SAFE - UDP PROTOCOL

Team Members

Group 1

Name	ID
Radhesh Sarma	2017B4A70886H
Simran Sahni	2017B5A70856H
Kotikalapudi Venkat Karthik	2017B4A70927H
Ashi Sinha	2017B5A71149H
Chatrik Singh S Mangat	2017B5A70822H

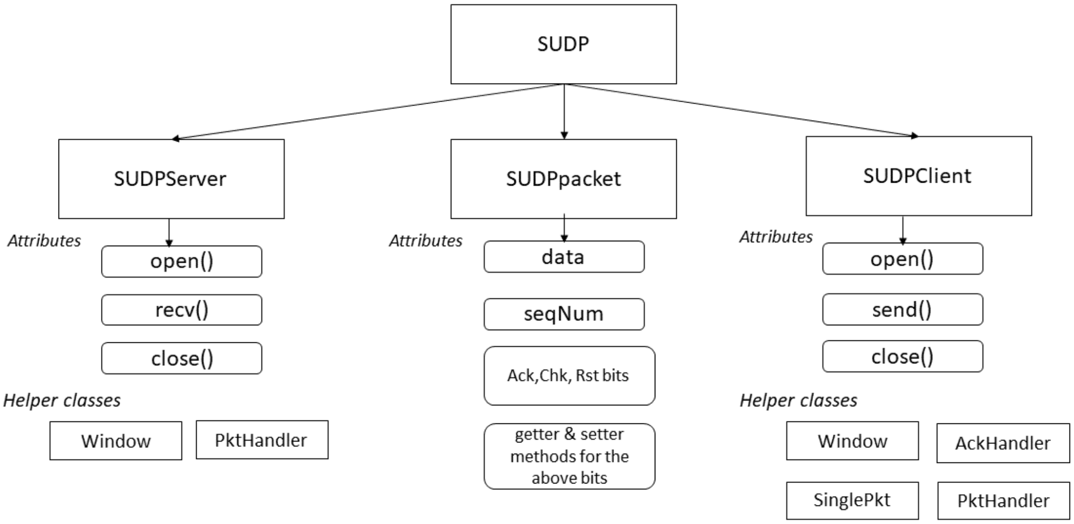
Implementation Details:

As per our proposed design document, the main aim of this modified implementation over the UDP protocol is to ensure reliability. We achieve this by using the Selective Repeat paradigm over the UDP as the transport layer protocol.

The toy application: Uploading service, that transfer/uploads 1 file at a time from the client to the server

Language used: Python

Overview of the protocol implementation files:



This SUDP protocol sends over the data in chunks of size 4096 for both client and server, and a window(ie. a OrderedDict) size of 256.
The checksum is calculated using the 'hashlib' module of python, & is used to check for the corruption of data.
Timer limit of 30 ms is used by the sender to wait for the acknowledgement receipt of a packet, corrupted and dropped packets simply timeout and are resent.
The pickle module is used to ensure data packet being sent as an object and retain its attributes. This module dumps the data in byte form for transmission via the socket.
You can send any type of file from the client to the server on being prompted to. Just ensure the same file format is in the code of the SUDPServer file.
Range of sequence numbers is: [0, 2^8 - 1]

Testing

Simulated and tested on various parameters and config using the tool: **mininet**
On 2 sample hosts named: laptop1 & laptop2

To install Mininet

```
pip install mininet
```

To test with desired bandwidth settings

```
sudo python config.py
```

Mininet will run and setup 2 hosts, laptop1 and laptop2

To run Client from laptop1

```
mininet> laptop1 python client.py
```

To run Server from laptop2

```
mininet> laptop2 python server.py
```

To change connection settings , go to Line 27 of config.py and change

For Example:

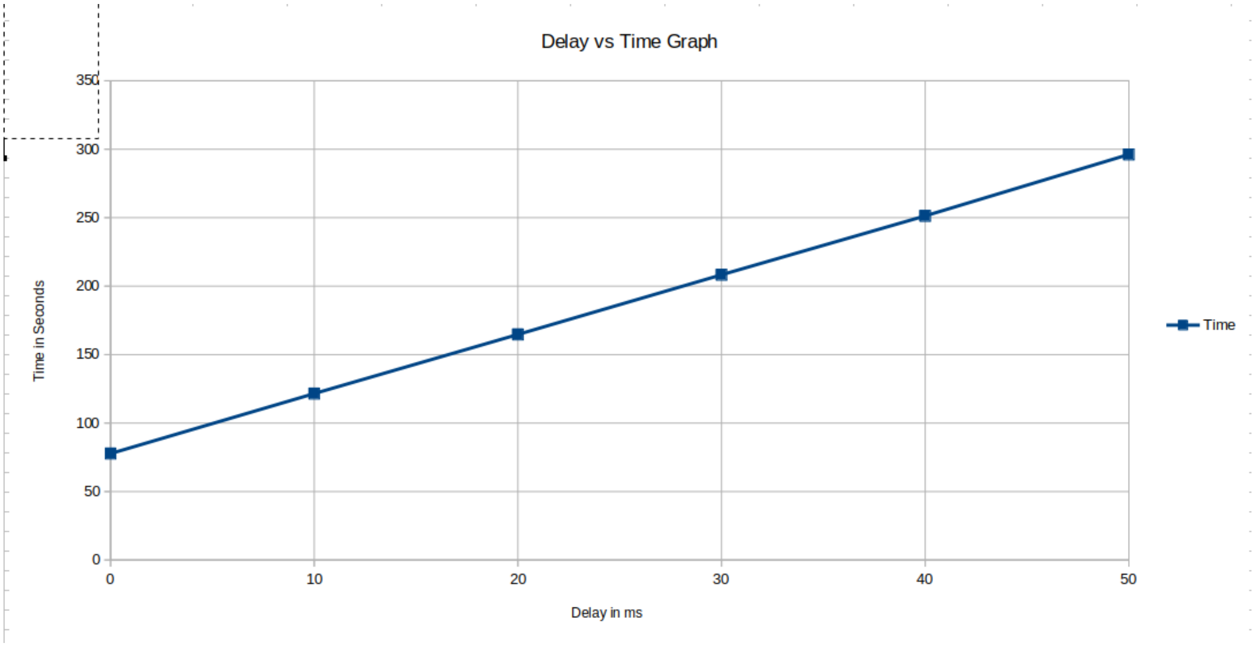
15 Mbps bandwidth and 2 ms delay on each link

```
linkopt = dict(bw=15, delay='2ms', loss=0)
```

Observations on tests:

Network delay induced against a constant bandwidth and time graph

bandwidth(mbps)	delay(ms)	Time(sec)
1000	0	77.71241188
1000	10	121.5177307
1000	20	164.6800358
1000	30	208.2847023
1000	40	251.2903602
1000	50	296.186737



Packet corruption with time

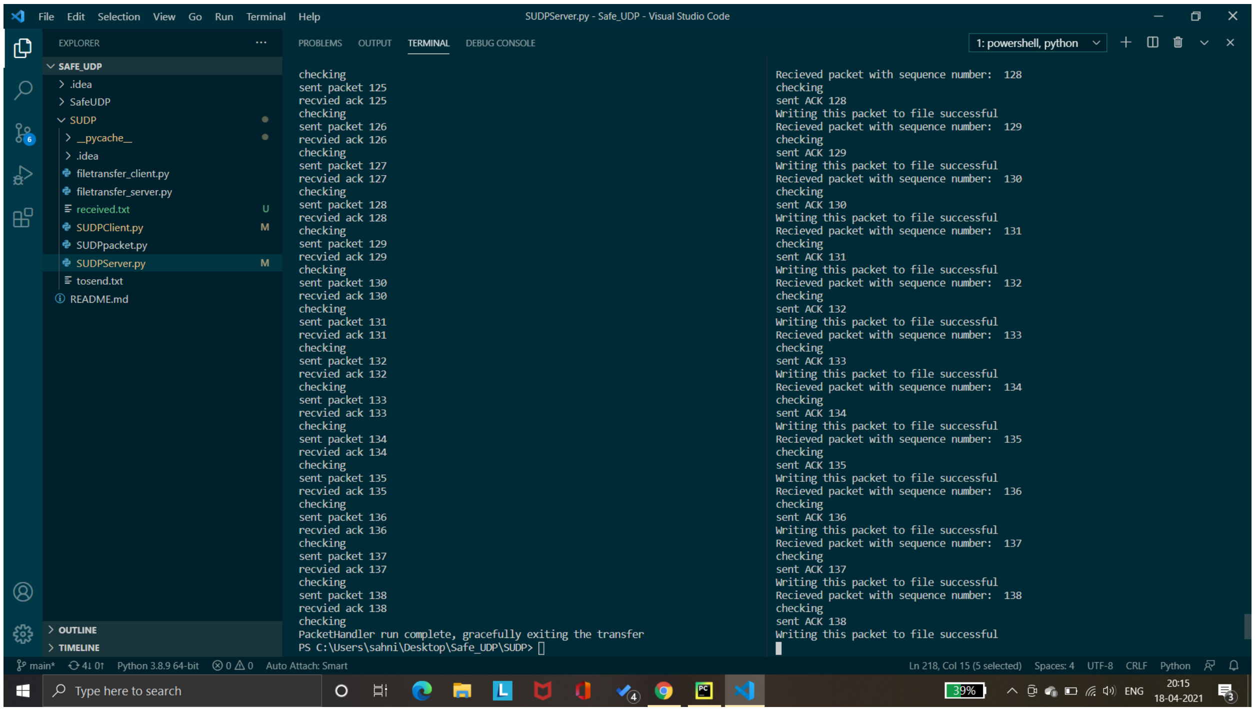
bandwidth(mbps)	Packet Corruption	Time(sec)
1000	0	87.5677188
1000	10	125.2346307
1000	20	169.8674558
1000	30	210.8576023

