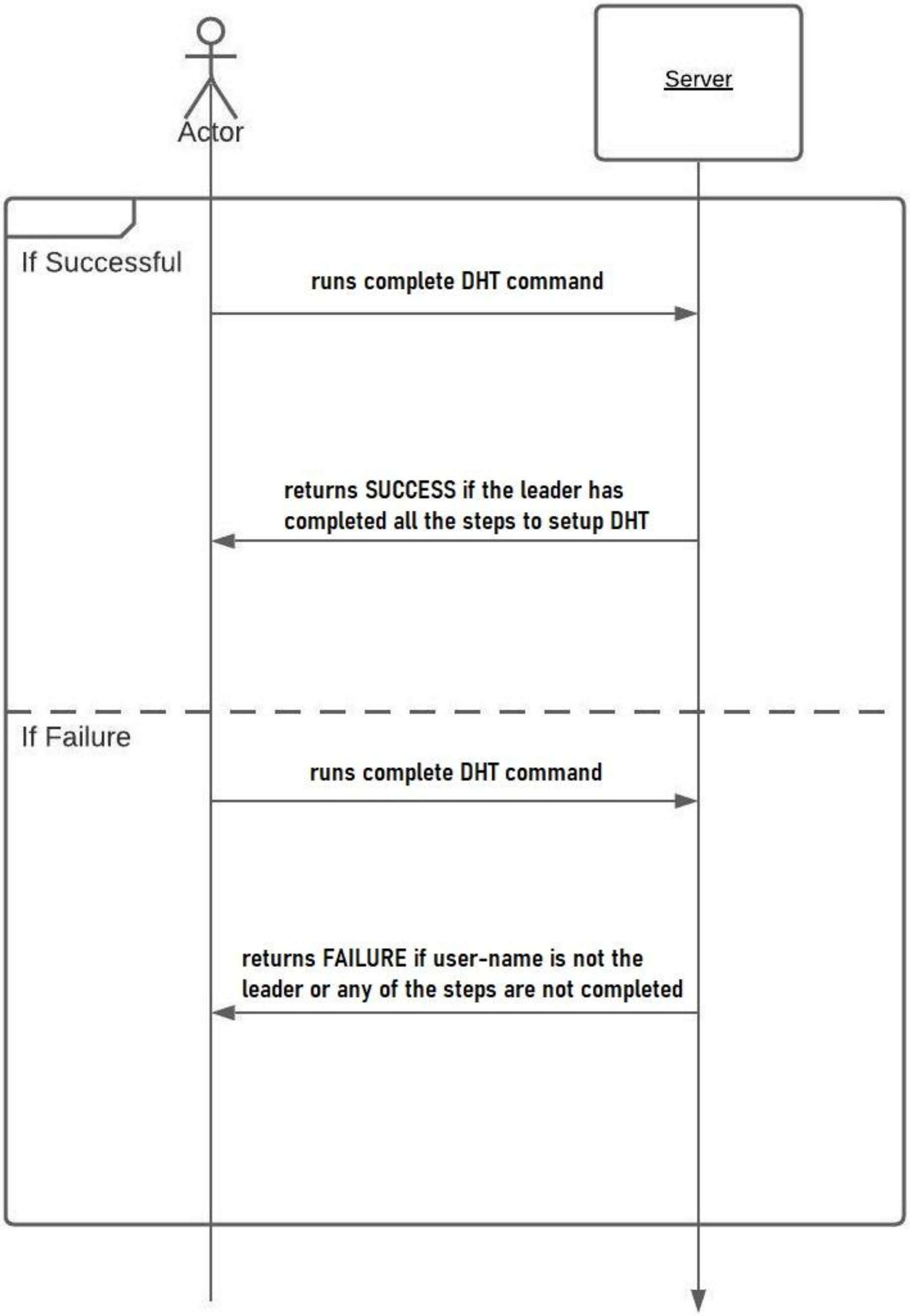
register:



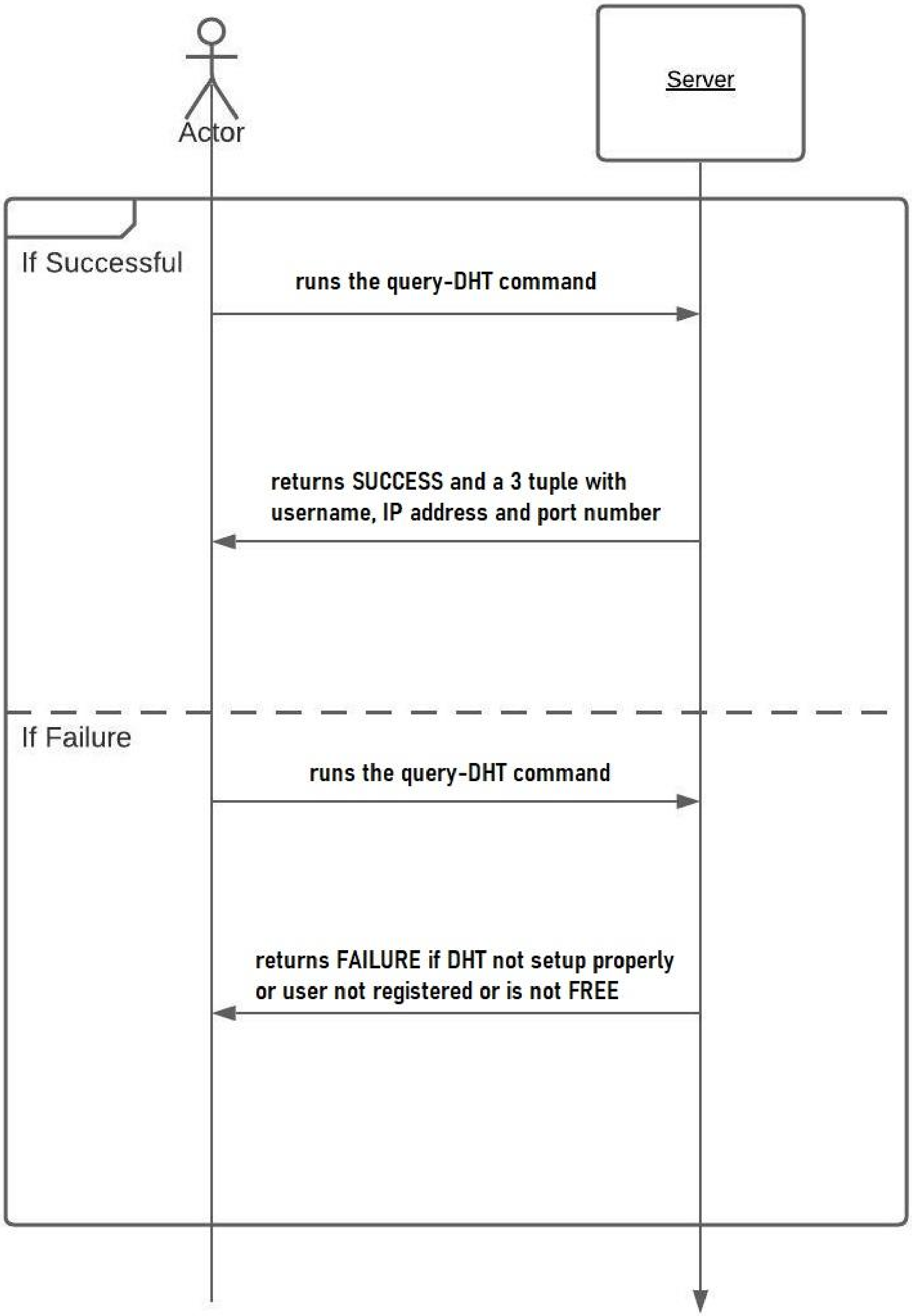
setup-dht:


