

# Test

In the test section, we design tests to verify the ability of LFTP to exchange file between clients and servers and serve multiple clients at the same time. Also, we'll exam the reliability of RDP. Besides, flow control and congestion control of RDP won't be absent in our test.

## LFTP

### Test Environment

- Client: Deployed at Macbook Pro 13' 2017 with MacOS Mojave connected with the SYSU-Secure WiFi in the east public classroom B104
- Server: Deployed at an i7-4250hq based server with Windows 10 1903 connected with the LAN in Zhishanyuan Dormitory 705

### Send File to Server

In this test, we try to send a 1.79 GB file and a 3 bytes file to check whether LFTP clients could send both large files and small files.

- 1.79 GB file, client side log

```
SEND: Fragment-39869 sends successfully!(Move Window)
SEND-RWND: 76000
39870 39870 39866
SEND: Data sends successfully
----- END SEND -----

Sent: 1794060040 bytes
Total: 1794060040 bytes
Sending file testvideo.mp4: 100% done.
Speed: 2637 KB/second
Sending done.
```

- 1.79 GB file, server side log

```
Recv: 1794060040 bytes
Total: 1794060040 byets
Receiving testvideo.mp4: 100% data received...
Speed: 2614 KB/s
Receiving testvideo.mp4: Done
Receiving testvideo.mp4: File Lock Released.
```

As you can see, the large file is successfully sent from the client to the server at the speed of about 2.6 MB/s. During sending, we met several timeout issues. But our protocol is strong enough to handle these condition and deliver the file safely to the server. (We can also open it at the server side!)

Along with supporting the transporting of large files, small files are also supported by our application:

- 3 bytes file, client side log

```
Sent: 3 bytes
Total: 3 bytes
Sending file 123: 100% done.
Speed: 0 KB/second
Sending done.
Emilia:code siskon$ █
```

- 3 bytes file, server side log

```
RCV_Packet: 1
back: 0
RECV: Window start pkt(SeqNum:1), buffer continual data
ACK pkt: 1
Recv: 3 bytes
Total: 3 byets
Receiving 123: 100% data received...
Speed: 0 KB/s
Receiving 123: Done
Receiving 123: File Lock Released.
```

Like what happened to the 1.79GB video, the 3 bytes file is safely delivered to the server. (The content string '123' stays unchanged!)

# Get File from Server

Besides sending files, getting files is well supported by our program. We use the files we sent earlier to test this function:

- 1.79 GB file, client side log

```
RCV: waiting packets from (172.18.35.186:58637)...  
RCV: No data received from (172.18.35.186:58637). Finish  
Received Length: 3960056  
Recv: 1794060040 bytes  
Total: 1794060040 bytes  
Receiving testvideo.mp4: 100% data received...  
Speed: 3485 KB/s  
Receiving testvideo.mp4: Done  
Emilia:code siskon$
```

- 1.79 GB file, server side log

```
Sent: 1794060040 bytes  
Total: 1794060040 bytes  
Sending file testvideo.mp4: 100% done.  
Speed: 3526 KB/s  
Sending done.  
Sending testvideo.mp4: Acquiring Reading Lock...  
Sending testvideo.mp4: Reading Lock Released.
```

- 3 bytes file, client side log

```
RCV_Packet: 0  
back: 0  
RCV: Window start pkt(SeqNum:0), buffer continual data  
ACK pkt: 0  
Response from server: OK3  
RCV_Packet: 1  
back: 0  
RCV: Window start pkt(SeqNum:1), buffer continual data  
ACK pkt: 1  
Received Length: 4  
Recv: 3 bytes  
Total: 3 bytes  
Receiving 123: 100% data received...  
Speed: 0 KB/s  
Receiving 123: Done
```

- 3 bytes file, server side log

```
Sent: 3 bytes
Total: 3 bytes
Sending file 123: 100% done.
Speed: 0 KB/s
Sending done.
Sending 123: Acquiring Reading Lock...
Sending 123: Reading Lock Released.
```

As we post above, the server send the file to our client at a higher speed (about 3.5 MB/s). Also, the content of files is transported perfectly without being damaged.