We observe that time significantly increases with even slight introduction of corrupt packets

Future scope:
1)Studies to optimize & observe performance using various bandwidths
2)Optimization in the code to improve performance in cases of such externalities, handling packet losses and corruptions quicker, using faster data-structures or optimal size of buffer windows
3) Efficient reset of the connection on setting of the Reset bit

Snapshot of the run:

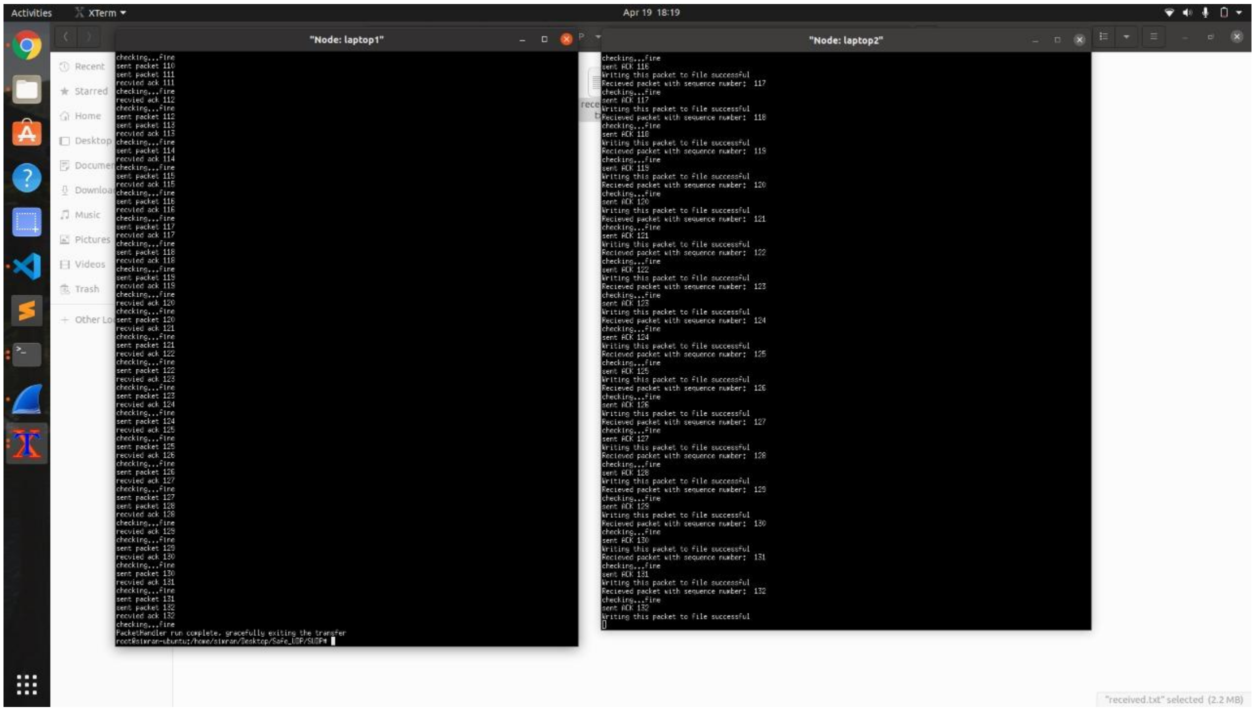
1)localhost



```
checking
sent packet 125
recvied ack 125
checking
sent packet 126
recvied ack 126
checking
sent packet 127
recvied ack 127
checking
sent packet 128
recvied ack 128
checking
sent packet 129
recvied ack 129
checking
sent packet 130
recvied ack 130
checking
sent packet 131
recvied ack 131
checking
sent packet 132
recvied ack 132
checking
sent packet 133
recvied ack 133
checking
sent packet 134
recvied ack 134
checking
sent packet 135
recvied ack 135
checking
sent packet 136
recvied ack 136
checking
sent packet 137
recvied ack 137
checking
sent packet 138
recvied ack 138
checking
PacketHandler run complete, gracefully exiting the transfer
PS C:\Users\sahni\Desktop\Safe_UDP\SUDP> 
```

```
Recieved packet with sequence number:  128
checking
sent ACK 128
Writing this packet to file successful
Recieved packet with sequence number:  129
checking
sent ACK 129
Writing this packet to file successful
Recieved packet with sequence number:  130
checking
sent ACK 130
Writing this packet to file successful
Recieved packet with sequence number:  131
checking
sent ACK 131
Writing this packet to file successful
Recieved packet with sequence number:  132
checking
sent ACK 132
Writing this packet to file successful
Recieved packet with sequence number:  133
checking
sent ACK 133
Writing this packet to file successful
Recieved packet with sequence number:  134
checking
sent ACK 134
Writing this packet to file successful
Recieved packet with sequence number:  135
checking
sent ACK 135
Writing this packet to file successful
Recieved packet with sequence number:  136
checking
sent ACK 136
Writing this packet to file successful
Recieved packet with sequence number:  137
checking
sent ACK 137
Writing this packet to file successful
Recieved packet with sequence number:  138
checking
sent ACK 138
Writing this packet to file successful
```

