



MEDIATEK

Throughput Test Issue Analysis SOP

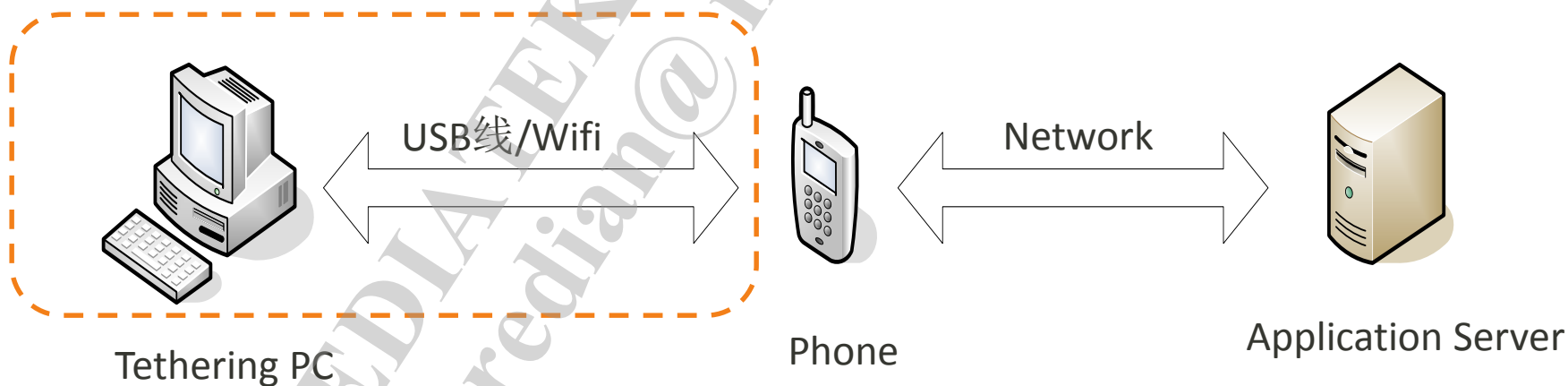
Outline

- 概述
- 影响Throughput的几个因素
- 如何做对比测试并获取有效的Log
- Throuhgput test issue 分析步骤和实例分析

概述

Overview

- Performance测试一般包括其上传和下载两方面
- 测试方式
 - FTP(Usb Tethering/Wifi Hotspot/AndFtp.apk)
 - HTTP(SpeedTest.apk)
 - 其它相关的测试(Browser, 运营商规定测试的APK)
- Debug tools
 - Wireshark



影响Throughput的几个因素

影响Throughput的几个因素

- User load和Eng load的测试结果不同。一般user load的测试结果会更好。
 - Mobilelog/APLog_xxx_xxx_xxx/ properties /ro.build.type] : [user]
- mtklog on/off的测试结果不同。一般mtklog off的结果会更好。
- Over Usb tethering测试时。请先确认其rndis的性能要达标。
 - 请参考补充部分：PC RNDIS performance check
- Window size的大小。Window Size对于tcp performance的测试影响很大，如果设置过小，需要重新设置window size.

影响Throughput的几个因素

- 测试APP版本不一样，导致其读写数据速率不一样.
- Modem和Network本身的限制.
- 文件系统对性能的影响
- Wifi本身性能的影响
 - 也是通过Iper测试其RX/TX
 - 测试步骤跟rndis基本相同，参考补充部分：PC RNDIS performance check

Window size的确认以及修改

- Window size确认
 - 可以通过Wireshark的方式

669040	95.073317000	192.168.1.9	1476	5001 TCP	1514 c/vm-cfg > complex-link [ACK] Seq=456007705 Ack=1 Win=65535 Len=1460
669041	95.073317000	192.168.1.9	1476	5001 TCP	946 c/vm-cfg > complex-link [PSH, ACK] Seq=456006813 Ack=1 Win=65535 Len=892
669042	95.073426000	192.168.1.11	5001	1476 TCP	60 complex-link > c/vm-cfg [ACK] Seq=1 Ack=45412354 Win=1887424 Len=0
669043	95.073447000	192.168.1.9	1476	5001 TCP	1514 c/vm-cfg > complex-link [ACK] Seq=456007705 Ack=1 Win=65535 Len=1460
669044	95.073450000	192.168.1.9	1476	5001 TCP	1514 c/vm-cfg > complex-link [ACK] Seq=456009165 Ack=1 Win=65535 Len=1460

- Window size设定
 - 查看注册表的方式(PC)
 - 通过Mobilelog/APLog_xx_xx/main_log or sys log查找关键字“Setting tx/rx TCP buffers” (mobile)

```
09-06 14:20:19.757224 418 1137 D SocketClient: SocketClient sendData done: 200 40 success
09-06 14:20:19.757439 1156 1281 D NetdConnector: RCV <- {200 40 success}
09-06 14:20:19.757863 1156 1288 D NetdConnector: RMV <- {200 40 success}
09-06 14:20:19.758294 1156 1288 D ConnectivityService: Setting tx/rx TCP buffers to 524288,1048576,2097152,262144,524288,1048576
09-06 14:20:19.764819 1156 2005 D DisplayManagerService: Display listener for pid 3187 died.
09-06 14:20:19.766533 1156 1203 V ActivityManager: Broadcast: Intent { act=android.net.conn.DATA_ACTIVITY_CHANGE flg=0x10 mCallingUid=1000 (has ex
09-06 14:20:19.768217 1156 1288 D ConnectivityService: sending notification AVAILABLE for NetworkRequest [ id=3, legacyType=-1, [] ]
09-06 14:20:19.768695 2234 2234 D ActivityThread: SVC-Calling onStartCommand: com.stv.stvpush.service.StvPushService@7cfaa18, flags=0, startId=2
```


Window size的确认以及修改

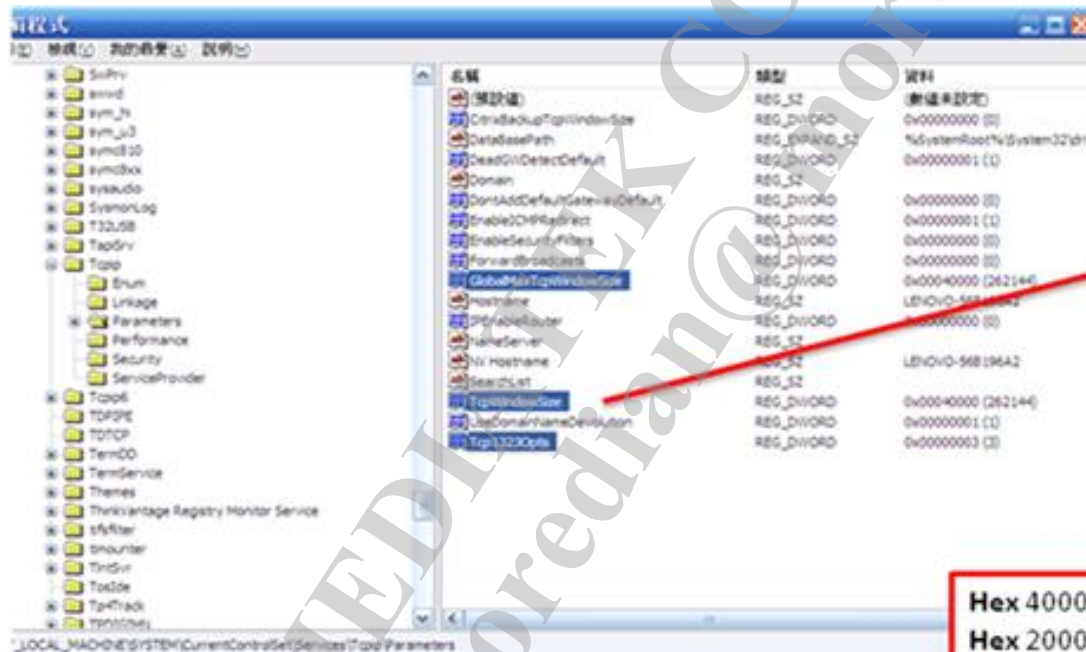
■ Window size修改(PC)

— PC Dos窗口中输入regedit

— 浏览至

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TCPIP\Parameters

注册表子键，在Parameters子键下创建或修改名为TCPWindowSize的REG_DWORD值，将该值设置为0X40000



Hex 40000 as 256K
Hex 20000 as 128K
Hex 10000 as 64K

Window size的确认以及修改

■ Window size修改(mobile)

— alps/system/core/rooddir/init.rc (Sw version before L branch)

```
# Define TCP buffer sizes for various networks
# ReadMin, ReadInitial, ReadMax, WriteMin, WriteInitial, WriteMax,
setprop net.tcp.buffer.size.default 4096,87380,110208,4096,16384,110208
setprop net.tcp.buffer.size.wifi 524288,1048576,2097152,262144,524288,1048576
setprop net.tcp.buffer.size.lte 524288,1048576,2097152,262144,524288,1048576
setprop net.tcp.buffer.size.umts 4094,87380,110208,4096,16384,110208
setprop net.tcp.buffer.size.hspa 4094,87380,262144,4096,16384,262144
setprop net.tcp.buffer.size.hsupa 4094,87380,262144,4096,16384,262144
setprop net.tcp.buffer.size.hsdpa 4094,87380,262144,4096,16384,262144
setprop net.tcp.buffer.size.hspap 4094,87380,1220608,4096,16384,1220608
setprop net.tcp.buffer.size.edge 4093,26280,35040,4096,16384,35040
setprop net.tcp.buffer.size.gprs 4092,8760,11680,4096,8760,11680
setprop net.tcp.buffer.size.evdo 4094,87380,262144,4096,16384,262144
```

— alps/frameworks/opt/telephony/src/java/com/android/internal/telephony/dataconnection/DataConnection.java(Sw version from L0 branch)

```
private static final String TCP_BUFFER_SIZES_GPRS = "4092,8760,11680,4096,8760,11680";
private static final String TCP_BUFFER_SIZES_EDGE = "4093,26280,35040,4096,16384,35040";
private static final String TCP_BUFFER_SIZES_UMTS = "4094,87380,524288,4096,16384,524288";
private static final String TCP_BUFFER_SIZES_1XRTT = "16384,32768,131072,4096,16384,102400";
private static final String TCP_BUFFER_SIZES_EVDO = "4094,87380,524288,4096,16384,524288";
private static final String TCP_BUFFER_SIZES_EHRPD = "131072,262144,1048576,4096,16384,524288";
private static final String TCP_BUFFER_SIZES_HSDPA = "4094,87380,524288,4096,16384,524288";
private static final String TCP_BUFFER_SIZES_HSPA = "4094,87380,524288,4096,16384,524288";
private static final String TCP_BUFFER_SIZES_LTE =
    "524288,1048576,2097152,262144,524288,1048576";
private static final String TCP_BUFFER_SIZES_HSPAP = "4094,87380,1220608,4096,16384,1220608";
```

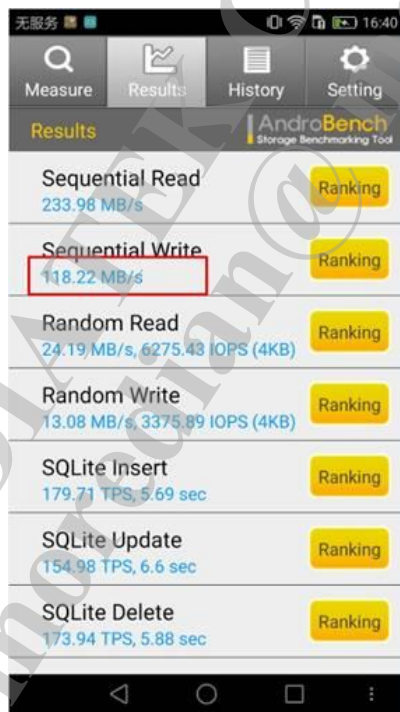
Window size的确认以及修改

■ Window size修改(mobile)

- 实时修改方法，需要有root权限的手机，只针对当次开机有效
 - adb shell
 - echo 2097152 4194304 8388608 > /proc/sys/net/ipv4/tcp_rmem
 - echo 2097152 4194304 8388608 > /proc/sys/net/ipv4/tcp_wmem
 - adb shell “cat /proc/sys/net/ipv4/tcp_rmem”
 - adb shell “cat /proc/sys/net/ipv4/tcp_wmem”

文件系统性能的确证

- 提供测试机使用的emmc物料型号和datasheet;
- 网络上下载androbenc.apk做一下storage测试, 并把测试结果截图发给MTK;
- MTK根据 androbenc, 以及emmc datasheet综合判定andftp测试指标是否正常;
- 一般emmc performance在datasheet中可以透过performance关键字查询到:



如何对比测试并抓取相关LOG

如何对比测试并抓取相关log

■ 初始条件:

1. 确认测试手机RF和antenna均是OK的
2. 安装了正确的驱动程序\
3. 选择相同category的参考手机。
4. APN, QOS, auth type等设置均相同
5. 多线程测试(FTP测试)
6. 如果case允许同时进行对比测试: 同一时间, 同一地点, 同一运营商的SIM卡, 用相同的软件, 同一个文件在同一个server(PC的操作系统需相同)。
7. 如果case不允许同时进行对比测试: 同一地点, 同一张SIM卡, 同一个PC, 用相同的软件, 同一个文件在同一个server, 并且交替测试。(测试机测试一次, 换参考机进行测试一次, 如此反复)。
8. 如果是tethering的话, 请修改PC的TCP window size并且重启PC。
 - [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters]"TcpWindowSize" 为Hex 80000
9. 设置成data prefer, Data prefer的设置方法: 工程模式->Telephony->勾上 "Mobile data service preferred"->开机重启测试

■ 测试步骤:

1. 确认在同一个频点和小区。
2. 测试手机和参考机同时/交替进行测试十次, 记录相应的速率以及时间。
3. 提供参考机和测试机的从开机到关机的完整log, 包括netlog, mobile log和modem log以及database以及PC侧的wireshark log。(如果参考机是其他平台的, 请至少提供netlog/PC wiresharklog)

如何对比测试并抓取相关log

■ 补充:

1. 限制PC侧的wireshark log大小: 以免PC 的loading太重以及log太大,请将packet大小设置一下: capture->option->双击需要capture的interface->limit each packet to [] byte->设置为128
2. limit手机侧的TCP package大小: 工程模式-> MTKlogger--> settings --> networkLog -->enable package limitation>选择Limited package size为128;
3. 让modem log无限大:MTK logger->Modem log->Limit log size->设置为storage支持的最大size。
4. 如何确认4G小区: 工程模式->telephony->LTE information->4G active intra-RAT meas (LTE), 有[serving info] 可以看到EARFCN和PCI 以及RSRP(需要除以4)和 RSRP(需要除以4)
5. 每测试一轮就保存一份log给我们, 而不要测试很多轮才保存一份log。
6. 如果您测试的是usb tethering 功能, 请务必在出现问题后, 先点击netlog 的stop 按钮, 再关闭usb tethering .