Actor
server
If Successful
tries to setup dht by giving size and username
after it meets the required criteria, server will return SUCCESS
If Failure
tries to setup dht by giving size and username
returns FAILURE if user is not registered or size !>=2 or dht is already registered

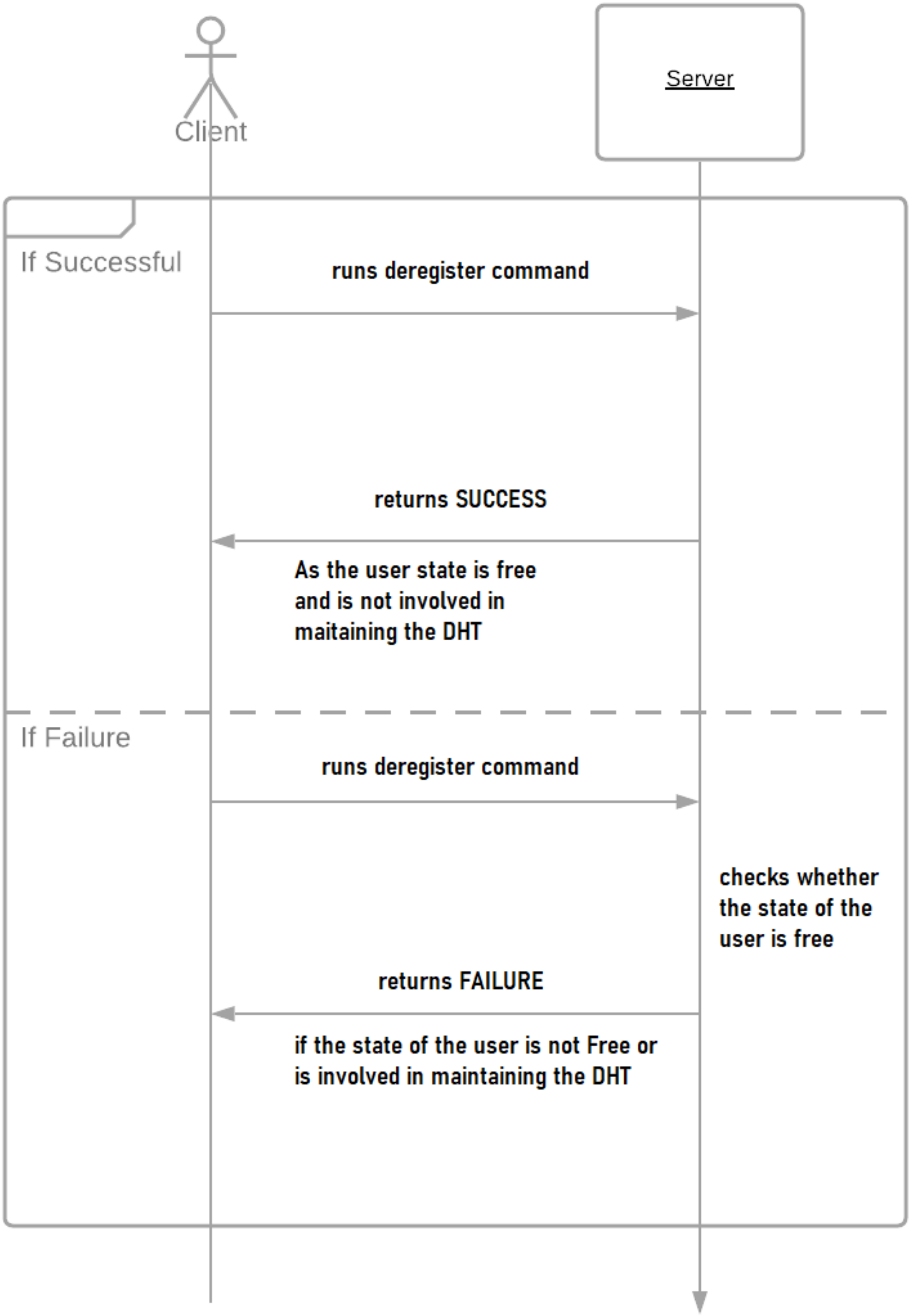
dht-complete:


Actor
Server
If Successful
runs complete DHT command
returns SUCCESS if the leader has completed all the steps to setup DHT
If Failure
runs complete DHT command
returns FAILURE if user-name is not the leader or any of the steps are not completed

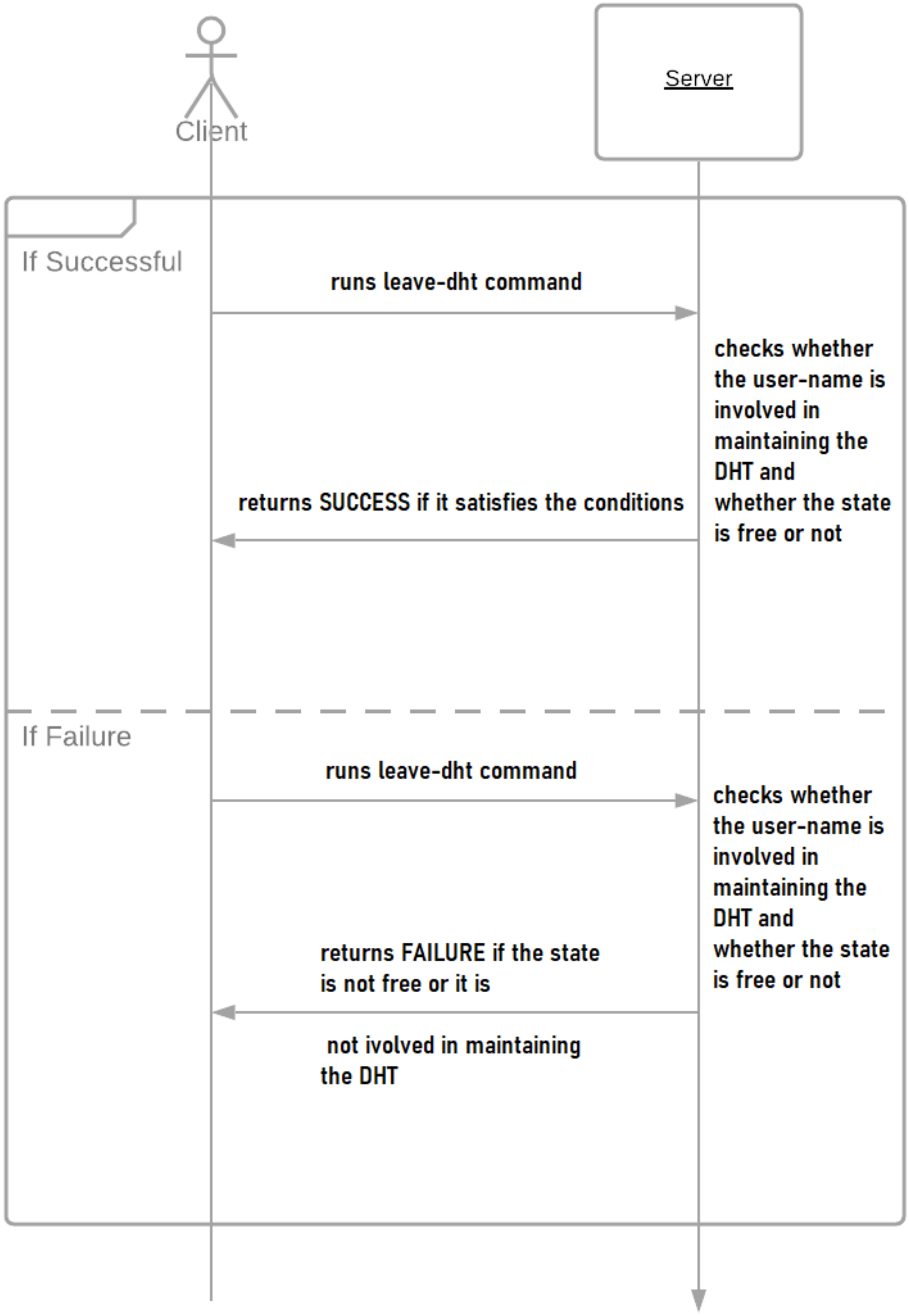
query-dht:


Actor
Server
If Successful
runs the query-DHT command
returns SUCCESS and a 3 tuple with
username, IP address and port number
If Failure
runs the query-DHT command
returns FAILURE if DHT not setup properly
or user not registered or is not FREE

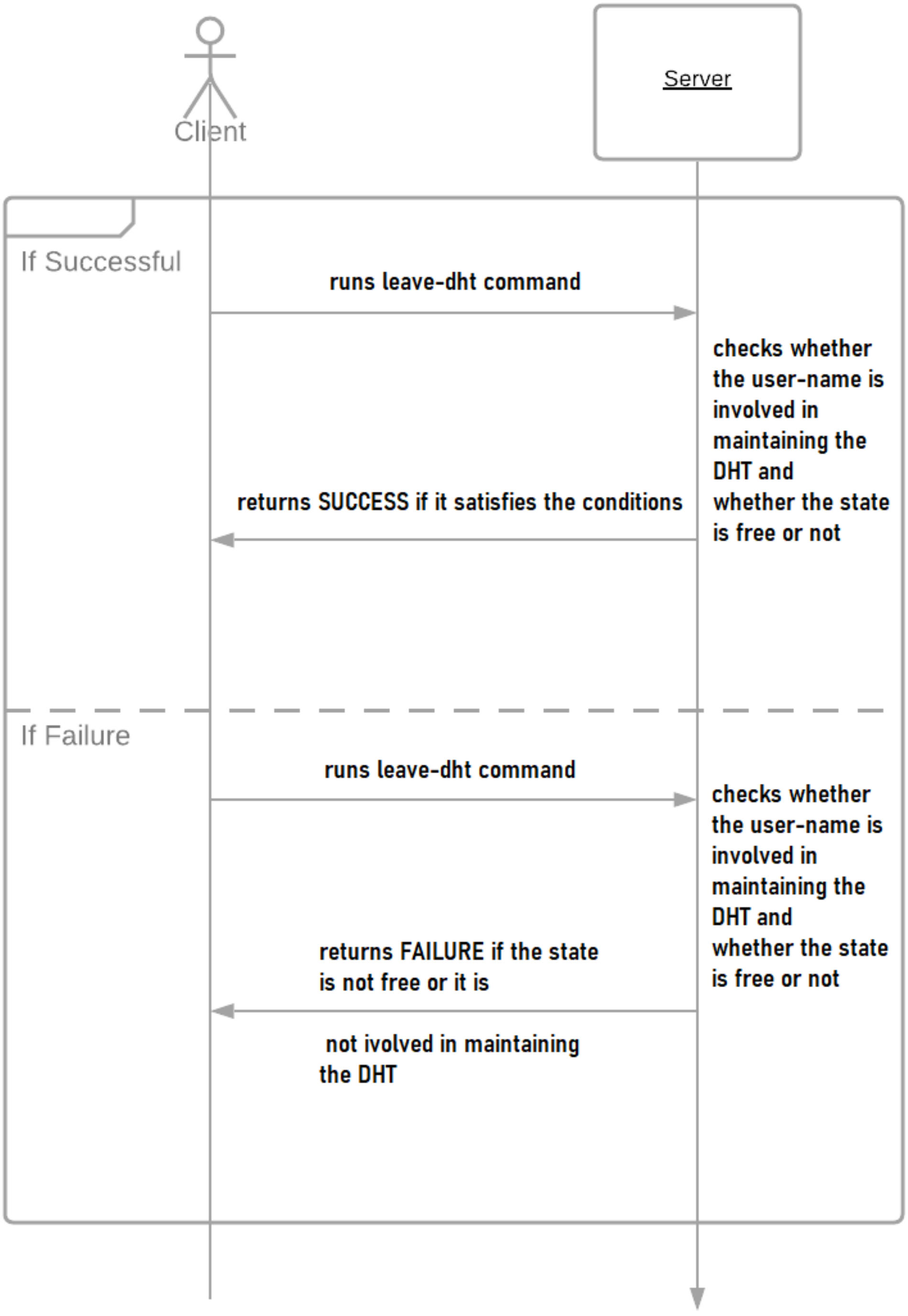
deregister:


Client
Server
If Successful
runs deregister command
returns SUCCESS
As the user state is free
and is not involved in
maitaining the DHT
If Failure
runs deregister command
checks whether
the state of the
user is free
returns FAILURE
if the state of the user is not Free or
is involved in maintaining the DHT

