

测试文档

16340176 潘鉴

lget 指令

这条命令的发送方是 Server，接收端是 Client。

运行 Client.py 代码，将提示你要输入的命令。

倘若命令不正确，将提示你命令格式不正确。

```
请输入命令: LFTP [lsend | lget] server_address file_name
LFTP lge 10.1.1.221 test.mp4
[Error] Invalid command!
```

倘若没有对应的文件，则服务器将返回 “fileNotExists” 的消息。

```
请输入命令: LFTP [lsend | lget] server_address file_name
LFTP lget 10.1.1.221 test.mp4
来自 ('10.1.1.221', 49529) 的消息是: fileNotExists
```

当输入了正确的命令，Client 向 Server 发送连接请求，倘若连接成功，Server 则返回 “连接允许” 的消息，表示 Server 准备好了。Client 则向 Server 发送 “请求数据” 的消息，表示 Client 准备好了接收数据了。Server 收到这个消息后立即发送数据。

之后便是接收数据的过程。当 Client 收到数据时，就会先把数据放进接收缓存里，当没有收到数据了，就从缓存里取出数据处理，处理后发送 ACK 确认包给 Server。

```
向 ('10.1.1.221', 49543) 发送请求数据
正在接收 test.mp4
收到数据包0
处理数据包0
发送ACK0
收到数据包1
处理数据包1
发送ACK1
收到数据包2
处理数据包2
发送ACK2
收到数据包3
收到数据包4
处理数据包3
发送ACK3
处理数据包4
发送ACK4
收到数据包5
处理数据包5
发送ACK5
收到数据包6
处理数据包6
发送ACK6
收到数据包7
处理数据包7
```

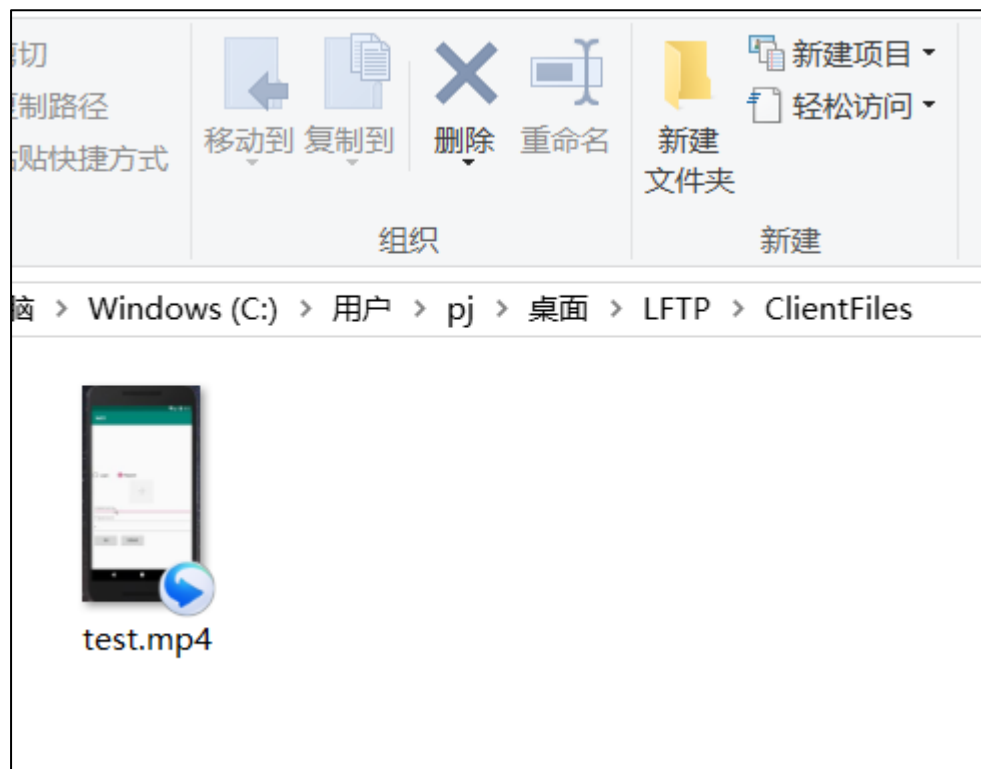
```
收到数据包200
收到数据包201
收到数据包202
收到数据包203
收到数据包204
收到数据包205
处理数据包200
发送ACK200
处理数据包201
发送ACK201
处理数据包202
发送ACK202
处理数据包203
发送ACK203
处理数据包204
发送ACK204
处理数据包205
发送ACK205
收到数据包206
处理数据包206
发送ACK206
收到数据包207
收到数据包208
处理数据包207
发送ACK207
处理数据包208
```