## Local Maximum Speed

A great question is that how fast can our program be when the network environment is perfect. So we test this program at local network (loopback adapter, tested on Macbook Pro). The result is incredible:

- Speed test, server side log:

```
Sent: 1794060040 bytes
Total: 1794060040 bytes
Sending file testvideo.mp4: 100% done.
Speed: 55050 KB/s
Sending done.
Sending testvideo.mp4: Acquiring Reading Lock...
Sending testvideo.mp4: Reading Lock Released.
```

- Speed test, client side log:

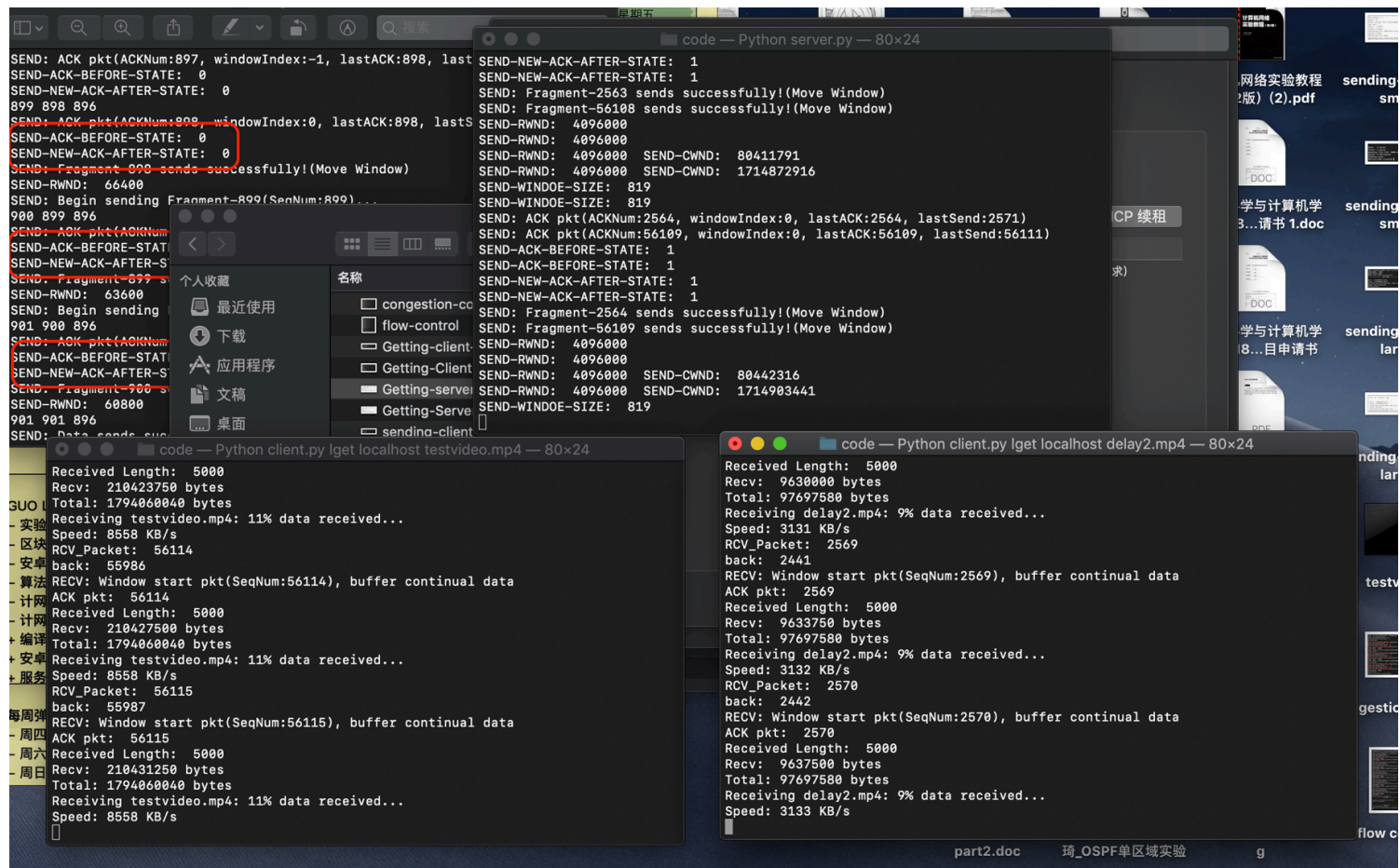
```
Recv: 1794060040 bytes
Total: 1794060040 bytes
Receiving testvideo.mp4: 100% data received...
Speed: 55050 KB/s
Receiving testvideo.mp4: Done
```

The file is sent at the speed of 55 MB/s, which can milking a LAN of about 500 Mbps dry. That means our program can be applied to many types of high-performance network and makes it work at its best speed.