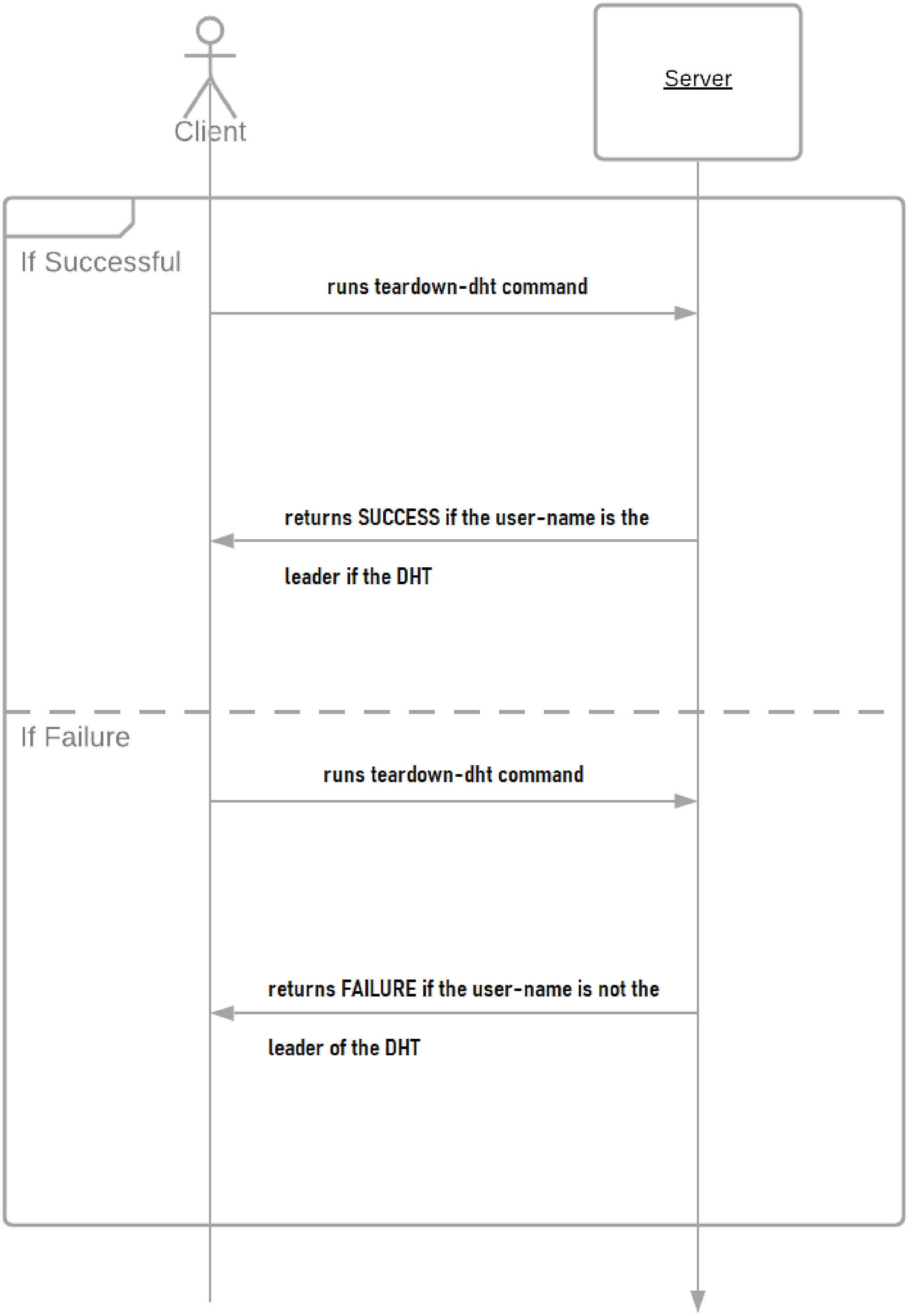
leave-dht:


Client
Server
If Successful
runs leave-dht command
checks whether the user-name is involved in maintaining the DHT and whether the state is free or not
returns SUCCESS if it satisfies the conditions
If Failure
runs leave-dht command
checks whether the user-name is involved in maintaining the DHT and whether the state is free or not
returns FAILURE if the state is not free or it is
not ivolved in maintaining the DHT

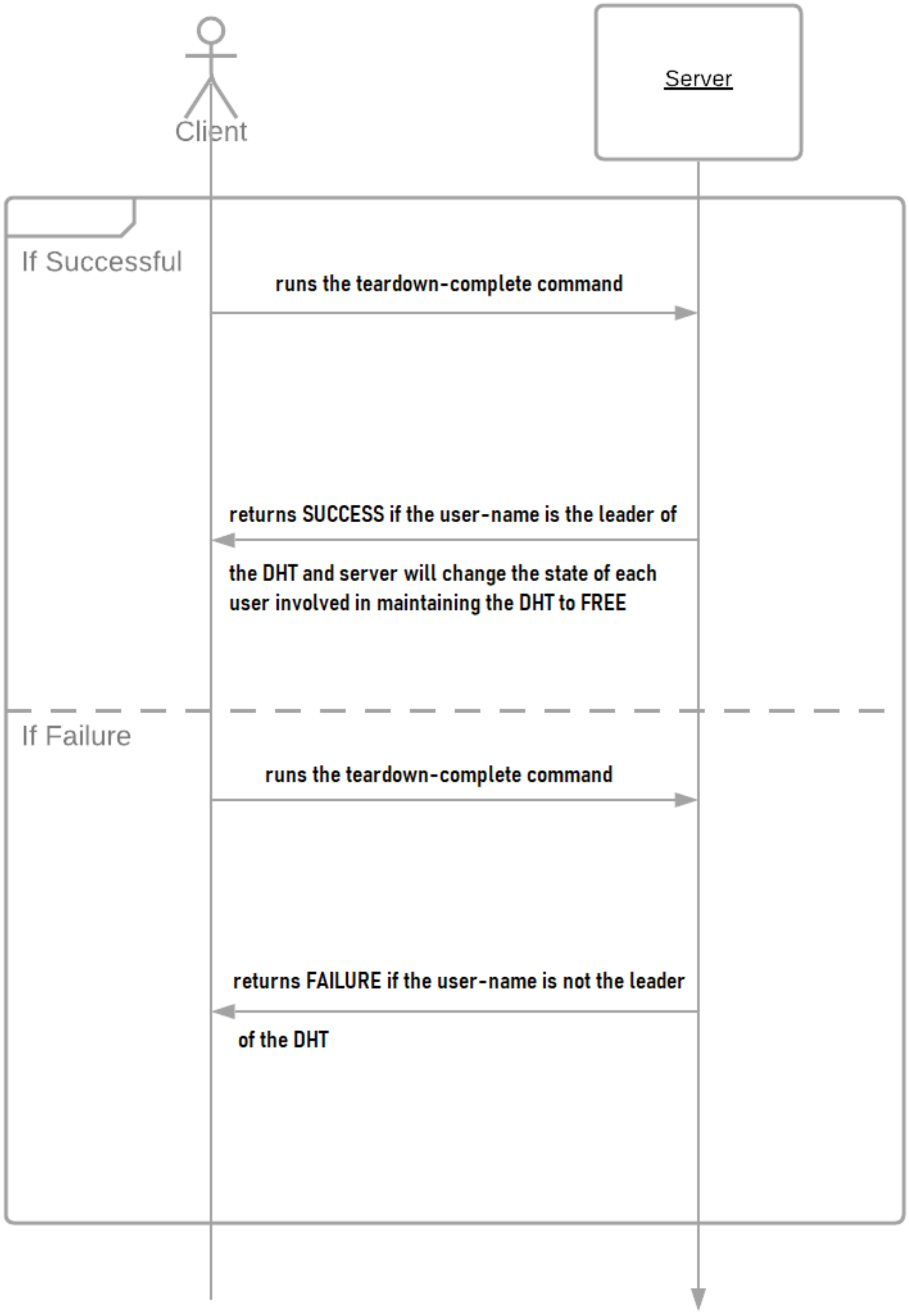
dht-rebuild:


Client
Server
If Successful
runs leave-dht command
checks whether the user-name is involved in maintaining the DHT and whether the state is free or not
returns SUCCESS if it satisfies the conditions
If Failure
runs leave-dht command
checks whether the user-name is involved in maintaining the DHT and whether the state is free or not
returns FAILURE if the state is not free or it is
not ivolved in maintaining the DHT

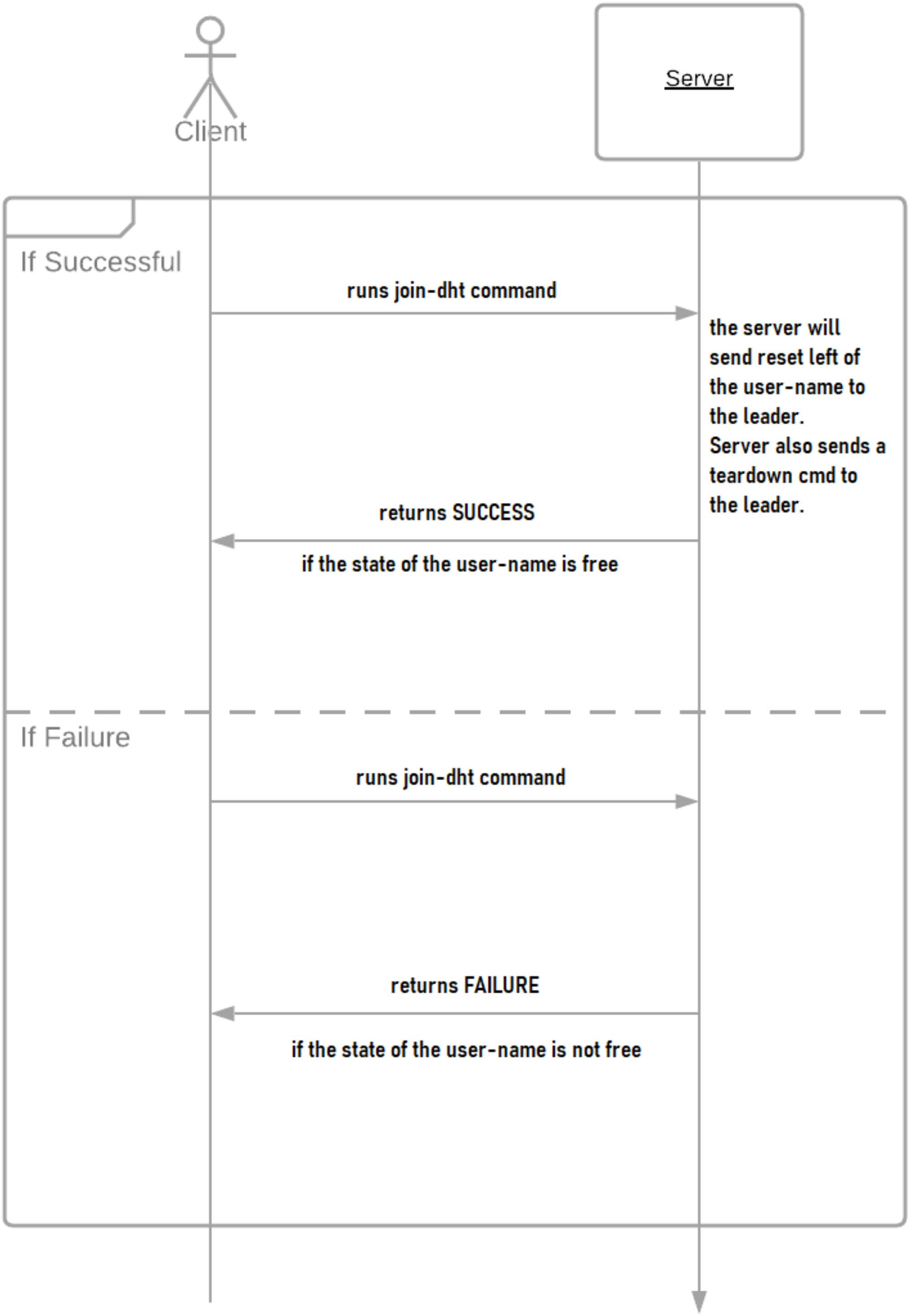
teardown-dht:


Client
Server
If Successful
runs teardown-dht command
returns SUCCESS if the user-name is the
leader if the DHT
If Failure
runs teardown-dht command
returns FAILURE if the user-name is not the
leader of the DHT

teardown-complete:


Client
Server
If Successful
runs the teardown-complete command
returns SUCCESS if the user-name is the leader of
the DHT and server will change the state of each
user involved in maintaining the DHT to FREE
If Failure
runs the teardown-complete command
returns FAILURE if the user-name is not the leader
of the DHT

join-dht:



Client
Server
If Successful
runs join-dht command
the server will send reset left of the user-name to the leader.
Server also sends a teardown cmd to the leader.
returns SUCCESS
if the state of the user-name is free
If Failure
runs join-dht command
returns FAILURE
if the state of the user-name is not free

1.C Data Structures Used and Methods Implemented

For implementing the DHT(Distributed Hash Table) socket project we start by designing a peer-to-peer application program where the server program will maintain the state information about the peers in the system. We have specified a peer node data structure that will house the left, right, query ports, the IP address, and also the state(Free, Leader, or InDHT). We have made the peer node to be serializable, in order to convert it into a byte array for communication. We have also used hashmap as an distributed hash table, where the hashmap is of the data type:

<integer, list of records>

The integer is for the position of the record as per the hash function. Since, there could be collisions in this hash function, we are storing a list of records for every position.

The port numbers that we have used are 28XY and the range is between 2801 to 2833. Where 28 indicates our group number and X is the peer number and Y will describe whether it is the left, right or query. We have used a separate thread for left, right and

After getting the peer nodes, the leader sends set-id commands to every other peer node(in the overlay ring).The format of the set id command is set-id<whitespace>identifier<whitespace>ring size and its corresponding left and right peers.

While querying the hashtable, the queries can come only from the processes that are not actually part of maintaining the table, once the query is made it can start at any process which is chosen by the server at random.