Prevent lost trace in Ethereal(WireShark)

The image shows the 'Wireshark: Capture Options' dialog box with three numbered annotations:

- 1** Double click the correct interface: Points to the 'MT6290' interface in the 'Capture' table.
- 2** Points to the 'Edit Interface Settings' sub-dialog box.
- 3** Do not list of packets in real time to avoid high PC loading: Points to the 'Update list of packets in real time' checkbox in the 'Display Options' section.

Capture Table:

Capture	Interface	Link-layer header	Prom. Mode	Snapshot [B]	Buffer [MB]	Capture Filter
<input type="checkbox"/>	Adapter for generic dialup and ...	Ethernet	enabled	default	2	
<input type="checkbox"/>	無線網路連線	Ethernet	enabled	default	2	
<input type="checkbox"/>	Local Area Connection 2	Ethernet	enabled	default	2	
<input checked="" type="checkbox"/>	MT6290 10.61.7.82	Ethernet	enabled	default	2	

Edit Interface Settings (MT6290):

- Interface: MT6290
- IP address: 10.61.7.82
- Link-layer header type: Ethernet
- ☒ Capture packets in promiscuous mode
- ☒ Limit each packet to 128 bytes
- Buffer size: 64 megabyte(s)
- Capture Filter: [Empty]
- Buttons: Help, OK, Cancel

Display Options:

- ☐ Update list of packets in real time
- ☒ Hide capture info dialog
- Name Resolution:**
 - ☒ Resolve MAC addresses
 - ☐ Resolve network-layer names
 - ☒ Resolve transport-layer name
 - ☒ Use external network name resolver

Capture Files:

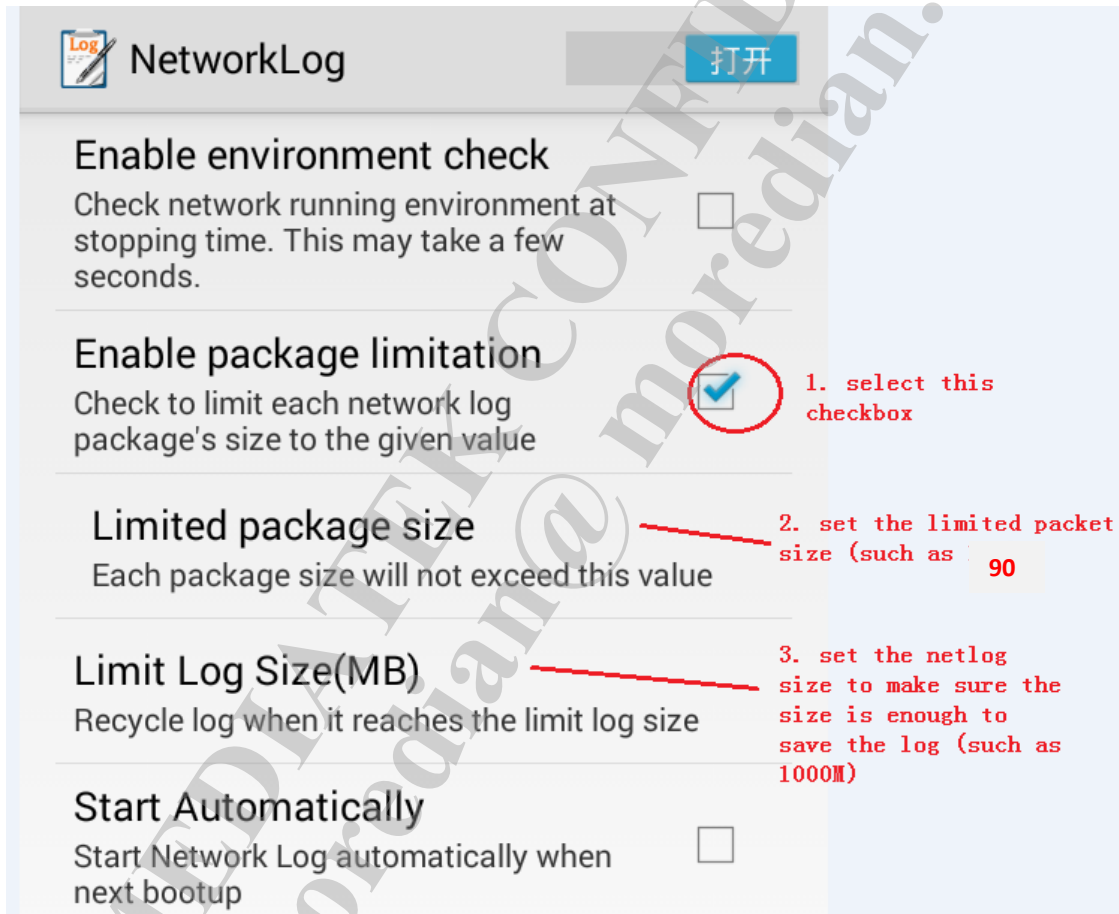
- File: [Empty] Browse...
- ☐ Use multiple files
- ☒ Use pcap-ng format
- ☒ Next file every 1 mebibyte(s)
- ☐ Next file every 1 minute(s)
- ☐ Ring buffer with 2 files
- ☐ Stop capture after 1 file(s)
- Stop Capture Automatically After...
 - ☐ 1 packet(s)
 - ☐ 1 mebibyte(s)
 - ☐ 1 minute(s)

Buttons: Help, Start, Close

enable the "Limit each packet to 128 bytes"

Prevent lost trace in netlog(Netlog)

Latest netlog tool also support limit packet size function



The screenshot shows the 'NetworkLog' configuration window. It has a title bar with a 'Log' icon and a '打开' (Open) button. The window contains several settings:

- Enable environment check**: Check network running environment at stopping time. This may take a few seconds. ☐
- Enable package limitation**: Check to limit each network log package's size to the given value. ☒ (Annotated with '1. select this checkbox')
- Limited package size**: Each package size will not exceed this value. (Annotated with '2. set the limited packet size (such as 90)')
- Limit Log Size(MB)**: Recycle log when it reaches the limit log size. (Annotated with '3. set the netlog size to make sure the size is enough to save the log (such as 1000M)')
- Start Automatically**: Start Network Log automatically when next bootup. ☐

How to dump Packets on Reference Phone

- Get Tcpcdump Log

- Root reference phone
- adb push ./tcpdump /data/local/tmp/
- adb shell chmod 777 /data/local/tmp/tcpdump
- adb shell
 - cd /data/local/tmp/
 - ./tcpdump -l any -s 0 -w ./tcpdump.cap



Tcpdump

How to dump Packets on Reference Phone

- Get iptables Log

- 如果您测试的是usb tethering功能，需要run这个命令，请务必先Run这个命令，再关闭usb tethering
- Root reference phone
- double click iptablesdump.bat



iptablesDump.rar

PS:tcpdump切包、合并、分割文件

- 将源tcpdump文件copy到wireshark安装目录
- Dos窗口切到wireshark安装目录
- 切包: `editcap -s 128 源文件 目标文件`
 - `editcap -s 128 tcpdump_NTLog_2016_0727_142548_start.cap
dut_cut.cap`
- 合并: `mergcap -w 目标文件 源文件1 源文件2 源文件3`
 - `mergcap -w real_dut.pcap tcpdump_NTLog_2016_0811_145026_start_2.cap tcpdump_NTLog_2016_0811_145026_start_1.cap
tcpdump_NTLog_2016_0811_145026_start.cap`
- 分割: `editcap -c 目标文件大小 源文件 目标文件`

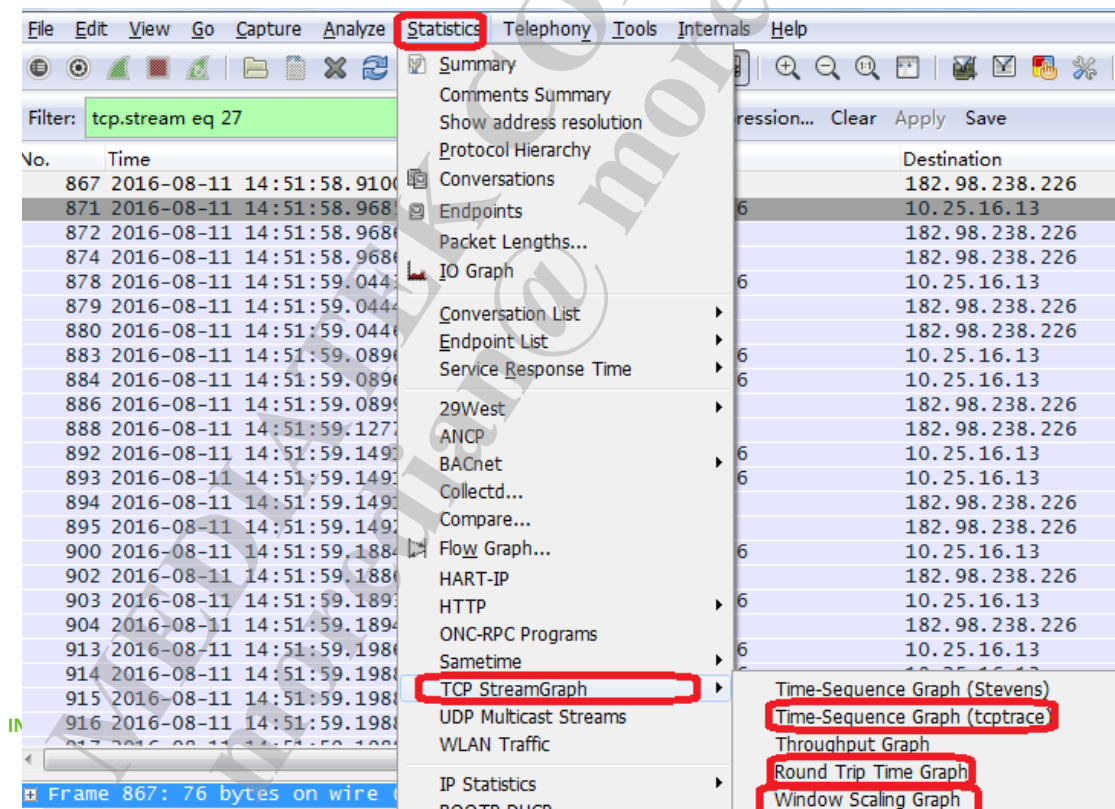
PS:找上传下载的tcp stream流

- 打开tcpdump文件
- 进入Statistics->Conversations菜单，找到传输数据量大的条目
 - bytes A->B大的是上传（红框蓝色背景）
 - Bytes B->A大的是下载(红框无背景)

Ethernet Fibre Channel FDDI IPv4: 12 IPv6 IPX JXTA NCP RSVP SCTP TCP: 20 Token Ring UDP: 3 USB WLAN													
TCP Conversations													
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A→B	Bytes A→B	Packets A←B	Bytes A←B	Rel Start	Duration	bps A→B	bps A←B
10.155.89.85	59150	119.145.88.100	80	4	304	4	304	0	0	0.000000000	7.0015	547.30	N/A
10.155.89.85	52537	203.208.39.197	80	38	12 634	22	6 520	16	6 114	1.695453000	0.8934	58386.25	54750.54
10.155.89.85	52199	203.208.43.121	80	2	1 100	2	1 100	0	0	5.241358000	41.0200	214.53	N/A
10.155.89.85	50882	119.145.88.166	80	18	3 433	11	1 512	7	1 921	14.092632000	2.3799	5082.50	6457.34
10.155.89.85	47031	203.208.39.205	80	9	3 178	5	1 754	4	1 424	14.259546000	2.7890	5031.13	4084.57
10.155.89.85	58365	112.96.28.182	80	27	3 881	19	2 068	8	1 813	14.765336000	92.2060	179.42	157.30
10.155.89.85	44266	203.208.43.121	80	7	1 268	4	754	3	514	14.794091000	4.5657	1321.15	900.62
10.155.89.85	53443	202.175.95.42	80	17	3 285	10	1 460	7	1 825	15.684150000	5.4472	2144.21	2680.26
10.155.89.85	40636	202.75.250.35	80	18	3 309	10	1 496	8	1 813	16.194900000	16.4959	725.51	879.25
10.155.89.85	53817	119.147.52.35	80	19	3 453	11	1 516	8	1 937	16.777122000	6.1387	1975.68	2524.33
10.155.89.85	54523	119.145.88.166	80	2 578	2 079 470	1 191	97 887	1 387	1 981 583	19.094854000	18.5840	42138.17	353027.28
10.155.89.85	39958	119.145.88.166	80	1 771	1 710 578	1 175	1 669 820	596	40 758	37.949740000	16.1279	828289.74	20217.41
10.155.89.85	35340	183.61.122.70	443	4	255	2	112	2	143	43.575768000	0.0355	25206.07	32182.74