# Multiple Users

To test the ability of our software to support multiple users' requests at the same time, we create two clients and make them get files at the same time:



```
SEND: ACK pkt(ACKNum:897, windowIndex:-1, lastACK:898, lastS
SEND-ACK-BEFORE-STATE: 0
SEND-NEW-ACK-AFTER-STATE: 0
899 898 896
SEND: ACK pkt(ACKNum:898, windowIndex:0, lastACK:898, lastS
SEND-ACK-BEFORE-STATE: 0
SEND-NEW-ACK-AFTER-STATE: 0
SEND: Fragment-898 sends successfully!(Move Window)
SEND-RWND: 66400
SEND: Begin sending Fragment-899(SenNum:899)...
900 899 896
SEND: ACK pkt(ACKNum:
SEND-ACK-BEFORE-STATE: 1
SEND-NEW-ACK-AFTER-S
SEND: Fragment-899 s
SEND-RWND: 63600
SEND: 901 900 896
SEND: ACK pkt(ACKNum:
SEND-ACK-BEFORE-STATE: 1
SEND-NEW-ACK-AFTER-S
SEND: Fragment-900 s
SEND-RWND: 60800
901 901 896
SEND: Data sends suc

SEND-NEW-ACK-AFTER-STATE: 1
SEND-NEW-ACK-AFTER-STATE: 1
SEND: Fragment-2563 sends successfully!(Move Window)
SEND: Fragment-56108 sends successfully!(Move Window)
SEND-RWND: 4096000
SEND-RWND: 4096000
SEND-RWND: 4096000 SEND-CWND: 80411791
SEND-RWND: 4096000 SEND-CWND: 1714872916
SEND-WINDOE-SIZE: 819
SEND-WINDOE-SIZE: 819
SEND: ACK pkt(ACKNum:2564, windowIndex:0, lastACK:2564, lastSend:2571)
SEND: ACK pkt(ACKNum:56109, windowIndex:0, lastACK:56109, lastSend:56111)
SEND-ACK-BEFORE-STATE: 1
SEND-NEW-ACK-AFTER-STATE: 1
SEND-NEW-ACK-AFTER-STATE: 1
SEND: Fragment-2564 sends successfully!(Move Window)
SEND: Fragment-56109 sends successfully!(Move Window)
SEND-RWND: 4096000
SEND-RWND: 4096000
SEND-RWND: 4096000 SEND-CWND: 80442316
SEND-RWND: 4096000 SEND-CWND: 1714903441
SEND-WINDOE-SIZE: 819

Received Length: 5000
Recv: 210423750 bytes
Total: 1794060040 bytes
Receiving testvideo.mp4: 11% data received...
Speed: 8558 KB/s
RCV_Packet: 56114
back: 55986
RCV: Window start pkt(SeqNum:56114), buffer continual data
ACK pkt: 56114
Received Length: 5000
Recv: 210427500 bytes
Total: 1794060040 bytes
Receiving testvideo.mp4: 11% data received...
Speed: 8558 KB/s
RCV_Packet: 56115
back: 55987
RCV: Window start pkt(SeqNum:56115), buffer continual data
ACK pkt: 56115
Received Length: 5000
Recv: 210431250 bytes
Total: 1794060040 bytes
Receiving testvideo.mp4: 11% data received...
Speed: 8558 KB/s

Received Length: 5000
Recv: 9630000 bytes
Total: 97697580 bytes
Receiving delay2.mp4: 9% data received...
Speed: 3131 KB/s
RCV_Packet: 2569
back: 2441
RCV: Window start pkt(SeqNum:2569), buffer continual data
ACK pkt: 2569
Received Length: 5000
Recv: 9633750 bytes
Total: 97697580 bytes
Receiving delay2.mp4: 9% data received...
Speed: 3132 KB/s
RCV_Packet: 2570
back: 2442
RCV: Window start pkt(SeqNum:2570), buffer continual data
ACK pkt: 2570
Received Length: 5000
Recv: 9637500 bytes
Total: 97697580 bytes
Receiving delay2.mp4: 9% data received...
Speed: 3133 KB/s
```

The requests are handles properly and the transfer speed is extremely high.