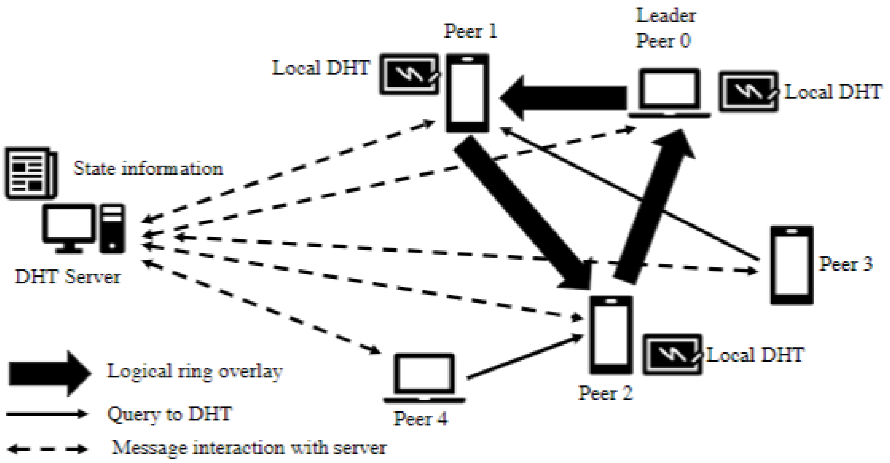
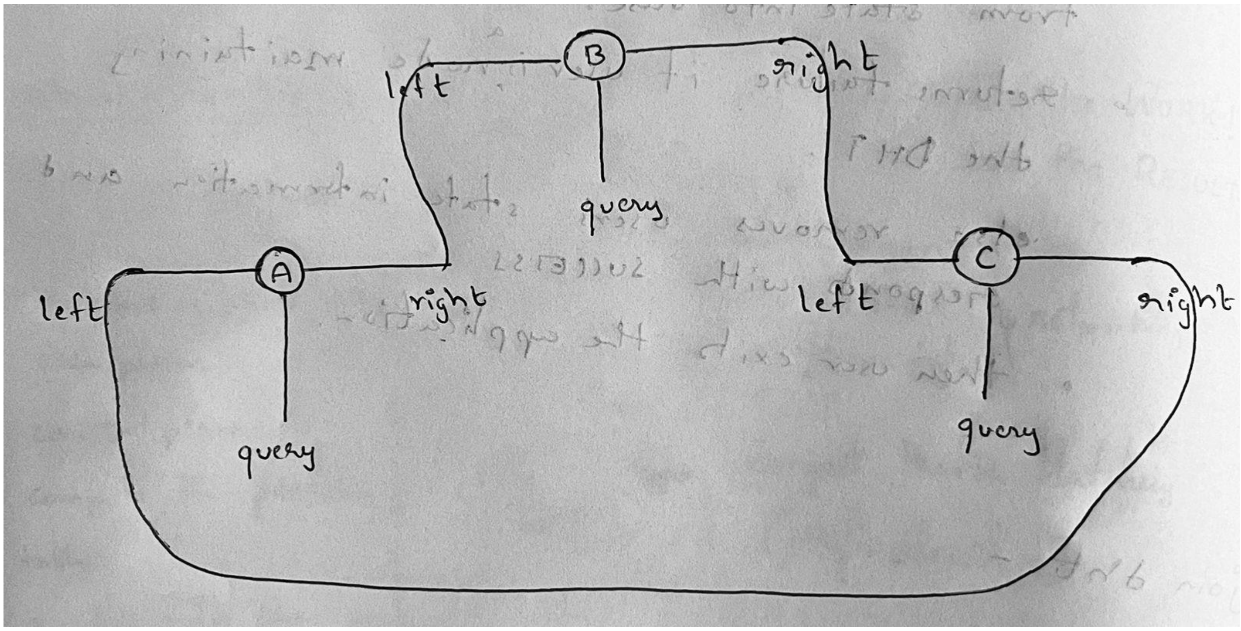


Figure 1: Architecture of the DHT application.

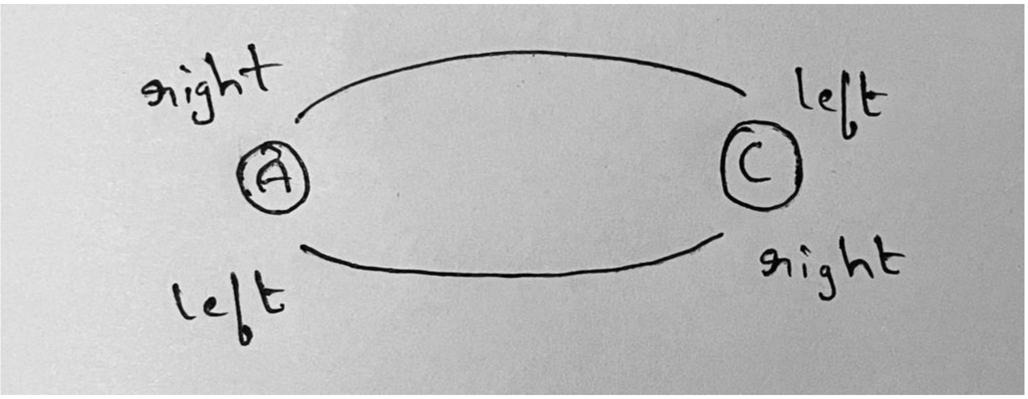
The query will start at the process that will be chosen by the server but will propagate throughout the ring till the requirements are met. If the one process has not met the requirements it will then pass the request on to the next port and instead of the next process sending its data to the query process it will pass it to the upstream process which will then pass the response on to the end hosts.

In this approach we are maintaining a reset-id command counter, which will keep count of the reset-id command execution .When a leave-dht command is issued the DHT is required to be rebuilt with n-1 nodes.The leave-dht command will remove the given node from the overlay ring. In the leave-dht, the node that we want to leave will firstly initiate the teardown command, which will delete its own local DHT. Now the node that wants to leave will initiate a reset-id command that will reset the node's right neighbours identifier to zero, the reset-id command will run throughout the ring with each node sending to its right node, this process will continue until it reaches the node which is leaving the overlay ring. Once it reaches the leaving node, it will reset-left and reset-right commands. Please find below the example for more clarity.

Let us assume the leaving node is B.



The node B must now reset one neighbour of each of its own left and right neighbours to remove itself from the logical ring. Node B will now reset the left neighbour(A) of B's right neighbour(C) to its left neighbour. Also, the right neighbour(C) of B's left neighbour to its right neighbour. Now the leaving node B will send the rebuild-dht command to its right neighbour, which is the new leader. On completion of rebuilding DHT it sends a message dht-rebuilt to the server.