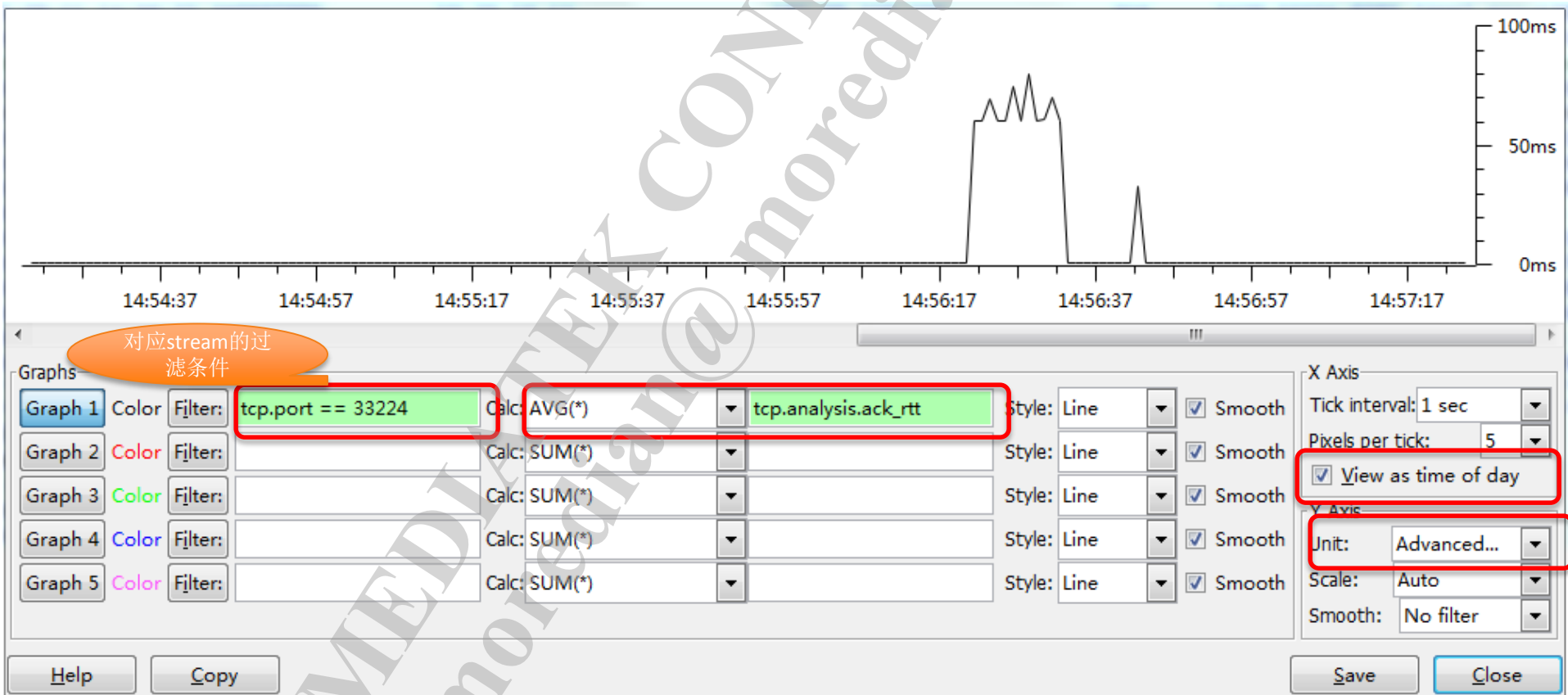
PS:如何显示其tcp trace、 window scaling、 RTT 、

- 找到对应tcp stream并follow stream
- Time-sequence graph: 1.点选网络给手机的数据包 2.Statistics->tcp streamgraphy→time-sequence graph
- window scaling/rtt 图:1.点选手机给网络的ACK包 2.Statistics->tcp streamgraphy→ round trip time graphy/window scaling graph



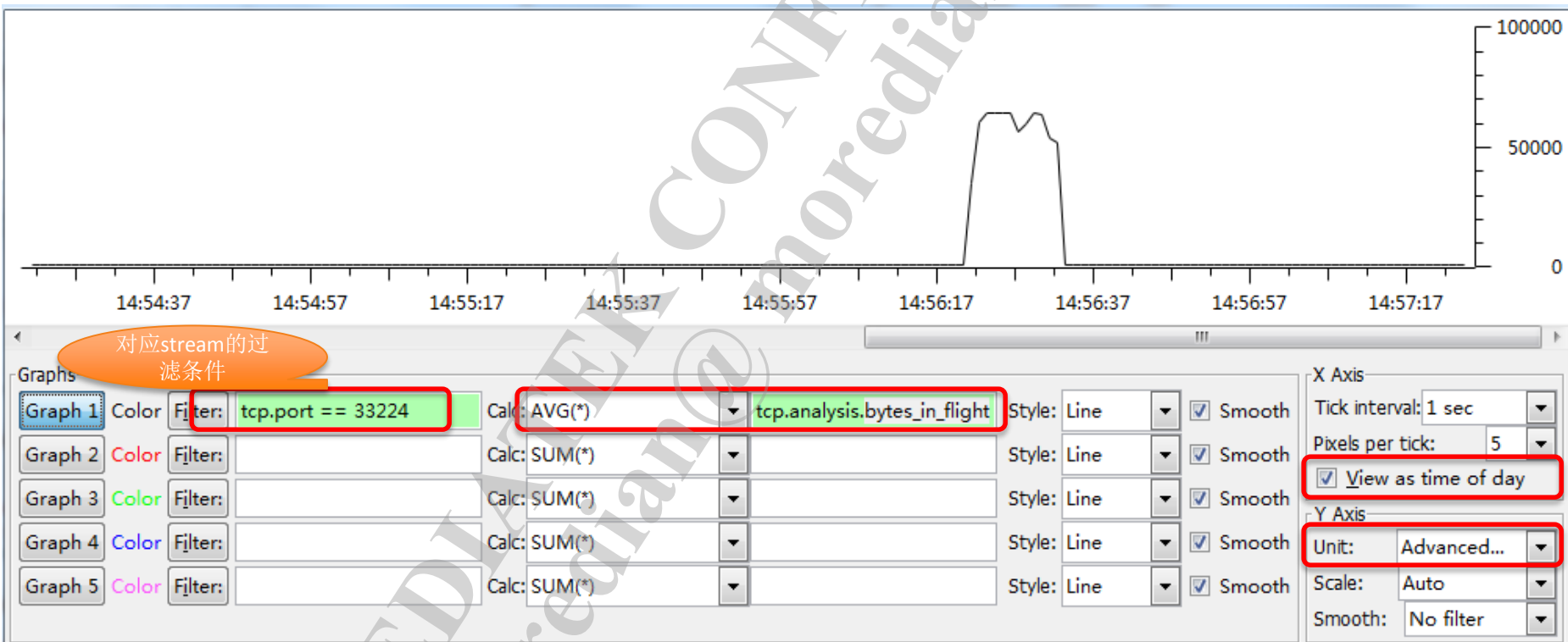
PS:IO Grap如何显示RTT

- 找到下到tcp stream并follow stream
- Statistics->IO Graph
- 如下图输入对应的过滤条件进行过滤



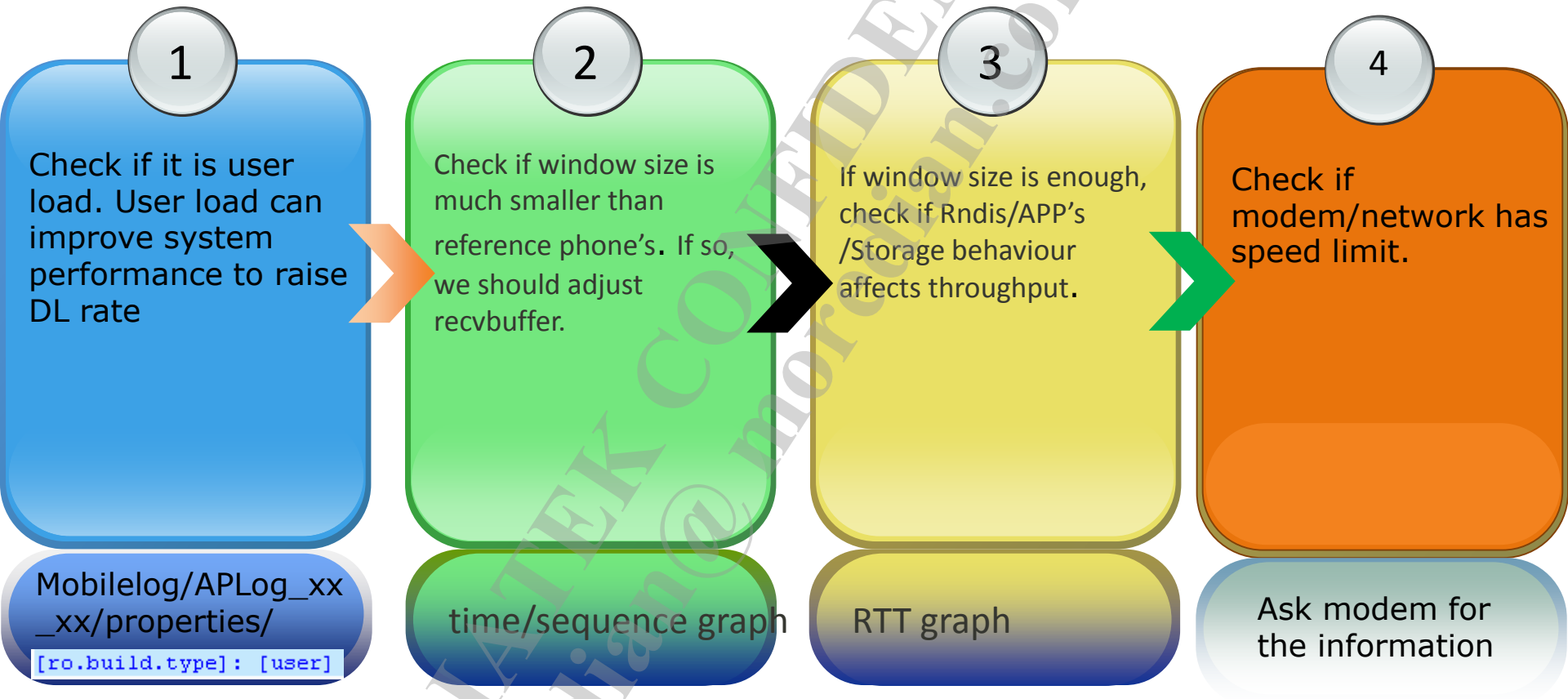
PS:IO Grap如何显示bytes_in_flight

- 找到下到tcp stream并follow stream
- Statistics->IO Graph
- 如下图输入对应的过滤条件进行过滤



Throuhgput test issue 分析步骤和实例分析

DL rate SOP



Eng load导致下载速率降低

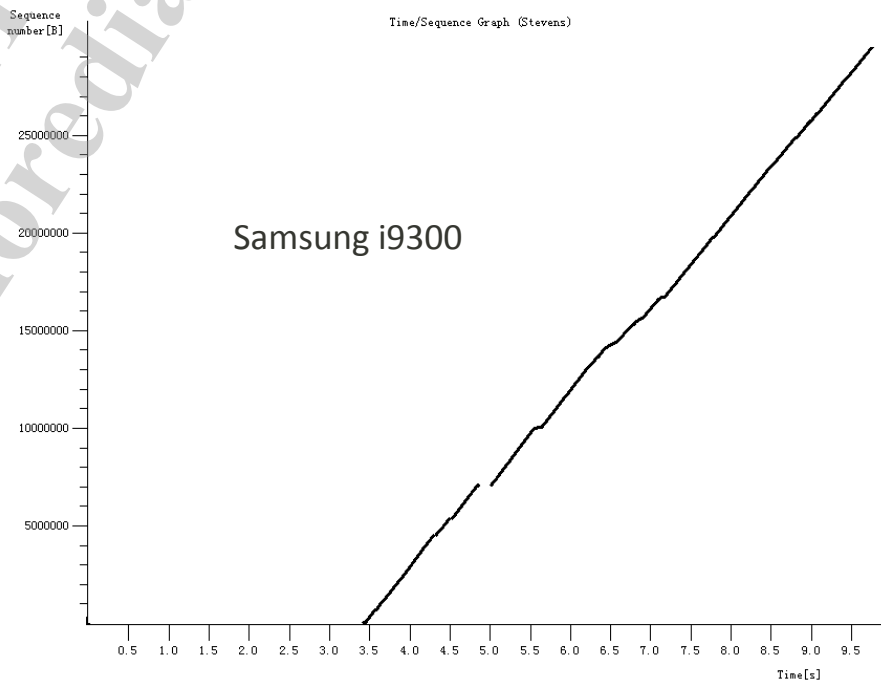
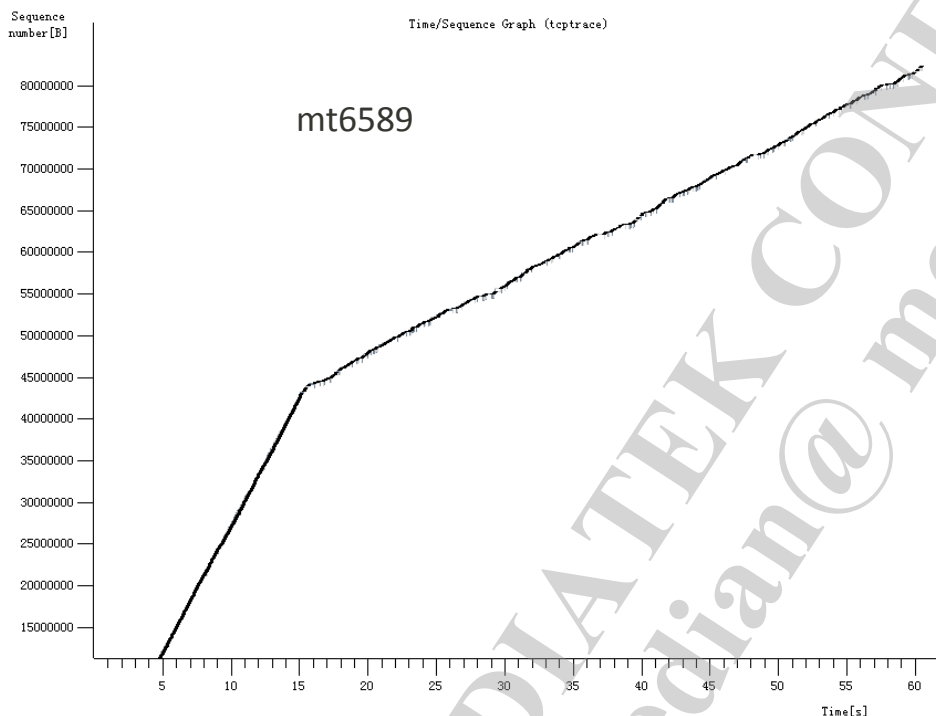
- Test APK
 - AndFTP.apk
- Test Item
 - APK Download Rate
- Debug Tool
 - Wireshark
- Test result

