In this project, you will develop a console-based client-server application that implements a simplified DHCP protocol using socket programming in Python language.

Implement the DHCP protocol

In this project, the server program listens to the requests from clients. Use `UDP` for the transport layer. The clients initially have no IP address. A new client follows the 4 steps of `DHCP` protocol to discover the DHCP server and get an IP address from the server. We assume, the range of IP addresses available to the server is `192.168.45.0/28`, a block of 16 IP addresses including `Net ID` and `Broadcast IP` address. Note that you cannot assign the Net ID or Broadcast IP addresses to any node in the network. Thus, there are 14 IP addresses available in this range: `192.168.45.1` to `192.168.45.14`. The server maintains a list of IP addresses, whether they have been assigned, to which client each IP address has been assigned, and when does the assigned IP address expire. We assume the lease time in our application is `60 second` since the IP address is assigned (when the server sends an `ACKNOWLEDGE` message). We identify a client by its `MAC` address. On the server, you should store the list of clients' MAC addresses and, associated IP address of each client, and a timestamp indicating the time the IP address will be expired.

Recall that the DHCP protocol contains 4 steps: `DISCOVER`, `OFFER`, `REQUEST`, `ACKNOWLEDGE`. First, the client sends a `DISCOVER` message. Then the server replies an `OFFER` message. Then the client sends a `REQUEST` message. Then the server assigns an IP address and replies an `ACKNOWLEDGE` message. Note that we identify a client by its MAC address. You should extract the MAC address of the client machine and send it to the server in the messages.

The server should also maintain a record for every client containing the following elements:

- Record number. (e.g. `1`)
- MAC address. (e.g. `A1:30:9B:D3:CE:18`)
- IP address. (e.g. `192.168.45.1`)
- Timestamp: This is a `datetime` object in `ISO` format. (e.g. `2022-02-02T11:42:08.761340`)
- Acked: This is a Boolean indicating whether the assigned IP address has been acknowledged by the server. (e.g. `False`)

All communication between the client and server must follow this message format (All uppercase and separated by spaces)

Clients can send the following requests to the server:

- `LIST`
- `DISCOVER A1:30:9B:D3:CE:18`
- `REQUEST A1:30:9B:D3:CE:18 192.168.45.1 2022-02-02T11:42:08.761340`
- `RELEASE A1:30:9B:D3:CE:18 192.168.45.1 2022-02-02T11:42:08.761340`
- `RENEW A1:30:9B:D3:CE:18 192.168.45.1 2022-02-02T11:42:08.761340`

Server can send the following responses to the clients:

- `DECLINE`
- `OFFER A1:30:9B:D3:CE:18 192.168.45.1 2022-02-02T11:42:08.761340`
- `ACKNOWLEDGE A1:30:9B:D3:CE:18 192.168.45.1 2022-02-02T11:42:08.761340`
- `LIST ALL RECORDS AS STRING` (Exception to the message format)

> Replace MAC, IP and TIMESTAMP values accordingly.

The steps are as following:

1. First, we run the server program. The server always listens to the requests come from the clients.
2. Now we run the client program. The client sends a `DISCOVER` message to the server, which includes the client's MAC address. Note: You should extract the MAC address of the client's system and enter that into the message. You should work with the hexadecimal notation, which is the notation for MAC addresses, e.g. `A1:30:9B:D3:CE:18`. The client starter code provides the function to retrieve the mac address.
3. When the server program receives a `DISCOVER` message:

- The server will check its records whether any IP address has already been assigned to the client, i.e., the server checks the list whether the MAC address in the message exists in the list or not.
- If a record was found, check its timestamp, which indicates the expiration time. If the timestamp has not expired yet, set the `Acked` element to True for this client. Then the server will directly send an `ACKNOWLEDGE` message to the client containing the client's MAC address, the IP address that has already been assigned to this client, and the timestamp, which indicates the expiration time.
- If the timestamp has expired, use the same IP address for the client. Update the timestamp in the server's record for this MAC address to `60 seconds from now`. Update the `Acked` element to `False` for this record. Then, the server will send an `OFFER` message to the client. This message contains the client's MAC address, the same IP address as the one that has been assigned earlier to this client, and a new timestamp indicating the expiration time for this IP address assignment. Note: If the client does not accept the offer by the timestamp, the assigned IP address will expire again but the server will not remove the record from its records. Note: In DHCP protocol, the lease time is a few hours long. But in this project, we will use a short lease time of 60 seconds.
- If no record was found for this client, check whether the given pool of IP addresses has not been fully occupied by the current clients. If the server reaches the last IP address in the pool, it will check the timestamps in the records of assigned IP addresses. If the timestamp of a record has already expired, the server will use that IP address for the new request. The server will update that record with the new client's MAC address and a new timestamp, indicating the expiration time of this IP address assignment, which will be `60 seconds from now`. Set the `Acked` element to `False` for this record.
- If the whole pool of IP addresses is already occupied by the clients and there is no expired IP address that the server can assign to the new client, the server declines the request and replies a `DECLINE` message to the client.
- Otherwise, the server will take the next available IP address and store a new record for this client at the server's records containing the client's MAC address, the assigned IP address, and a timestamp, which is `60 seconds from now`. Set the `Acked` element to `False` for this record. Then it will send an `OFFER` message to the client containing these three pieces of information. Note: If the client does not accept the offer by the timestamp, the IP address will expire but the server will not remove the record from its records.

4. When the client receives an `OFFER`, it will check whether the MAC address in the message matches the client's MAC address. If yes, it checks whether the timestamp has expired. If not, it will send a `REQUEST` message containing the client's MAC address and the IP address and timestamp offered by the server. If the client receives a `DECLINE` message from the server, it

will display a message on the screen for the user indicating that the request was declined, and the client program terminates. The user may run the client program later and try again.

5. When the server receives a `REQUEST`, it will first check its records whether an IP address has been assigned to the client and the IP address in the `REQUEST` message matches with the IP address in the record. If not, it will send a `DECLINE` message to the client. Otherwise, the server will check the timestamp in its record has expired. If yes, it will send a `DECLINE` message to the client. Otherwise, the server will set the `Acked` element to `True` for this record and send an `ACKNOWLEDGE` message containing the client's MAC address, and the assigned IP address and timestamp.
6. When the client receives an `ACKNOWLEDGE` message, it checks whether the MAC address in the message matches its own MAC address. If not, it will display a proper message and terminate the program; Otherwise, display a message to the user indicating that the IP address `x.x.x.x` was assigned to this client, which will be expired at time TTT. Then the client will display the menu given in the next step for the user. Note that we do NOT actually setup the client's machine with the given IP address. You are developing a simple application, which is not expected to change your instance's setup.
7. At this point, display a menu for the client containing three options.

- **release**: If the client chose to release, send a `RELEASE` message to the server, containing its MAC address, IP address and timestamp.
- **renew**: If the client chose to renew, send a `RENEW` message to the server, containing its MAC address, IP address and timestamp.
- **quit**: If the client chose to quit, terminate the client's program. Note that the client does NOT release its IP address when it quits.

8. If the server receives a `RELEASE` message, it will check its records. If it finds the client's MAC address in the record, it will release the IP address assigned to the client. In the server's records, the server will set the timestamp to the current time, indicating that the IP assignment has expired. It will also set the `Acked` element to `False`. If the server did not find the client's MAC address in the records, it does nothing. Note that the server does not reply any message to the client for a `RELEASE` request. Once the client program sent the `RELAESE` message, it will display the above menu for the user again.
9. If the server receives a `RENEW` message, it will check its records. If it finds the client's MAC address in the record, it will renew the IP address assigned to the client. In the server's records, the server will set the timestamp to `60 seconds from now`. It will also set the `Acked` element to `True`. Then, the server will send an `ACKNOWLEDGE` message to the client, containing the client's MAC address, and the assigned IP address and the new timestamp. If the server did not find the client's MAC address in the records, it checks the pool whether any IP address is available and will take it. If not, it checks the timestamps of the records whether any of them has expired and will take it. Then it removes that record from the records. If the server failed to get a new IP address, it will send a `DECLINE` message to the client. If the server succeeded to get a new IP address, it will store a new record in its records containing the client's MAC address, the new IP address, and a new timestamp, which is `60 seconds from now`. It will also set the `Acked` element to `False`. Then, the server will send an `OFFER` message to the client containing the client's MAC address, the new IP address, and timestamp.
10. Make sure the server never assigns more than one IP address to a MAC address at the same time or assigns the same IP address to more than one MAC Address. Make sure when a client sends a `RELEASE` and immediately `RENEW`, it receives the same IP address as it used to have.
11. Make sure that the clients and the server display all feedback messages on the screen to demonstrate each step of the process. Start the messages with `server:` or `client:`, e.g. `server: I see that the client with MAC address XXX is discovering`. `server: I assigned IP address x.x.x.x to client with MAC address XXX`.