## RDP

## Flow Control

Flow control changes the sending rate according to the receive window size in the receiver side. In the test, we deliberately set receive side application retrieves data smaller to let data buffered in the buffer and results in the change of the rwnd. We log out the rwnd received in the sender side to see whether the rwnd changed. Obviously, the rwnd changed, which means flow control works!

```
code — Python client.py lsend 172.18.35.186:8080 delay2.mp4 — 80x45
SEND: ACK pkt(ACKNum:571, windowIndex:0, lastACK:571, lastSend:572)
SEND-ACK-BEFORE-STATE: 1
SEND-NEW-ACK-AFTER-STATE: 1
SEND: Fragment-571 sends successfully!(Move Window)
SEND-RWND: 72000
SEND: Begin sending Fragment-572(SeqNum:572)...
573 572 571
SEND: ACK pkt(ACKNum:572, windowIndex:0, lastACK:572, lastSend:573)
SEND-ACK-BEFORE-STATE: 1
SEND-NEW-ACK-AFTER-STATE: 1
SEND: Fragment-572 sends successfully!(Move Window)
SEND-RWND: 69200
SEND: Begin sending Fragment-573(SeqNum:573)...
574 573 571
SEND: ACK pkt(ACKNum:573, windowIndex:0, lastACK:573, lastSend:574)
SEND-ACK-BEFORE-STATE: 1
SEND-NEW-ACK-AFTER-STATE: 1
SEND: Fragment-573 sends successfully!(Move Window)
SEND-RWND: 66400
SEND: Begin sending Fragment-574(SeqNum:574)...
575 574 571
SEND: ACK pkt(ACKNum:574, windowIndex:0, lastACK:574, lastSend:575)
SEND-ACK-BEFORE-STATE: 1
SEND-NEW-ACK-AFTER-STATE: 1
SEND: Fragment-574 sends successfully!(Move Window)
SEND-RWND: 63600
SEND: Begin sending Fragment-575(SeqNum:575)...
576 575 571
SEND: ACK pkt(ACKNum:575, windowIndex:0, lastACK:575, lastSend:576)
SEND-ACK-BEFORE-STATE: 1
SEND-NEW-ACK-AFTER-STATE: 1
SEND: Fragment-575 sends successfully!(Move Window)
SEND-RWND: 60800
576 576 571
SEND: Data sends successfully
----- END SEND -----

Sending file delay2.mp4: 24% done.
Speed: 75 KB/second

----- BEGIN SEND -----
('172.18.35.186', 44010)
SEND: Begin sending Fragment-576(SeqNum:576)...
```

## Congestion Control

Congestion Control is tuning the state of sender's among slow start, congestion avoidance and fast recovery. In our application, we design these three states into three numbers, 0, 1, 2 representing slow start, congestion avoidance and fast recovery in order.



We log out the sender information in the terminal to see whether the congestion control state changed. As we can see, the state at the beginning is 0, slow start and then it changed into 1, congestion avoidance state. This shows the success of the implementation of congestion control in our application.

Also, it works to control the send rate while using.

```
SEND: ACK pkt(ACKNum:897, windowIndex:-1, lastACK:898, lastSend:899)
SEND-ACK-BEFORE-STATE: 0
SEND-NEW-ACK-AFTER-STATE: 0
899 898 896
SEND: ACK pkt(ACKNum:898, windowIndex:0, lastACK:898, lastSend:899)
SEND-ACK-BEFORE-STATE: 0
SEND-NEW-ACK-AFTER-STATE: 0
SEND: Fragment 898 sends successfully!(Move Window)
SEND-RWND: 66400
SEND: Begin sending Fragment-899(SeqNum:899)...
900 899 896
SEND: ACK pkt(ACKNum:899, windowIndex:0, lastACK:899, lastSend:900)
SEND-ACK-BEFORE-STATE: 0
SEND-NEW-ACK-AFTER-STATE: 1
SEND: Fragment 899 sends successfully!(Move Window)
SEND-RWND: 63600
SEND: Begin sending Fragment-900(SeqNum:900)...
901 900 896
SEND: ACK pkt(ACKNum:900, windowIndex:0, lastACK:900, lastSend:901)
SEND-ACK-BEFORE-STATE: 1
SEND-NEW-ACK-AFTER-STATE: 1
SEND: Fragment 900 sends successfully!(Move Window)
SEND-RWND: 60800
901 901 896
SEND: Data sends successfully
```