DL rate (Mbps)	对比机	测试机	测试条件(wifi)
AndFTP.apk	35.08	16.67	几乎同时
lperf.apk	40.97	47.39	几乎同时

FTP Download Rate

■ Analysis

- 从time/sequence图可看出，mt6589一开始速率比较快，发生window update后，速率下降。对比机一直速率比较稳定。



36761	2012-12-17 17:52:42.246761	192.168.100.101	192.168.100.103	68	TCP	[TCP ACKed lost segment] 49097 > ricardo-lm [ACK] Seq=1 Ack=44298529 Win=4
36762	2012-12-17 17:52:42.249022	192.168.100.101	192.168.100.103	68	TCP	[TCP Window Update] 49097 > ricardo-lm [ACK] Seq=1 Ack=44298529 Win=8576
36763	2012-12-17 17:52:42.252741	192.168.100.101	192.168.100.103	68	TCP	[TCP Window Update] 49097 > ricardo-lm [ACK] Seq=1 Ack=44298529 Win=17152

Eng load导致下载速率降低

■ Analysis

- 测试机 window size最大为522880Bytes，对比机window size最大为785024Bytes，window size对应的是rcv buffer，两者的window size不是差别很大。

测试机

192.168.100.101	192.168.100.103	68	TCP	49097 > ricardo-lm [ACK] Seq=1 Ack=8806441 Win=522880 [TCP CHECKSUM INCORRECT
192.168.100.101	192.168.100.103	68	TCP	49097 > ricardo-lm [ACK] Seq=1 Ack=8812233 Win=522880 [TCP CHECKSUM INCORRECT
192.168.100.101	192.168.100.103	68	TCP	49097 > ricardo-lm [ACK] Seq=1 Ack=8895145 Win=522880 [TCP CHECKSUM INCORRECT
192.168.100.101	192.168.100.103	68	TCP	49097 > ricardo-lm [ACK] Seq=1 Ack=8900937 Win=522880 [TCP CHECKSUM INCORRECT
192.168.100.101	192.168.100.103	68	TCP	49097 > ricardo-lm [ACK] Seq=1 Ack=8903833 Win=522880 [TCP CHECKSUM INCORRECT
192.168.100.101	192.168.100.103	68	TCP	49097 > ricardo-lm [ACK] Seq=1 Ack=8906729 Win=522880 [TCP CHECKSUM INCORRECT
192.168.100.101	192.168.100.103	68	TCP	49097 > ricardo-lm [ACK] Seq=1 Ack=8909625 Win=522880 [TCP CHECKSUM INCORRECT
192.168.100.101	192.168.100.103	68	TCP	49097 > ricardo-lm [ACK] Seq=1 Ack=8912521 Win=522880 [TCP CHECKSUM INCORRECT

对比机

192.168.100.102	192.168.100.100	90	TCP	59973 > de-noc [ACK] Seq=1 Ack=15879985 Win=785024 Len=0 TSV=2439446 TSER=22388
192.168.100.102	192.168.100.100	90	TCP	59973 > de-noc [ACK] Seq=1 Ack=15884329 Win=785024 Len=0 TSV=2439446 TSER=22388
192.168.100.102	192.168.100.100	90	TCP	59973 > de-noc [ACK] Seq=1 Ack=15887225 Win=785024 Len=0 TSV=2439446 TSER=22388
192.168.100.102	192.168.100.100	90	TCP	59973 > de-noc [ACK] Seq=1 Ack=15890121 Win=785024 Len=0 TSV=2439446 TSER=22388
192.168.100.102	192.168.100.100	90	TCP	59973 > de-noc [ACK] Seq=1 Ack=15892481 Win=785024 Len=0 TSV=2439446 TSER=22388
192.168.100.102	192.168.100.100	90	TCP	59973 > de-noc [ACK] Seq=1 Ack=15969601 Win=785024 Len=0 TSV=2439449 TSER=22388
192.168.100.102	192.168.100.100	90	TCP	59973 > de-noc [ACK] Seq=1 Ack=15976841 Win=785024 Len=0 TSV=2439449 TSER=22388
192.168.100.102	192.168.100.100	90	TCP	59973 > de-noc [ACK] Seq=1 Ack=15979737 Win=785024 Len=0 TSV=2439450 TSER=22388

Eng load导致下载速率降低

■ Analysis

- 对比测试机和对比机，测试机有比较多的window full和window update，可知，测试机系统 read速率比较慢，就是说从网络中收到包后，没有及时地去读取数据。

测试机

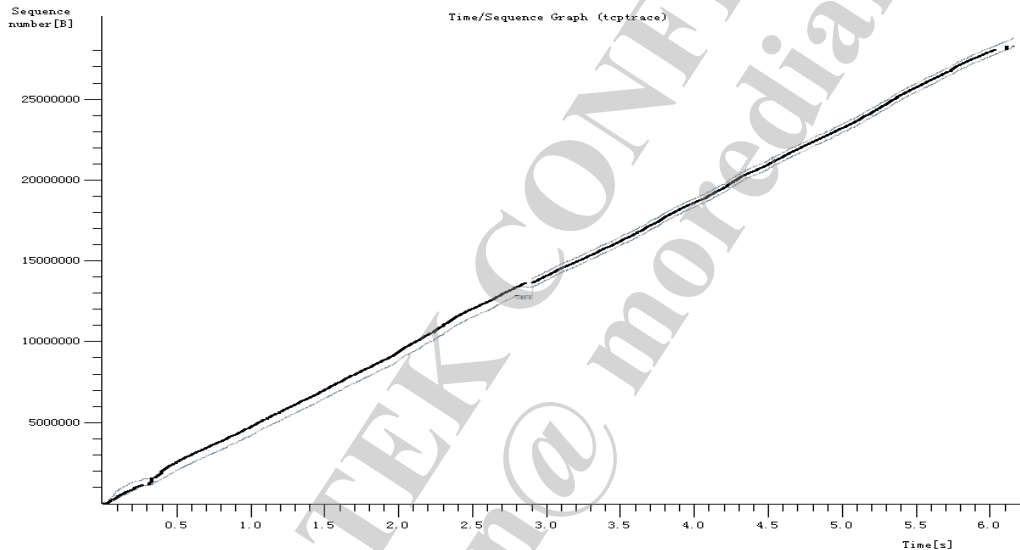
192.168.100.101	192.168.100.103	68 TCP	[TCP Window Update] 49097 > ricardo-lm [ACK] Seq=1 Ack=1605793 Win=9216
192.168.100.101	192.168.100.103	68 TCP	[TCP Window Update] 49097 > ricardo-lm [ACK] Seq=1 Ack=1615929 Win=7040
192.168.100.101	192.168.100.103	68 TCP	[TCP Window Update] 49097 > ricardo-lm [ACK] Seq=1 Ack=1633305 Win=4928
192.168.100.101	192.168.100.103	68 TCP	[TCP Window Update] 49097 > ricardo-lm [ACK] Seq=1 Ack=1633305 Win=13504
192.168.100.101	192.168.100.103	68 TCP	[TCP Window Update] 49097 > ricardo-lm [ACK] Seq=1 Ack=1656473 Win=9216
192.168.100.101	192.168.100.103	68 TCP	[TCP Window Update] 49097 > ricardo-lm [ACK] Seq=1 Ack=1668057 Win=4928
192.168.100.101	192.168.100.103	68 TCP	[TCP Window Update] 49097 > ricardo-lm [ACK] Seq=1 Ack=1689777 Win=4928
192.168.100.101	192.168.100.103	68 TCP	[TCP Window Update] 49097 > ricardo-lm [ACK] Seq=1 Ack=1710049 Win=9216

192.168.100.103	192.168.100.101	1516 FTP-DATA	[TCP Window Full] FTP Data: 1448 bytes
192.168.100.103	192.168.100.101	1516 FTP-DATA	[TCP Window Full] FTP Data: 1448 bytes
192.168.100.103	192.168.100.101	1516 FTP-DATA	[TCP Window Full] FTP Data: 1448 bytes
192.168.100.103	192.168.100.101	1516 FTP-DATA	[TCP Window Full] FTP Data: 1448 bytes
192.168.100.103	192.168.100.101	1516 FTP-DATA	[TCP Window Full] FTP Data: 1448 bytes
192.168.100.103	192.168.100.101	1516 FTP-DATA	[TCP Window Full] FTP Data: 1448 bytes
192.168.100.103	192.168.100.101	1516 FTP-DATA	[TCP Window Full] FTP Data: 1448 bytes
192.168.100.103	192.168.100.101	1516 FTP-DATA	[TCP Window Full] FTP Data: 1448 bytes
192.168.100.103	192.168.100.101	1516 FTP-DATA	[TCP Window Full] FTP Data: 1448 bytes
192.168.100.103	192.168.100.101	1516 FTP-DATA	[TCP Window Full] FTP Data: 1448 bytes
192.168.100.103	192.168.100.101	1516 FTP-DATA	[TCP Window Full] FTP Data: 1448 bytes

Eng load导致下载速率降低

■ Analysis

- 测试机烧成user load后，FTP DL rate上升为39.237Mbps，速率稳定，没有window update和window full包。



■ Solution

- Eng load相对user load添加较多log，影响系统performance，进而影响read速率

Rndis driver 性能不达标导致下载速率降低

- Test APK
 - FTP Test over USB tethering
- Test Item
 - 终端业务的上下行并发速率
- Debug Tool
 - Wireshark
- Test result
 - 下行速率不达标

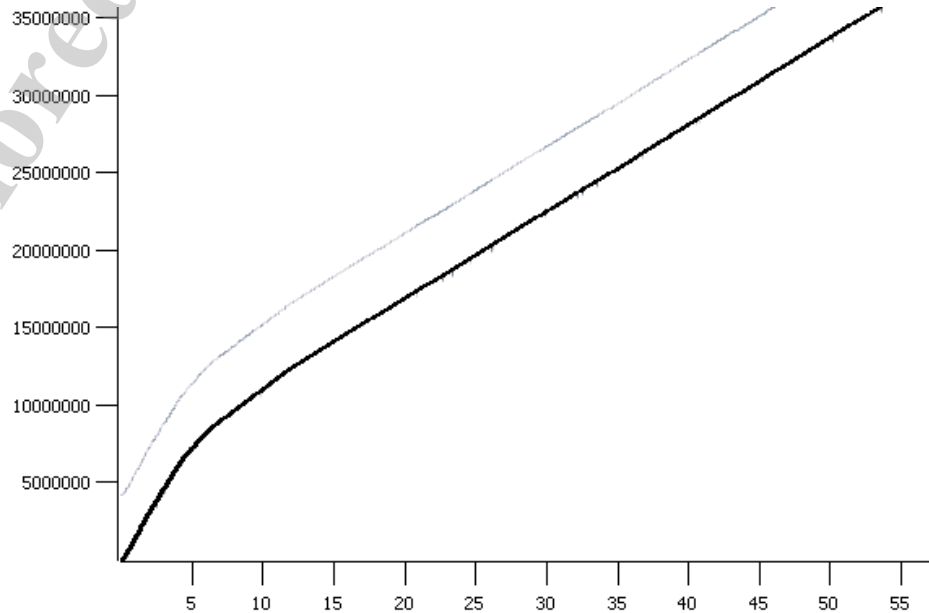
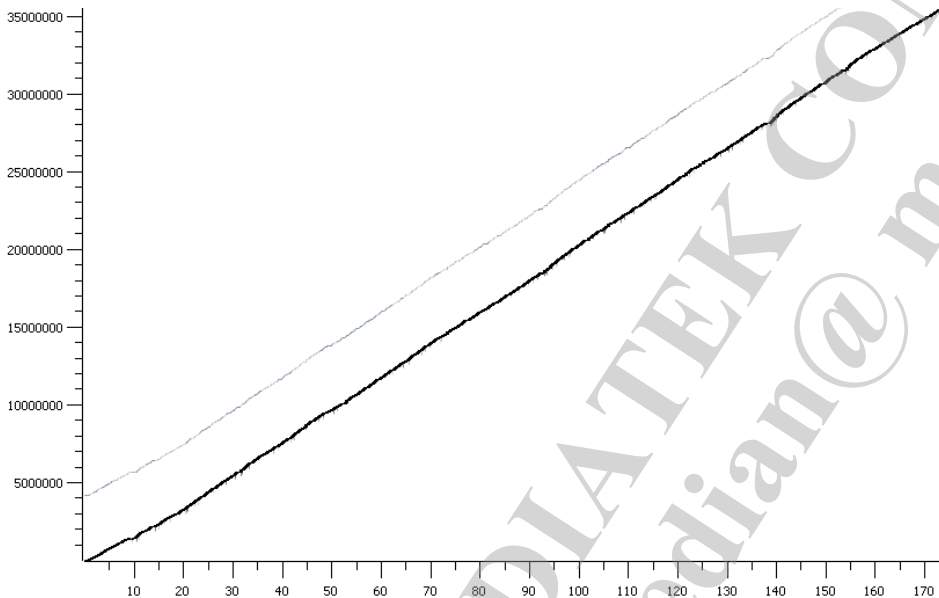
Rndis driver 性能不达标导致下载速率降低