当接收完后，Client 会统计好成功接收的数据量，和总共花费的时间。

```
收到数据包266
处理数据包266
发送ACK266
收到数据包267
处理数据包267
成功接收的数据包数量：267
总共花费时间为:3.096711秒
```

其中包 267 检测到是结束标记，不必发送 ACK 包。

这时本地的目录下会有一个 ClientFiles 的文件一夹，里面保存着所接收的文件。



Isend 指令

这条命令的发送方是 Client，接收端是 Server。

同样要输入正确的指令和正确的本地文件名，否则会报错。

同样先向 Server 发送“请求连接”的数据包，Server 回一个“连接允许”的数据包，之后向 Server 发送“请求接收”的数据包，只有当 Server 回一个“接收允许”的时候才开始发送数据。

```
请输入命令: LFTP [lsend | lget] server_address file_name
LFTP lsend 10.1.1.221 test.mp4
来自 ('127.0.0.1', 61068) 的消息是: 连接允许
LFTP lsend ('127.0.0.1', 61068) test.mp4
来自 ('127.0.0.1', 61068) 的数据是: 接收允许
正在发送 test.mp4
packet 0加入缓冲区
send packet:0
```

要发送的数据先放进发送缓存里，同时监测来自接收方的 ACK 确认包，没有收到 ACK 包时就从缓存里发送包。

```
receive ack packet:224
-----cwnd: 25
-----rwnd: 43
packet 224~224从缓冲区删除
receive ack packet:225
-----cwnd: 26
-----rwnd: 44
packet 225~225从缓冲区删除
packet 250加入缓冲区
send packet:250
packet 251加入缓冲区
send packet:251
packet 252加入缓冲区
send packet:252
packet 253加入缓冲区
send packet:253
receive ack packet:226
-----cwnd: 27
-----rwnd: 45
packet 226~226从缓冲区删除
packet 254加入缓冲区
send packet:254
packet 255加入缓冲区
send packet:255
```

发送完了会统计包数和花费的时间。

```
packet 48583加入缓冲区
send packet:48583
packet 48584加入缓冲区
send packet:48584
packet 48585加入缓冲区
send packet:48585
packet 48586加入缓冲区
send packet:48586
packet 48587加入缓冲区
send packet:48587
=====48587=====
test6.mp4 发送完毕，发送数据包的数量：48587
总共花费时间为:48.359963秒
```

接收端统计接收数量

```
rwnd 43
处理数据包48582
发送ACK48582
rwnd 44
处理数据包48583
发送ACK48583
rwnd 45
处理数据包48584
发送ACK48584
rwnd 46
处理数据包48585
发送ACK48585
rwnd 47
处理数据包48586
发送ACK48586
rwnd 48
处理数据包48587
成功接收的数据包数量：48587
```

可靠性传输的测试

首先传输得到的数据是是可以正常打开的。

其次分析丢包的情况发送端与接收端的动作。

通过打印的过程，我们可以看到，Seq 为 200 的包丢掉了，因此接收方一直发送 ACK199，尽管它收到大于 199 的包。

```
发送ACK199
收到数据包210
收到数据包211
收到数据包212
收到数据包213
收到数据包214
收到数据包215
rwnd 43
处理数据包210
发送ACK199
rwnd 44
处理数据包211
发送ACK199
rwnd 45
处理数据包212
发送ACK199
rwnd 46
处理数据包213
发送ACK199
rwnd 47
处理数据包214
发送ACK199
rwnd 48
处理数据包215
发送ACK199
收到数据包216
rwnd 48
处理数据包216
发送ACK199
收到数据包217
rwnd 48
处理数据包217
发送ACK199
收到数据包218
收到数据包219
收到数据包220
收到数据包221
```

而发送方一直收到 ACK199, 则触发 seq200 数据包的超时函数, 重新发送 200 以后的包。

```

-----rwnd: 46
packet 198~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 47
packet 199~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 48
packet 200~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 41
packet 200~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 42
packet 200~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 43
packet 200~199从缓冲区删除
receive ack packet:199