12. We assume there is also another type of client named `admin`. When we run the admin client, it will send a `LIST` message to the server. When the server receives the `LIST` message, it will send the contents of its records to the admin client. The admin client will receive that and display its content as a list of records.
13. **Extra credit**: We assume there is also an `attacker` client. This client run a DoS attack. In a DoS attack, the attacker spoofs many messages and sends them to the server, which will exhaust the server resources. Thus, the server starts to deny the requests from eligible clients. When you run the attacker client program, it will send several `DISCOVER` messages to the server to fill the entire pool of available IP addresses at the server. In our example, the pool contains 14 IP addresses, so 14 messages will be sufficient to run the attack. For each message, the attacker creates a random MAC address and enters that to the message. Thus, the server assumes these messages arrive from different clients and assign all IP addresses in its pool to these fraudulent MAC addresses. Now if a legitimate user sends a `DISCOVER` message, the server will respond with a `DECLINE` message.

Notes to test the program

Run the following steps and take a screenshot of your terminal at each step. Zoom in the screenshot and make it readable. Create a `report.pdf` file and include the screenshots of each step in the report.

Test your program as following:

1. Run the server on `server`.
2. Run the client on `client1`. Make sure that the client receives the first available IP address. The menu will be displayed for the client. Choose **quit**.
3. Now we want to make sure that the same client cannot receive a new IP address from another terminal. Run the client on `client1` again. Make sure that the server verifies that the client already has an IP address `x.x.x.x`. The menu will be displayed for the client.
4. Run the client on `client2`. Now make sure that `client2` receives the next available IP address. The menu will be displayed for the client. Wait for 10 seconds and then choose **renew**. Make sure the server immediately sends an `ACKNOWLEDGE` and renews the timestamp. Then, the menu will be displayed again. Wait for 1 minute and then choose **renew**. Make sure that you go over `DISCOVER` - `OFFER` - `REQUEST` - `ACKNOWLEDGE`, get the same IP address and a new lease time. Then, the menu will be displayed again. Now, choose **quit**.
5. Run the admin client on `admin` to display the list of current clients. Make sure that the list contains the 2 current clients.
6. On `client1`, choose **release**. `client1` will display a proper message and the menu. Make sure that the server has released the client's IP address. The server should display a message on the screen indicating that.
7. Run the admin client on `admin` to display the list of current clients. Make sure that the list contains only `client2`.
8. On `client1`, again choose **release**. `Client1` will display a proper message and the menu. Make sure that the server verifies that the IP address has already been released. The server should display a message on the screen indicating that.
9. Run the admin client on `admin` to display the list of current clients. Make sure that the list contains only the client on `client2`.
10. On `client1`, choose **renew**. Make sure that the server assigns the same IP address as before to `client1` and replies to the client. The server should display a message on the screen

indicating that. Then `client1` displays a proper message and the menu. Choose **quit**.

11. Run the admin client on `admin` to display the list of current clients. Make sure that the list contains the 2 current clients.
12. Release IP addresses of both `client1` and `client2`. Then, run the admin client on `admin`. Make sure that the list replied by the server indicates that both clients have released their IP addresses.
13. **Extra credit test**: Run the attacker client on `attacker`. Then on `client1`, run the client. Make sure that the server declines the `DISCOVER` message from `client1`. Then, run the admin client. Make sure that 14 IP addresses have been offered to the fraudulent MAC addresses that have been created by the attacker.

Submission

Submit to Cougar Courses with the following files only (no `.zip` files):

1. server.py
2. client.py
3. admin.py
4. attacker.py (extra credit)
5. report.pdf

In the report, you need to provide your team member names, your program design and implementation details along with your testing. It is important to follow the testing instruction given above and include at least 13 screenshots, one for each step.