■ Analysis

- 从time/sequence图可看出，测试机对比机一直速率比较稳定，window size也足够的，但是明显可以看出，对比机服务器给包的速率比测试机快。

测试机

对比机

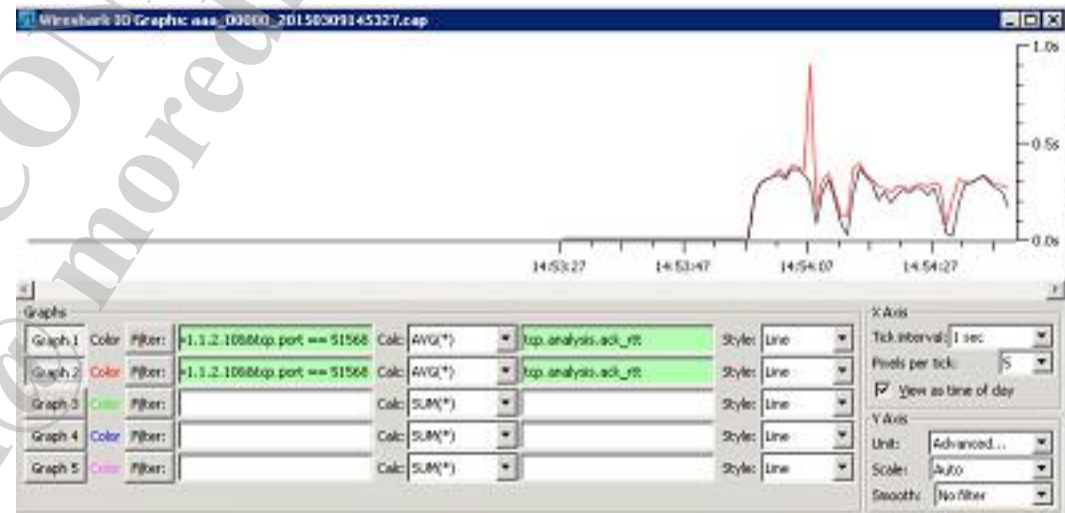
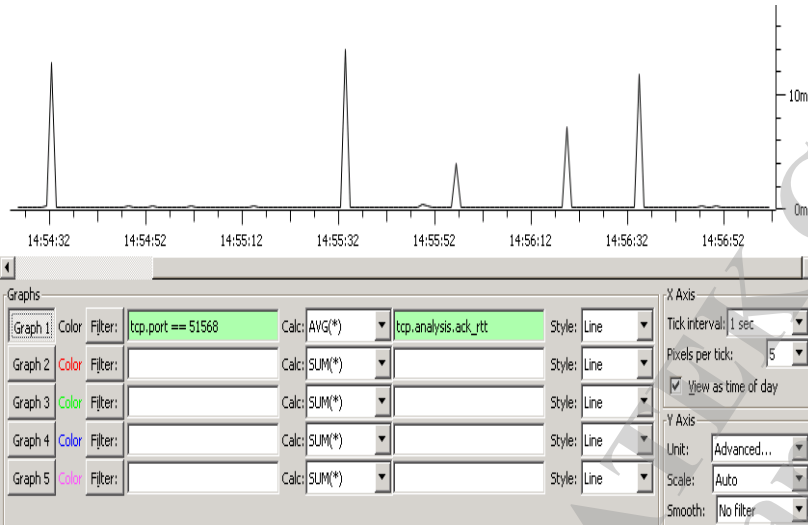


Rndis driver 性能不达标导致下载速率降低

■ Analysis

- 截取pc上rndis interface, mobile rndis interface上经过包的RTT图来看, 可以看到测试机RTT时间在经过rndis后明显增大。

测试机



■ Solution

经rndis同事确认, 当前客户有打开rndis debug开关, 导致其大量吐log 拉低了Rndis性能, 关闭该开关后解决问题。

App版本不一样导致其速率不一样

- Test APK
 - MobileMarket v3.1_p33 vs MobileMarket v3.1_AndroidJT
- Test Item
 - APK Download Rate
- Debug Tool
 - Wireshark

App版本不一样导致其速率不一样

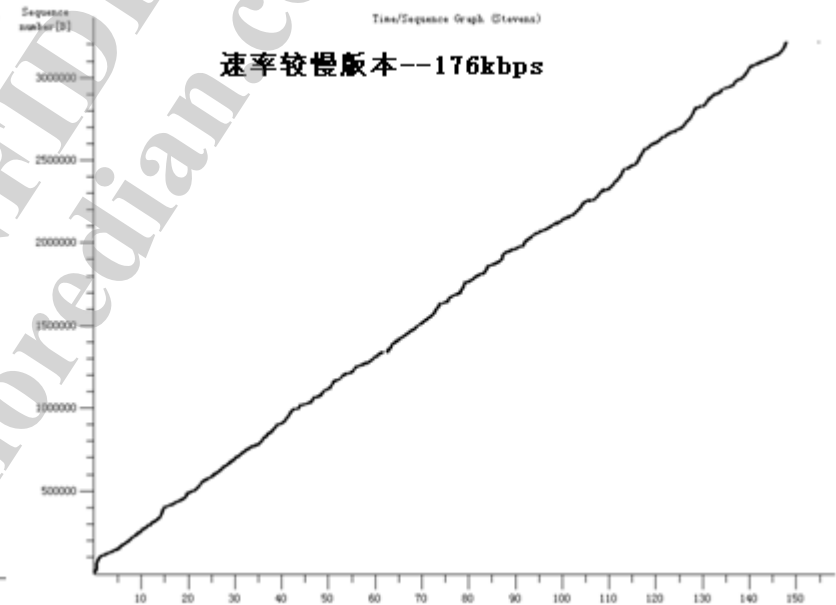
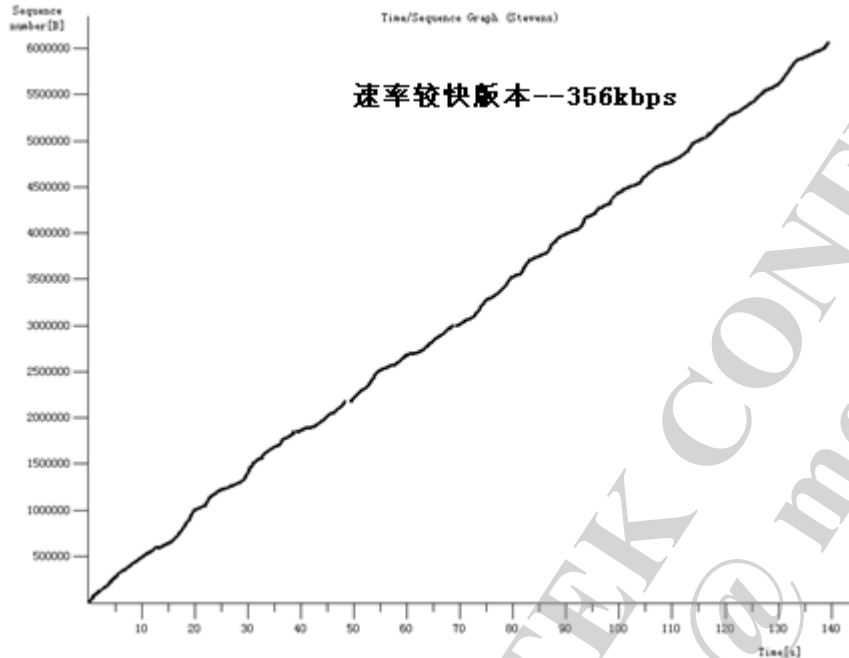
■ Test result

- 同一版本在不同load上下载速率是一样的
- 两个版本在同一load上下载速率不一，一个将近180kbps，一个将近370kbps

	ICS2. MP		ICS2. TDD. FPB		测试条件
MM3.1_p33	Tcp. stream=60	179kbps	Tcp. stream=55	179kbps	几乎同时
	Tcp. stream=64	179kbps	Tcp. stream=61	179kbps	几乎同时
	Tcp. stream=73	179kbps	Tcp. stream=69	179kbps	2个load间隔几分钟，download
	Tcp. stream=38	176kbps	Tcp. stream=48	169kbps	2个load间隔几分钟，download
MM3.1_ CTAndroid_ JT	Tcp. stream=10	358kbps	Tcp. stream=4	370kbps	几乎同时
	Tcp. stream=22	364kbps	Tcp. stream=13	361kbps	2个load间隔几分钟，download

App版本不一样导致其速率不一样

■ Analysis

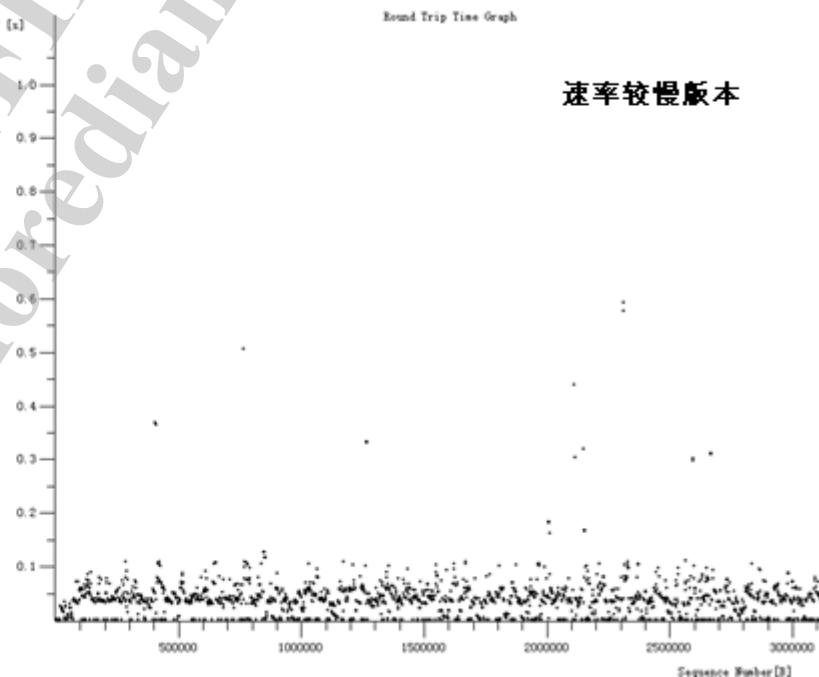
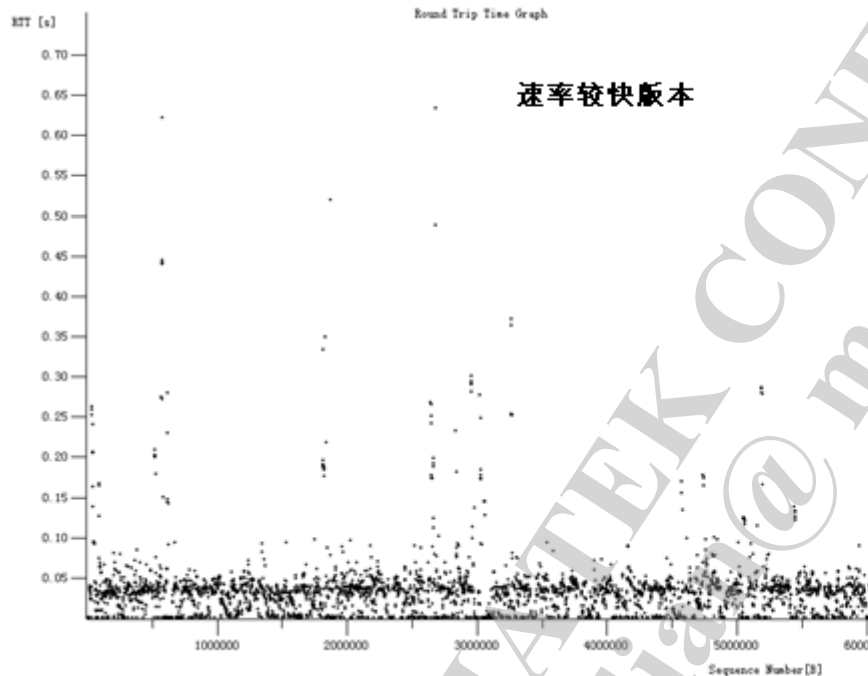


