Statistics

I implemented the solution in Python. There are two files: *client.py* and *server.py* in which the Client and Server classes can be found.

The code used to test the client and server on the same machine is located in *main.py.* The user can choose the protocol (TCP or UDP) and the mechanism (streaming - without ACK or stop-and-wait - with ACK).

I performed tests only on localhost. The results obtained when sending 1GB of data (in this case, I used 5000 photos) are printed after each execution. They can be observed in the following tables:

BUFFER\_SIZE = 512

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | TCP | TCP ACK | UDP | UDP ACK |
| MB sent | 1043.64 | 1043.64 | 1043.64 | 1043.64 |
| MB received | 1043.64 | 1043.64 | 996.55 | 1043.64 |
| Msg sent | 2139851 | 2139851 | 2139851 | 2139851 |
| Msg received | 2139872 | 2139851 | 2043304 | 2139851 |
| Time (seconds) | 35.41 | 103.33 | 16.32 | 88.57 |

BUFFER\_SIZE = 1024

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | TCP | TCP ACK | UDP | UDP ACK |
| MB sent | 1043.64 | 1043.64 | 1043.64 | 1043.64 |
| MB received | 1043.64 | 1043.64 | 1021.33 | 1043.64 |
| Msg sent | 1071163 | 1071163 | 1071163 | 1071163 |
| Msg received | 1071451 | 1071163 | 1048271 | 1071163 |
| Time (seconds) | 17.74 | 55.59 | 8.35 | 44.69 |

As expected, the results show that TCP is more reliable than UDP. In the case of both TCP with ACK and without ACK, all data is received by the server. Only when using UDP, some packages are lost. However, UDP is faster than TCP. The time taken to exchange messages depends on the buffer size (it is reduced by half when doubling the buffer size).

In my case, when sending photos, TCP is preferred in order to be able to save the pictures on the server, as the correct order of the packages is ensured and none are lost. The steps include sending the file size and file name, so that the server can reconstruct each photo as it is sent by the client. When using the stop-and-wait mechanism, no additional checking is needed but for streaming, the client needs to know when the server has received the entire file before starting to send another. The server sends a message to the client (“sent”) when it has received *file\_size* bytes.