```

```

-----cwnd: 67
-----rwnd: 47
packet 200~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 48
packet 200~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 48
packet 200~199从缓冲区删除
发送超时，出现丢包，重新发送分组
更新cwnd值为1 更新sssthresh值为33
重新 send packet:200~248
重新发送packet: 200
重新发送packet: 201
重新发送packet: 202
重新发送packet: 203
重新发送packet: 204
重新发送packet: 205

```

流控制的测试

将接收缓存的容量设为 50。通过打印 rwnd 值可以看到, rwnd 在 40 多的居多。

```

-----rwnd: 48
packet 200~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 44
packet 200~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 45
packet 200~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 46
packet 200~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 47
packet 200~199从缓冲区删除
receive ack packet:199
-----cwnd: 67
-----rwnd: 48

```

分析原因，是因为接收方的处理速度与发送方的发送速度大致相同。现在为了分析流控制是否起作用，我们将接收方的处理数据降低，每处理完一次，将用 `time.sleep` 停止 0.01 秒。

packet 168~168从缓冲区删除 receive ack packet:169 -----cwnd: 66 -----rwnd: 14 packet 169~169从缓冲区删除 receive ack packet:170 -----cwnd: 66 -----rwnd: 15 packet 170~170从缓冲区删除 receive ack packet:171 -----cwnd: 66 -----rwnd: 16 packet 171~171从缓冲区删除 receive ack packet:172 -----cwnd: 66 -----rwnd: 17 packet 172~172从缓冲区删除 receive ack packet:173 -----cwnd: 66 -----rwnd: 18 packet 173~173从缓冲区删除 receive ack packet:174 -----cwnd: 66 -----rwnd: 19 packet 174~174从缓冲区删除 receive ack packet:175 -----cwnd: 66 -----rwnd: 20 packet 175~175从缓冲区删除 receive ack packet:176 -----cwnd: 66 -----rwnd: 21	packet 52~52从缓冲区删除 packet 85加入缓冲区 send packet:85 packet 86加入缓冲区 send packet:86 packet 87加入缓冲区 send packet:87 packet 88加入缓冲区 send packet:88 packet 89加入缓冲区 send packet:89 packet 90加入缓冲区 send packet:90 receive ack packet:53 -----cwnd: 52 -----rwnd: 38 packet 53~53从缓冲区删除 packet 91加入缓冲区 send packet:91 packet 92加入缓冲区 send packet:92 receive ack packet:54 -----cwnd: 53 -----rwnd: 39 packet 54~54从缓冲区删除 packet 93加入缓冲区 send packet:93 packet 94加入缓冲区
--	--

可以看到，由于接收方的处理速度降低，`rwnd` 值不再是 40 多，大多在 10-50 的范围之内。因为收到 ACK 包的速率慢，因此也出现了发送方连续发多个包的情况。

将接收缓存的大小改为 10 个数据包，也出现相似的结果。

packet 2518~2518从缓冲区删除	receive ack packet:2526
packet 2527加入缓冲区	-----cwnd: 9
send packet:2527	-----rwnd: 8
receive ack packet:2519	packet 2526~2526从缓冲区删除
-----cwnd: 9	receive ack packet:2527
-----rwnd: 1	-----cwnd: 9
packet 2519~2519从缓冲区删除	-----rwnd: 8
receive ack packet:2520	packet 2527~2527从缓冲区删除
-----cwnd: 9	packet 2528加入缓冲区
-----rwnd: 2	send packet:2528
packet 2520~2520从缓冲区删除	packet 2529加入缓冲区
receive ack packet:2521	send packet:2529
-----cwnd: 9	packet 2530加入缓冲区
-----rwnd: 3	send packet:2530
packet 2521~2521从缓冲区删除	packet 2531加入缓冲区
receive ack packet:2522	send packet:2531
-----cwnd: 9	packet 2532加入缓冲区
-----rwnd: 4	send packet:2532
packet 2522~2522从缓冲区删除	packet 2533加入缓冲区
receive ack packet:2523	send packet:2533
-----cwnd: 9	packet 2534加入缓冲区
-----rwnd: 5	send packet:2534
packet 2523~2523从缓冲区删除	packet 2535加入缓冲区
receive ack packet:2524	send packet:2535
-----cwnd: 9	packet 2536加入缓冲区
-----rwnd: 6	send packet:2536
	receive ack packet:2528

最终收到的数据均正常，以上结果说明流控制是起作用的。

拥塞控制的测试

因为最终代码是同时有流控制和拥塞控制的，为了单独测试拥塞控制，应该要使得流控制没有影响。从上面的结果可以看到，当接收方和发送方速度相差不大时，流控制的作用很小。而且 cwnd 比较大，说明当前测试的环境并没有出现拥塞。为了测试拥塞控制，使用软件将接收方的下载速度限制为 200k/s（正常是 1M/s）。