- 从tcpdump中看出两个版本速率都比较稳定。

App版本不一样导致其速率不一样

- Analysis

- 2个版本RTT都在正常范围内

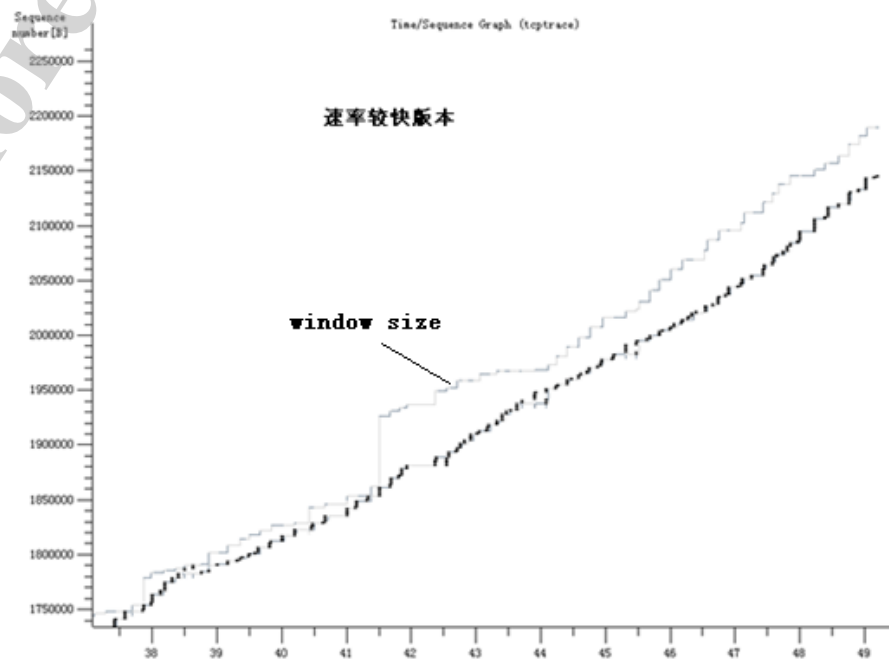
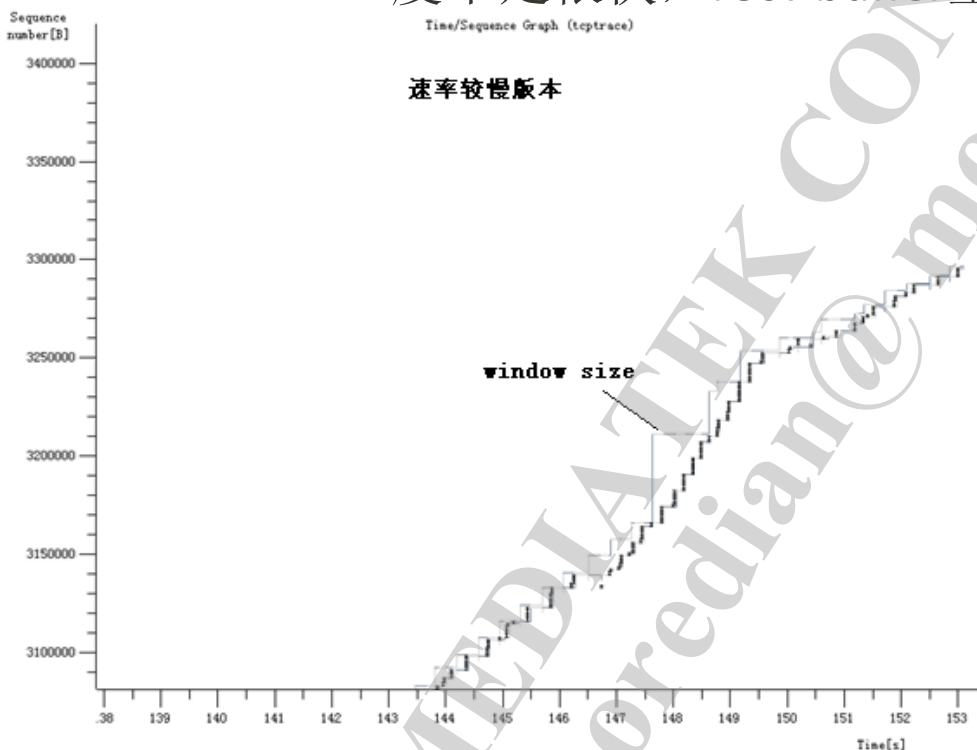


App版本不一样导致其速率不一样

■ Analysis

— 速率下降的可能原因：

- 比较较快版本和较慢版本，发现较慢版本window size相对比较小，可能是mobile接收到数据包后，APP read/write处理速度不是很快，recv buffer里面数据一直没被取走，所以



App版本不一样导致其速率不一样

- 关于APP部分，read/write添加log:
 - App 调用 read 读取网络数据，会直接 call `IoBridge.recvfrom()`(libcore/luni/src/main/java/net/PlainSocketImpl.java)

```
System.out.println("(MMdebug..)");
```

```
int readCount = IoBridge.recvfrom(true, fd, buffer, offset, byteCount, 0, null, false);
```

```
System.out.println("(MMdebug:rc => " + offset + "byteCount => " + byteCount + "readCount => " + readCount
```

- 慢版本read 2次间隔差不多为0.3s，从网络包read一次基本不到1ms

慢版本 AP Log:

```
09-19 13:56:25.260 1907 2136 | System.out: (MMdebug..  
09-19 13:56:25.391 1907 2136 | System.out: (MMdebug:rc=>0byteCount => 8192readCount => 1380 +0.13s  
09-19 13:56:25.927 1907 2136 | System.out: (MMdebug.. +0.536s  
09-19 13:56:25.927 1907 2136 | System.out: (MMdebug:rc=>0byteCount => 8192readCount => 8192 +0s  
09-19 13:56:26.294 1907 2136 | System.out: (MMdebug.. +0.367s  
09-19 13:56:26.294 1907 2136 | System.out: (MMdebug:rc=>0byteCount => 8192readCount => 8192 +0s  
09-19 13:56:26.670 1907 2136 | System.out: (MMdebug.. +0.376s  
09-19 13:56:26.670 1907 2136 | System.out: (MMdebug:rc=>0byteCount => 8192readCount => 8192 +0s  
09-19 13:56:27.031 1907 2136 | System.out: (MMdebug.. +0.361s  
09-19 13:56:27.032 1907 2136 | System.out: (MMdebug:rc=>0byteCount => 8192readCount => 8192 +0.001s  
09-19 13:56:27.390 1907 2136 | System.out: (MMdebug.. +0.362s  
09-19 13:56:27.390 1907 2136 | System.out: (MMdebug:rc=>0byteCount => 8192readCount => 8192 +0s  
09-19 13:56:27.804 1907 2136 | System.out: (MMdebug.. +0.414s  
09-19 13:56:27.804 1907 2136 | System.out: (MMdebug:rc=>0byteCount => 8192readCount => 8192 +0s  
09-19 13:56:28.163 1907 2136 | System.out: (MMdebug.. +0.359s  
09-19 13:56:28.163 1907 2136 | System.out: (MMdebug:rc=>0byteCount => 8192readCount => 8192 +0s
```


App版本不一样导致其速率不一样

■ Analysis

- 快版本read 2次间隔为0.1~0.2s，从网络包read一次基本不到1ms

快版本log:

```
09-19 14:08:40.875 3099 3299 | System.out: (MMdebug..  
09-19 14:08:41.022 3099 3299 | System.out: (MMdebug:rc=>0byteCount=> 8192readCount=> 2760      +0.147s  
09-19 14:08:41.361 3099 3299 | System.out: (MMdebug..                                +0.339s  
09-19 14:08:41.361 3099 3299 | System.out: (MMdebug:rc=>0byteCount=> 8192readCount=> 5520      +0s  
09-19 14:08:41.552 3099 3299 | System.out: (MMdebug..                                +0.191s  
09-19 14:08:41.552 3099 3299 | System.out: (MMdebug:rc=>0byteCount=> 8192readCount=> 8192      +0s  
09-19 14:08:41.725 3099 3299 | System.out: (MMdebug..                                +0.173s  
09-19 14:08:41.726 3099 3299 | System.out: (MMdebug:rc=>0byteCount=> 8192readCount=> 8192      +0s  
09-19 14:08:41.912 3099 3299 | System.out: (MMdebug..                                +0.186s  
09-19 14:08:41.912 3099 3299 | System.out: (MMdebug:rc=>0byteCount=> 8192readCount=> 8192      +0s  
09-19 14:08:42.152 3099 3299 | System.out: (MMdebug..                                +0.24s  
09-19 14:08:42.153 3099 3299 | System.out: (MMdebug:rc=>0byteCount=> 8192readCount=> 8192      +0.001s  
09-19 14:08:42.351 3099 3299 | System.out: (MMdebug..                                +0.199s  
09-19 14:08:42.352 3099 3299 | System.out: (MMdebug:rc=>0byteCount=> 8192readCount=> 8192      +0.001s  
09-19 14:08:42.617 3099 3299 | System.out: (MMdebug..                                +0.265s  
09-19 14:08:42.617 3099 3299 | System.out: (MMdebug:rc=>0byteCount=> 8192readCount=> 8192      +0s  
09-19 14:08:42.808 3099 3299 | System.out: (MMdebug..                                +0.191s  
09-19 14:08:42.808 3099 3299 | System.out: (MMdebug:rc=>0byteCount=> 8192readCount=> 8192
```

- 所以，对比2个版本的log，发现APP行为不同，thread调用状况不同，导致速率不同，mobile read包的速率以及网络状况并不是影响download下载速率的直接原因。

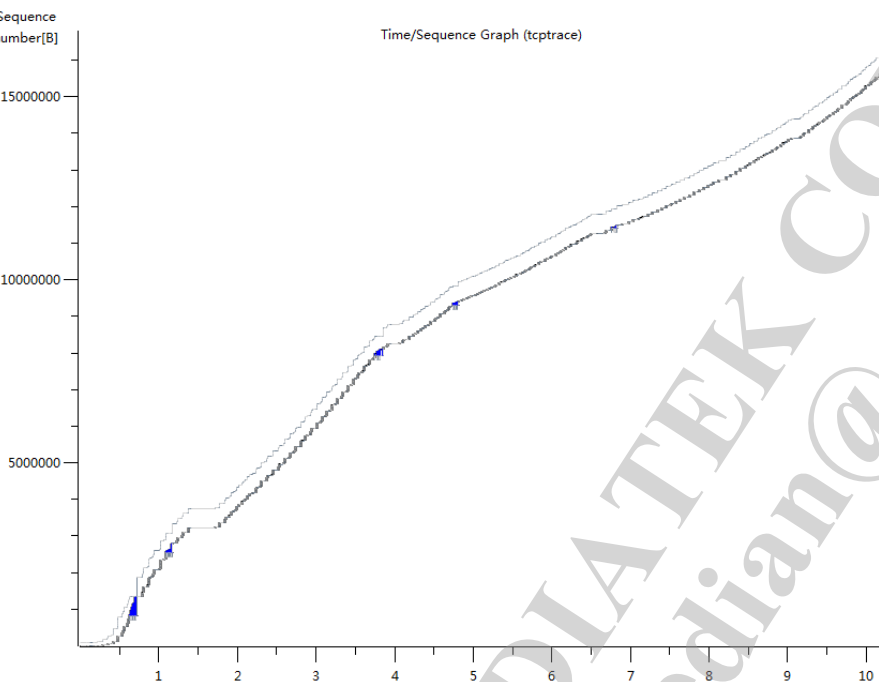
Window size & 线程数导致其速率不一样

- Test APK
 - Speedtest.apk
- Test Item
 - APK Download Rate
- Debug Tool
 - Wireshark