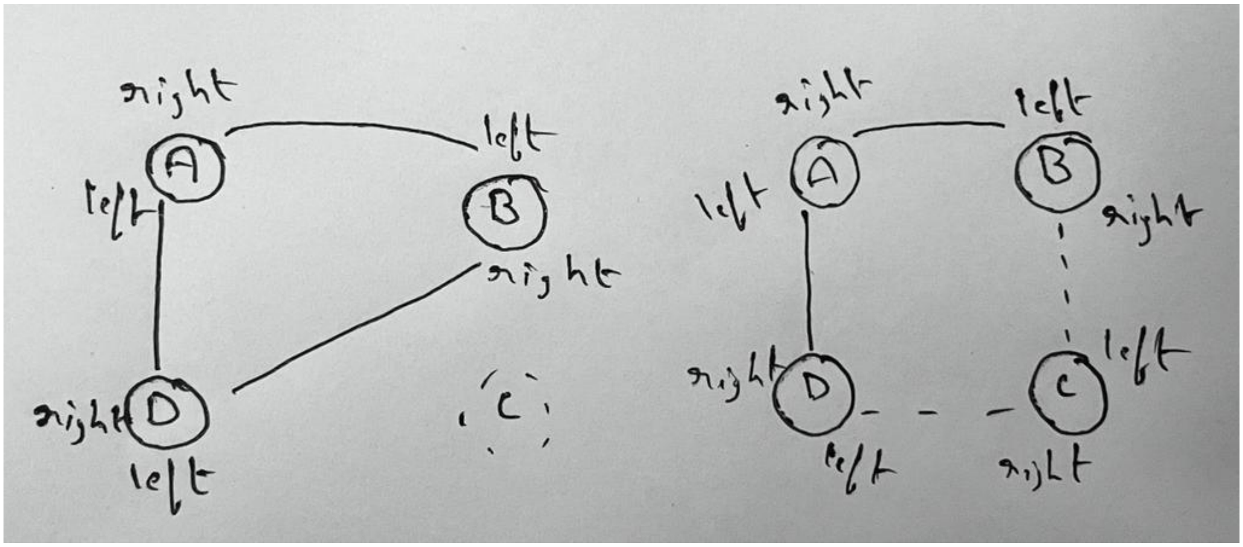


DHT state after B left

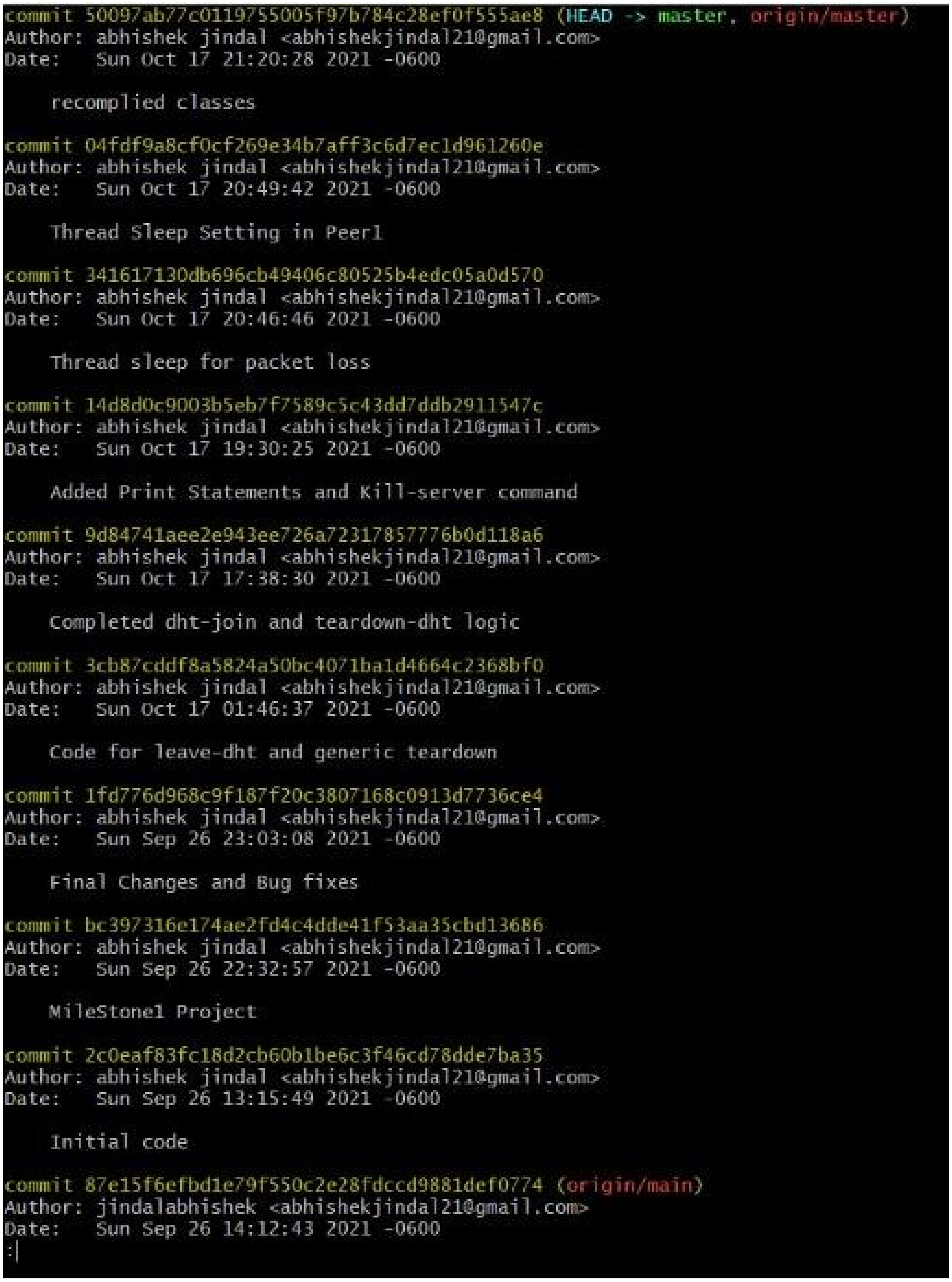
Now if we want to add another node to the ring, we utilize the **join-dht** command. Let us take an example of a ring with nodes A, B, and D, where B is the leader. And if we now want to introduce a new Node C to the ring, we will use the join-dht command. We have come up with the following design for this command.

1. After receiving the join-dht <user-name> command, the server sends a reset-left command to the joining node such that the leader node becomes the left peer of the joining node.
2. Server would also send a teardown command to the leader node so that redistribution of the data can take place.
3. As part of teardown, the leader sends a reset-left command to its existing right peer node so that the new node becomes the left peer of it. Further leader will send a reset-right command to this new node so that the right peer of the new node becomes the right peer of the leader node.
4. In the end, the leader node sets its right peer to the joining (new) node.
5. Please note that this command executes a bit differently than leave-dht command. In the leave-dht command, we were first resetting the neighbours of the leaving node and then resetting the ids of all the other nodes. But in join-dht command, we first reset the neighbours such that the new node gets added to the ring and then the leader node initiates the reset-id command that will run throughout the ring until it comes back to the leader again.
6. Leader now initiates the rebuild-dht command to re-distribute the data in this new overlay ring.
7. After the DHT gets rebuilt, it will send the dht-rebuilt command to the server.