<pre> receive ack packet:2599 -----cwnd: 18 -----rwnd: 48 packet 2599~2599从缓冲区删除 receive ack packet:2600 -----cwnd: 18 -----rwnd: 48 packet 2600~2600从缓冲区删除 packet 2601加入缓冲区 send packet:2601 packet 2602加入缓冲区 send packet:2602 packet 2603加入缓冲区 send packet:2603 packet 2604加入缓冲区 send packet:2604 packet 2605加入缓冲区 send packet:2605 packet 2606加入缓冲区 send packet:2606 packet 2607加入缓冲区 send packet:2607 packet 2608加入缓冲区 send packet:2608 packet 2609加入缓冲区 send packet:2609 packet 2610加入缓冲区 send packet:2610 packet 2611加入缓冲区 send packet:2611 packet 2612加入缓冲区 send packet:2612 packet 2613加入缓冲区 send packet:2613 </pre>	<pre> send packet:2671 receive ack packet:2673 -----cwnd: 21 -----rwnd: 45 packet 2673~2673从缓冲区删除 receive ack packet:2674 -----cwnd: 21 -----rwnd: 46 packet 2674~2674从缓冲区删除 packet 2695加入缓冲区 send packet:2695 packet 2696加入缓冲区 send packet:2696 receive ack packet:2675 -----cwnd: 21 -----rwnd: 47 packet 2675~2675从缓冲区删除 receive ack packet:2676 -----cwnd: 21 -----rwnd: 48 packet 2676~2676从缓冲区删除 receive ack packet:2677 -----cwnd: 22 -----rwnd: 48 packet 2677~2677从缓冲区删除 packet 2697加入缓冲区 send packet:2697 packet 2698加入缓冲区 send packet:2698 packet 2699加入缓冲区 send packet:2699 packet 2700加入缓冲区 send packet:2700 receive ack packet:2678 </pre>
--	--

可以看到现在 cwnd 值比先前明显小了，因为出现丢包的现象，很快就进入拥塞避免的状态，增加得比较慢。而一开始的时候还是翻倍增加。

-----cwnd: 1 -----rwnd: 48 packet 0~0从缓冲区删除 receive ack packet:1 -----cwnd: 2 -----rwnd: 48 packet 1~1从缓冲区删除 packet 2加入缓冲区 send packet:2 packet 3加入缓冲区 send packet:3 packet 4加入缓冲区 send packet:4 packet 5加入缓冲区 send packet:5 receive ack packet:2 -----cwnd: 3 -----rwnd: 47 packet 2~2从缓冲区删除 packet 6加入缓冲区 send packet:6 packet 7加入缓冲区 send packet:7 receive ack packet:3 -----cwnd: 4 -----rwnd: 48 packet 3~3从缓冲区删除 receive ack packet:4	packet 298~297从缓冲区删除 receive ack packet:297 -----cwnd: 35 -----rwnd: 46 packet 298~297从缓冲区删除 receive ack packet:297 -----cwnd: 35 -----rwnd: 47 packet 298~297从缓冲区删除 receive ack packet:297 -----cwnd: 35 -----rwnd: 48 packet 298~297从缓冲区删除 receive ack packet:297 -----cwnd: 35 -----rwnd: 48 packet 298~297从缓冲区删除 发送超时，出现丢包，重新发送分组 更新cwnd值为1 更新sssthresh值为17 重新 send packet:298~333 重新发送packet: 298 重新发送packet: 299 重新发送packet: 300 重新发送packet: 301 重新发送packet: 302 重新发送packet: 303 重新发送packet: 304 重新发送packet: 305 重新发送packet: 306 重新发送packet: 307 重新发送packet: 308 重新发送packet: 309 重新发送packet: 310
--	--

接收的速度明显变慢。但最终成功接收，没有出错。说明拥塞控制有效。

多线程的测试

```

来自 ('10.1.1.207', 56095) 的数据是:  lsend#test6.mp4
正在运行的线程数量:  4
等待客户端发起连接...
来自 ('10.1.1.207', 56095) 的数据是:  ACK
正在接收 test6.mp4
收到数据包0
收到数据包1
rwnd 7
处理数据包0

```

在有连接的情况下用另一台机器尝试新的连接，可以运行，接收文件正常。