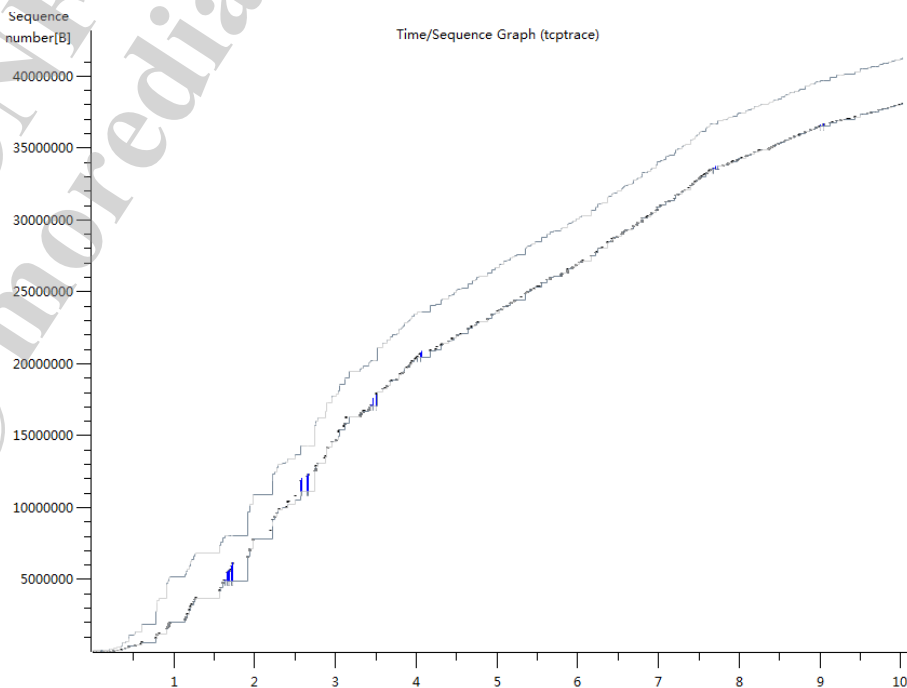
Window size & 线程数导致其速率不一样

- 查看对比机和测试机tcp trace
 - 发现两者服务器给包都很顺畅，但是对比机会比测试机快很多

测试机

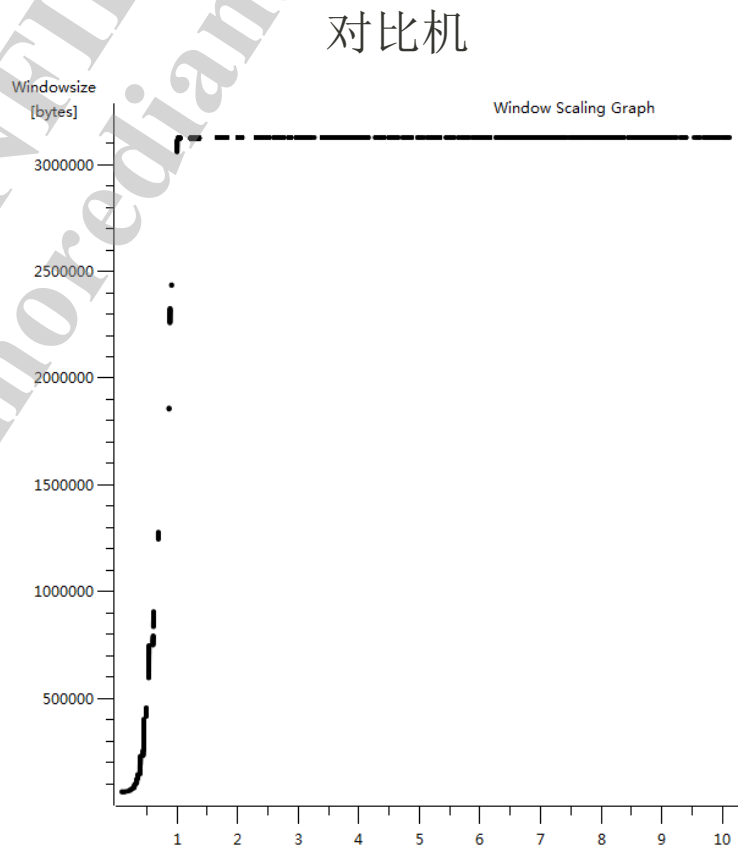
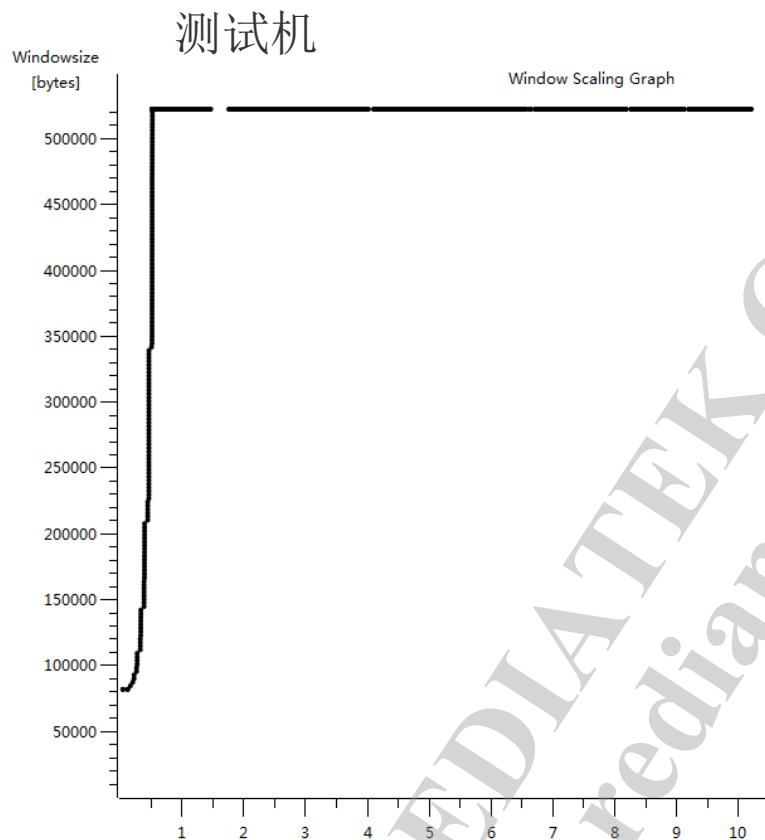


对比机



Window size & 线程数导致其速率不一样

- 查看对比机和测试机window size
 - 发觉对比机是测试机的6倍



Window size & 线程数导致其速率不一样

- 查看tcpbuffer: 对比机readbuffer是测试机的4倍

//测试机

— 08-11 14:50:37.249521 1149 1258 D

ConnectivityService:TcpBufferSizes:

524288,1048576,2097152,262144,524288,1048576

//对比机:

— 08-13 11:28:25.497 D/ConnectivityService(1099):TcpBufferSizes:
2097152,4194304,8388608,262144,524288,1048576}}

- DataConnection.java文件中, 修改 private static final String
TCP_BUFFER_SIZES_LTE ="
2097152,4194304,8388608,262144,524288,1048576 ";

Window size & 线程数导致其速率不一样

- 查看两手机下载线程数，发觉测试机只有2条，对比机有4条

— 测试机

10.25.16.13	33195	182.98.238.226	8080	22	1 785	12	959	10	826	78.723411000	0.5346	14352.07	12361.64
10.25.16.13	33196	182.98.238.226	8080	17 802	16 871 943	6 348	443 063	11 454	16 428 880	81.300285000	10.2650	345299.08	12803770.78
10.25.16.13	33197	182.98.238.226	8080	22 798	21 819 979	7 976	551 899	14 822	21 268 080	81.311601000	10.2354	431365.76	16623189.04
10.25.16.13	57511	174.140.137.37	443	24	7 893	15	2 627	9	5 266	89.840256000	244.3235	86.02	172.43
10.25.16.13	55579	117.27.245.62	443	44	12 113	20	2 843	24	9 270	91.744147000	60.3778	376.69	1228.27
10.25.16.13	54387	117.27.245.62	80	144	96 853	72	4 630	72	92 223	91.754813000	30.1748	1227.51	24450.33
10.25.16.13	60461	117.27.245.62	80	68	40 124	34	2 290	34	37 834	91.756937000	30.1926	606.77	10024.70
10.25.16.13	54010	174.140.137.33	80	41	24 177	21	3 045	20	21 132	91.879748000	300.8498	80.97	561.93
10.25.16.13	33198	182.98.238.226	8080	10 740	10 473 156	7 129	10 226 472	3 611	246 684	92.704401000	20.6550	3960863.78	95544.36
10.25.16.13	33199	182.98.238.226	8080	10 257	9 979 814	6 792	9 742 542	3 465	237 272	92.705575000	20.7214	3761348.17	91604.70
10.25.16.13	56289	110.232.178.82	443	26	6 600	15	2 429	11	4 171	104.424550000	1.8933	10263.78	17624.63

— 对比机

10.186.54.3	33324	203.208.39.217	80	9	1 852	5	1 262	4	590	51.291019000	15.4265	654.46	305.97
10.186.54.3	41412	182.98.238.226	8080	22	1 785	12	959	10	826	53.176942000	0.3599	21314.25	18358.26
10.186.54.3	41413	182.98.238.226	8080	10 745	10 246 995	3 685	255 391	7 060	9 991 604	55.599695000	10.2247	199823.54	7817651.07
10.186.54.3	41414	182.98.238.226	8080	12 181	11 608 519	4 187	294 539	7 994	11 313 980	55.600754000	10.1266	232684.44	8937991.49
10.186.54.3	41415	182.98.238.226	8080	10 265	9 269 775	3 911	277 539	6 354	8 992 236	55.603696000	10.1285	219215.21	7102550.85
10.186.54.3	41416	182.98.238.226	8080	7 036	6 719 651	2 402	166 971	4 634	6 552 680	55.609015000	10.2144	130772.55	5132092.94
10.186.54.3	46835	106.122.248.169	443	23	10 116	13	2 409	10	7 707	66.167560000	0.3747	51433.83	164549.83
10.186.54.3	39182	106.122.248.169	80	115	70 955	51	3 159	64	67 796	66.169312000	0.6569	38472.08	825657.79
10.186.54.3	53182	106.122.248.169	80	659	453 919	323	18 401	336	435 518	66.179662000	1.4590	100895.81	2388019.27
10.186.54.3	41417	182.98.238.226	8080	2 864	2 773 739	1 918	2 709 175	946	64 564	67.097248000	5.4866	3950236.65	94140.50
10.186.54.3	41418	182.98.238.226	8080	2 153	2 059 887	1 424	2 009 671	729	50 216	67.101761000	5.4860	2930618.11	73227.87
10.186.54.3	41419	182.98.238.226	8080	3 606	3 447 823	2 380	3 363 367	1 226	84 456	67.106560000	5.4784	4911420.27	123328.47
10.186.54.3	41420	182.98.238.226	8080	1 966	1 836 003	1 267	1 787 359	699	48 644	67.120440000	5.4522	2622603.93	71375.67
10.186.54.3	41440	203.208.40.40	80	9	6 891	8	6 815	1	76	69.030551000	0.4170	130753.13	N/A

Window size & 线程数导致其速率不一样

- 对于其上传下载线程不一致问题，让客户在/external/okhttp/okhttp/src/main/java/com/squareup/okhttp/internal/http/HttpEngine.java中sendRequest()中加上log打印发给网络的URL

```
Request request = networkRequest(userRequest);
```

改成：

```
Request request = networkRequest(userRequest);
```

```
System.out.println(request.urlString());
```

Window size & 线程数导致其速率不一样

- 查看log发现传给服务器的参数CT为-1，此参数代表前手机的网络类型（通过调用getNetworkType()获得），speed服务器用此参数来决定其允许客户端上传下载的的线程数。
 - CT=-1,代表其网络类型unknown，而实际此时其网络应该是LTE。
08-13 11:29:32.574325 4633 4706 I System.out:
<https://www.speedtest.net/api/android/config.php?ct=-1&dct=0&brand=Meizu....>
 - 而MTK内部的手机，在LTE情况下，发给网络的值是CT=15
<https://www.speedtest.net/api/android/config.php?ct=15&dct=0&....>
- 最后查log发现，在doNotifyDataConnection的时候，会发networktype=139（LTEA），正常MTK平台默认会将LTEA<E综合成LTE发出来。
 - 08-13 11:34:26.933765 1865 1865 D DefaultPhoneNotifier:
doNotifyDataConnection apnType=default,networkType=139,
state=CONNECTED

Window size &线程数导致其速率不一样

- 请客户还原这部分修改后，再测试此问题通过.

Modem分配资源较少导致下载速率降低

- Test APK
 - Speedtest.apk
- Test Item
 - 终端业务的下载速率
- Debug Tool
 - Wireshark
- Test result
 - 下行速率对比机低

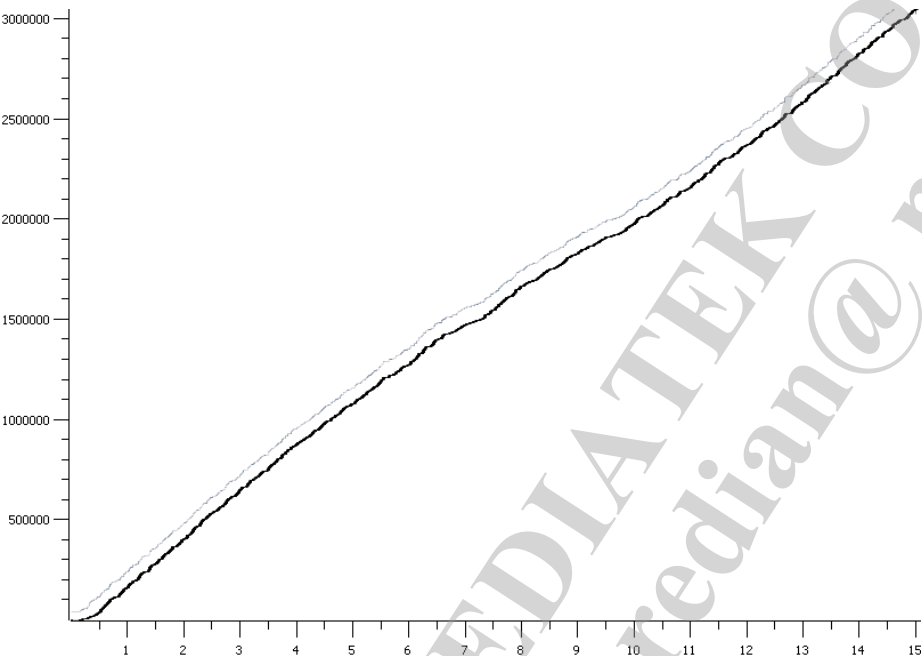
DL rate (Mbps)	M81	HTC E9	测试条件(GPRS,CMWAP)
Speedtest.apk	215.7KBps	1003.2KBps	交替测试