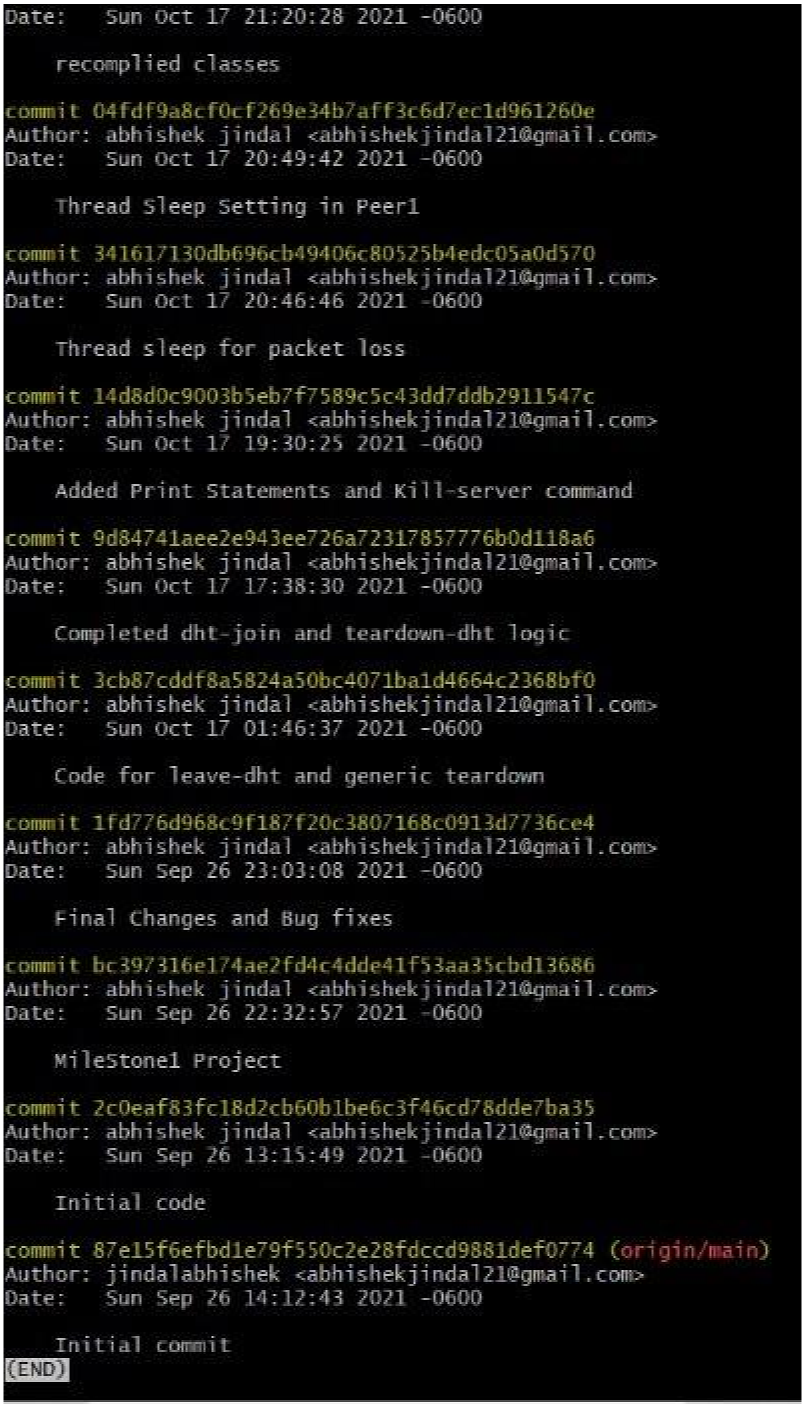
Please note that join-dht also uses the same teardown and reset-id commands which leave-dht was using. The way we differentiate is by parsing the queries and taking action accordingly. This way we have maintained the generosity of these commands in all the tasks.


right
A
left
left
B
right
right
D
left
C
right
left
A
B
left
right
left
right
D
C
left
right

1.D Snapshots of Commits in Git:



```
commit 50097ab77c0119755005f97b784c28ef0f555ae8 (HEAD -> master, origin/master)
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Oct 17 21:20:28 2021 -0600

    recomplied classes

commit 04fdf9a8cf0cf269e34b7aff3c6d7ec1d961260e
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Oct 17 20:49:42 2021 -0600

    Thread Sleep Setting in Peer1

commit 341617130db696cb49406c80525b4edc05a0d570
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Oct 17 20:46:46 2021 -0600

    Thread sleep for packet loss

commit 14d8d0c9003b5eb7f7589c5c43dd7ddb2911547c
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Oct 17 19:30:25 2021 -0600

    Added Print Statements and Kill-server command

commit 9d84741aee2e943ee726a72317857776b0d118a6
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Oct 17 17:38:30 2021 -0600

    Completed dht-join and teardown-dht logic

commit 3cb87cddf8a5824a50bc4071ba1d4664c2368bf0
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Oct 17 01:46:37 2021 -0600

    Code for leave-dht and generic teardown

commit 1fd776d968c9f187f20c3807168c0913d7736ce4
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Sep 26 23:03:08 2021 -0600

    Final Changes and Bug fixes

commit bc397316e174ae2fd4c4dde41f53aa35cbd13686
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Sep 26 22:32:57 2021 -0600

    MileStone1 Project

commit 2c0eaf83fc18d2cb60b1be6c3f46cd78dde7ba35
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Sep 26 13:15:49 2021 -0600

    Initial code

commit 87e15f6efbd1e79f550c2e28fdccd9881def0774 (origin/main)
Author: jindalabhishek <abhishekjindal21@gmail.com>
Date:   Sun Sep 26 14:12:43 2021 -0600
:
```



```
Date:   Sun Oct 17 21:20:28 2021 -0600

    recomplied classes

commit 04fdf9a8cf0cf269e34b7aff3c6d7ec1d961260e
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Oct 17 20:49:42 2021 -0600

    Thread Sleep Setting in Peer1

commit 341617130db696cb49406c80525b4edc05a0d570
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Oct 17 20:46:46 2021 -0600

    Thread sleep for packet loss

commit 14d8d0c9003b5eb7f7589c5c43dd7ddb2911547c
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Oct 17 19:30:25 2021 -0600

    Added Print Statements and Kill-server command

commit 9d84741aee2e943ee726a72317857776b0d118a6
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Oct 17 17:38:30 2021 -0600

    Completed dht-join and teardown-dht logic

commit 3cb87cddf8a5824a50bc4071ba1d4664c2368bf0
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Oct 17 01:46:37 2021 -0600

    Code for leave-dht and generic teardown

commit 1fd776d968c9f187f20c3807168c0913d7736ce4
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Sep 26 23:03:08 2021 -0600

    Final Changes and Bug fixes

commit bc397316e174ae2fd4c4dde41f53aa35cbd13686
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Sep 26 22:32:57 2021 -0600

    MileStone1 Project

commit 2c0eaf83fc18d2cb60b1be6c3f46cd78dde7ba35
Author: abhishek jindal <abhishekjindal21@gmail.com>
Date:   Sun Sep 26 13:15:49 2021 -0600

    Initial code

commit 87e15f6efbd1e79f550c2e28fdccd9881def0774 (origin/main)
Author: jindalabhishek <abhishekjindal21@gmail.com>
Date:   Sun Sep 26 14:12:43 2021 -0600

    Initial commit
(END)
```

1.E Video Link

DHT Socket Project Final Demo- Group 28