2)On 2 hosts using mininet



"Node: laptop1"

```
checking...fine
sent packet 110
sent packet 111
recvied ack 111
checking...fine
recvied ack 112
checking...fine
sent packet 112
sent packet 113
recvied ack 113
checking...fine
sent packet 114
recvied ack 114
checking...fine
sent packet 115
recvied ack 115
checking...fine
sent packet 116
recvied ack 116
checking...fine
sent packet 117
recvied ack 117
checking...fine
sent packet 118
recvied ack 118
checking...fine
sent packet 119
recvied ack 119
checking...fine
recvied ack 120
checking...fine
sent packet 120
recvied ack 121
checking...fine
sent packet 121
recvied ack 122
checking...fine
sent packet 122
recvied ack 123
checking...fine
sent packet 123
recvied ack 124
checking...fine
sent packet 124
recvied ack 125
checking...fine
sent packet 125
recvied ack 126
checking...fine
sent packet 126
recvied ack 127
checking...fine
sent packet 127
sent packet 128
recvied ack 128
checking...fine
recvied ack 129
checking...fine
sent packet 129
recvied ack 130
checking...fine
sent packet 130
recvied ack 131
checking...fine
sent packet 131
sent packet 132
recvied ack 132
checking...fine
PacketHandler run complete, gracefully exiting the transfer
root@simran-ubuntu:/home/simran/Desktop/Safe_UDP/SUDP# 
```

"Node: laptop2"

```
checking...fine
sent ACK 116
Writing this packet to file successful
Recieved packet with sequence number:  117
checking...fine
sent ACK 117
Writing this packet to file successful
Recieved packet with sequence number:  118
checking...fine
sent ACK 118
Writing this packet to file successful
Recieved packet with sequence number:  119
checking...fine
sent ACK 119
Writing this packet to file successful
Recieved packet with sequence number:  120
checking...fine
sent ACK 120
Writing this packet to file successful
Recieved packet with sequence number:  121
checking...fine
sent ACK 121
Writing this packet to file successful
Recieved packet with sequence number:  122
checking...fine
sent ACK 122
Writing this packet to file successful
Recieved packet with sequence number:  123
checking...fine
sent ACK 123
Writing this packet to file successful
Recieved packet with sequence number:  124
checking...fine
sent ACK 124
Writing this packet to file successful
Recieved packet with sequence number:  125
checking...fine
sent ACK 125
Writing this packet to file successful
Recieved packet with sequence number:  126
checking...fine
sent ACK 126
Writing this packet to file successful
Recieved packet with sequence number:  127
checking...fine
sent ACK 127
Writing this packet to file successful
Recieved packet with sequence number:  128
checking...fine
sent ACK 128
Writing this packet to file successful
Recieved packet with sequence number:  129
checking...fine
sent ACK 129
Writing this packet to file successful
Recieved packet with sequence number:  130
checking...fine
sent ACK 130
Writing this packet to file successful
Recieved packet with sequence number:  131
checking...fine
sent ACK 131
Writing this packet to file successful
Recieved packet with sequence number:  132
checking...fine
sent ACK 132
Writing this packet to file successful
```

Link to the github repository:
https://github.com/Simran-Sahni/Safe_UDP

Bibliography:

1) https://github.com/chetanborse007/SelectiveRepeat
2) RUDP RFC