Modem分配资源不足导致下载速率降低

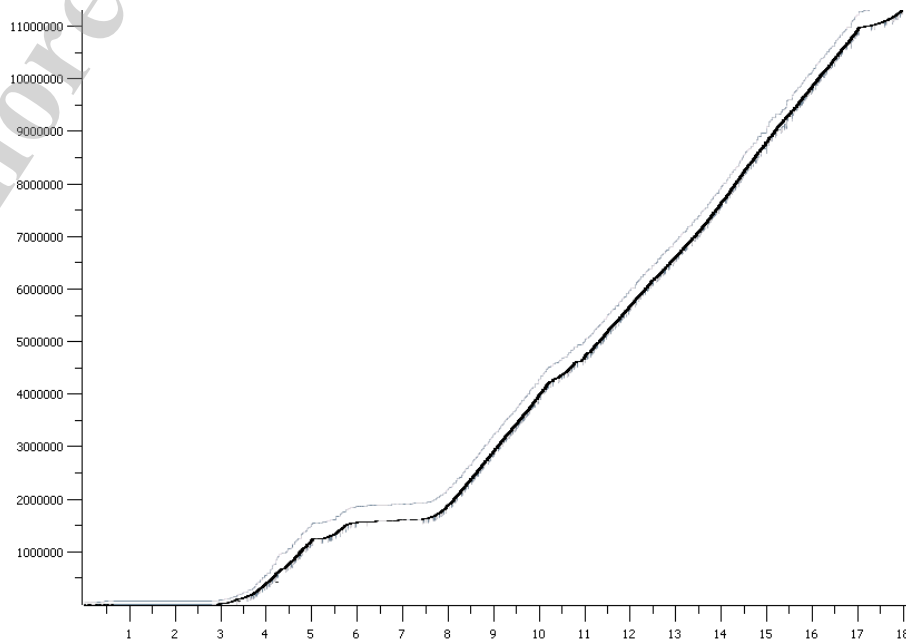
■ Analysis

- 从time/sequence图可看出，测试机一直速率比较稳定，window size也足够的，但是明显可以看出，对比机最开始其服务器给包跟对比机速度差不多，一段时间后速率明显比测试机快。

M81



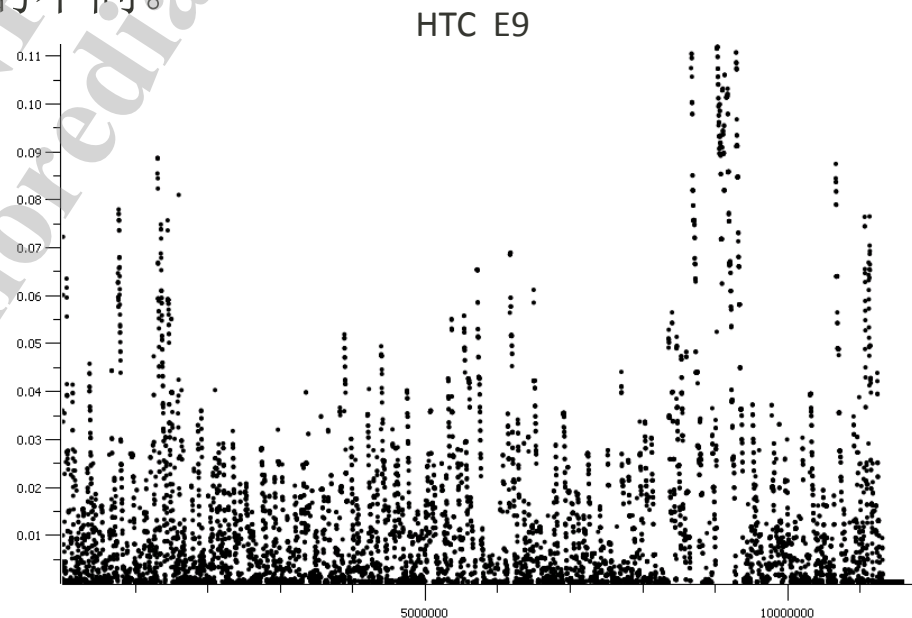
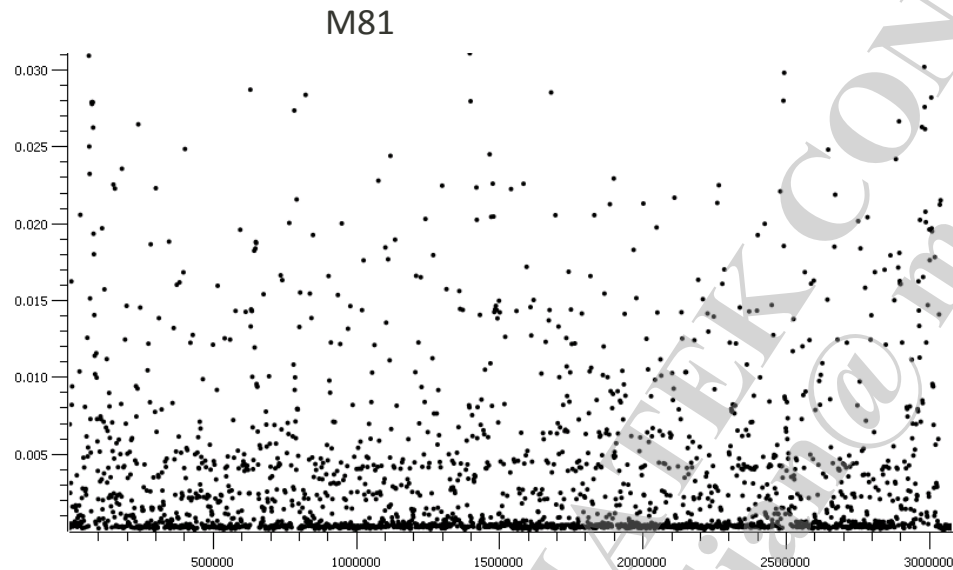
HTC E9



Modem分配资源不足导致下载速率降低

■ Analysis

- 从rtt图可看出，测试机的rtt是明显对比机低的，综合以上来看，当前两手从Ap来看都是正常的，极有可能是modem和network那边的速率不一样导致其速度的不同。



■ Solution

- 经modem同事确认，此问题两手机确实注册在同一个cell，但是network有时分配给测试机的TB size有时候会比对机小，因此拉低了整个手机测试的下行速率。

UL rate SOP

1

Check if it is user load. User load can improve system performance to raise UL rate .

Mobilelog/APLog_xx_xx/properties/

[ro.build.type]: [user]

2

Check if bytes_in_flight is much smaller than reference phone's. If so, we should enlarge sendbuffer.

IO Graph

3

Check if APP's temporary write buffer is smaller. If there are too many not-full-sized segments, it is a sign that write buffer is smaller..

4

Check bandwidth. If NW assigns mobile a lower bandwidth, RTT will increase, slow the rate..

Ask modem for the information

Environment

- Test Item
 - Http Upload Rate by browser
- Test equipment
 - MX8960（类似基站，发射信号）
- Test software version
 - ALPS.GB.FDD2.MP(android 2.3)
- Test steps
 - Upload a file（nearly >5 Mb）by mobile browser, summary file rate by wireshark
- Debug Tool
 - Wireshark

Http upload rate

- Test result

- 在sony实验室，zte73v2 upload rate为1.2Mbps（3G数据连接），对比机（sony urushi）upload rate为2.8Mbps

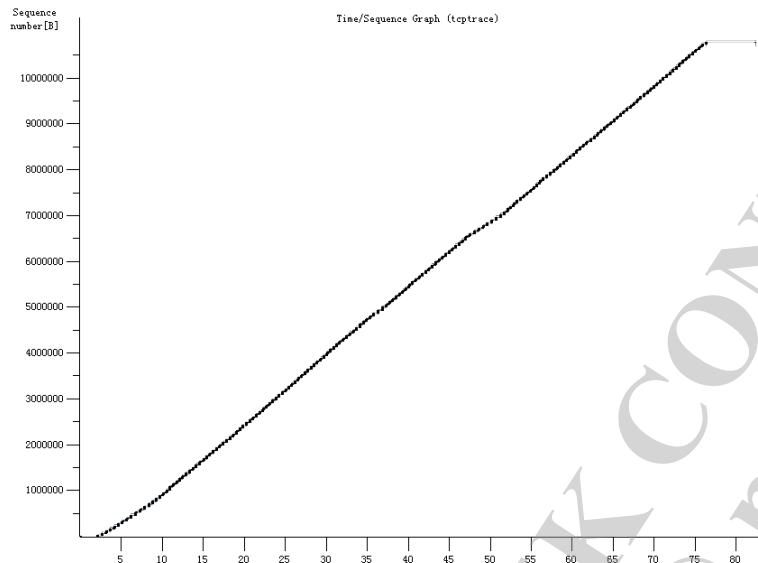
HSPA_0035_PERF HTTP upload over HSUPA throughput performance		
	DUT Alcatel OT903 SW: 010 04	BENCHMARK Sony Urushi SW: 4.1.B.0.431
	(Kbit/sec)	(Kbit/sec)
Attempt #1	1233	2697
Attempt #2	1215	2877
Attempt #3	1215	2877
Average	1221	2817
Deviation vs benchmark	43%	
KPI Status	FAILED	

- Test Purpose

- Zte73v2 upload rate is expected as 2.6Mbps(对比机的95%)

Sony实验室upload rate

■ Analysis



10.63.60.231	10.63.48.224	TCP	1440	57308 > vcom-tunnel [ACK] Seq=8010566 Ack=1 Wi
10.63.60.231	10.63.48.224	TCP	712	57308 > vcom-tunnel [PSH, ACK] Seq=8011954 Ack
10.63.60.231	10.63.48.224	TCP	1440	57308 > vcom-tunnel [ACK] Seq=8012614 Ack=1 Wi
10.63.60.231	10.63.48.224	TCP	712	57308 > vcom-tunnel [PSH, ACK] Seq=8014002 Ack
10.63.48.224	10.63.60.231	TCP	52	vcom-tunnel > 57308 [ACK] Seq=1 Ack=8007858 Wi
10.63.48.224	10.63.60.231	TCP	52	vcom-tunnel > 57308 [ACK] Seq=1 Ack=8009906 Wi
10.63.60.231	10.63.48.224	TCP	1440	57308 > vcom-tunnel [ACK] Seq=8014662 Ack=1 Wi
10.63.60.231	10.63.48.224	TCP	712	57308 > vcom-tunnel [PSH, ACK] Seq=8016050 Ack
10.63.60.231	10.63.48.224	TCP	1440	57308 > vcom-tunnel [ACK] Seq=8016710 Ack=1 Wi
10.63.60.231	10.63.48.224	TCP	712	57308 > vcom-tunnel [PSH, ACK] Seq=8018098 Ack

Sony实验室upload rate

■ Analysis

- 针对GB http upload trace发现，AP（external/apache-http/src/org/apache/http/entity/InputStreamEntity.java）调用write socket，每次只写入2k，由于write buffer速度比TCP/IP stack 发送速度慢，所以每次未组成full size包就被发送，于是增大AP write buffer为4k,在wifi实网测试，速率提升20%~30%

2K				4K			
1	Avg. bytes/sec	1162931.427		Avg. bytes/sec	1560950.430		
	Avg. MBit/sec	9.303		Avg. MBit/sec	12.488		
	Avg. packet size	750.621 bytes		Avg. packet size	942.024 bytes		
	Avg. packets/sec	1549.292		Avg. packets/sec	1657.018		
	Between first and last packet	10.172 sec		Between first and last packet	7.430 sec		
	Bytes	11829792		Bytes	11598196		
	Packets	15760	15760 0	Packets	12312	12312 0	
2	Traffic ▲ Captured ◀ Displayed ◀ Marked ◀			Traffic ▲ Captured ◀ Displayed ◀ Marked ◀			
	Avg. bytes/sec	1443534.059		Avg. bytes/sec	1633222.575		
	Avg. MBit/sec	11.548		Avg. MBit/sec	13.066		
	Avg. packet size	750.665 bytes		Avg. packet size	977.700 bytes		
	Avg. packets/sec	1923.008		Avg. packets/sec	1670.474		
	Between first and last packet	8.195 sec		Between first and last packet	7.079 sec		
	Bytes	11829724		Bytes	11562280		
	Packets	15759	15759 0	Packets	11826	11826 0	

Sony实验室upload rate

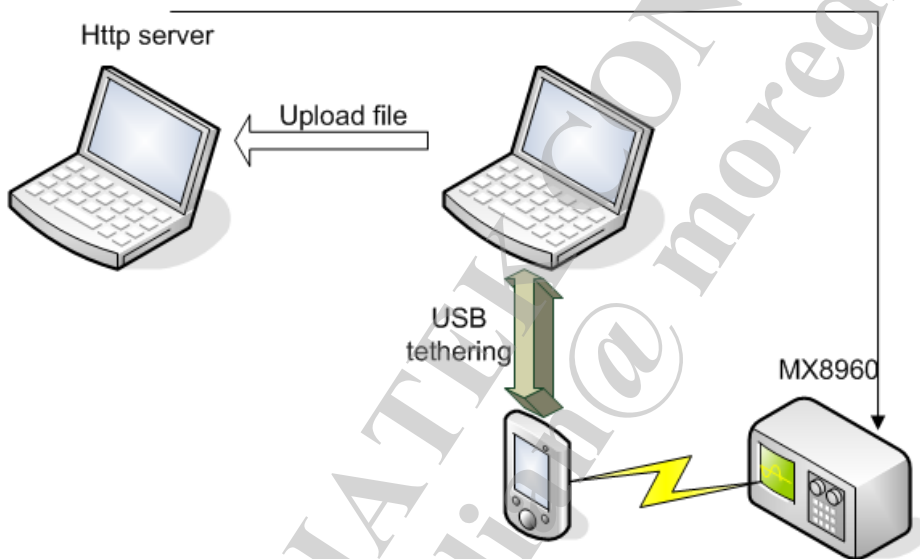
■ Analysis

- 在sony实验室测试修改了write buffer的load（由原来的2k变为8k），速率由1.2Mbps变为1.5Mbps，包的分组数量有减小，但是仍未达到2.8Mbps要求。

10.63.60.227	10.63.48.224	TCP	1440	[TCP segment of a reassembled PDU]
10.63.48.224	10.63.60.227	TCP	52	vcom-tunnel > 50019 [ACK] Seq=1 Ack=4744006
10.63.48.224	10.63.60.227	TCP	52	vcom-tunnel > 50019 [ACK] Seq=1 Ack=4746782
10.63.60.227	10.63.48.224	TCP	1440	[TCP segment of a reassembled PDU]
10.63.60.227	10.63.48.224	TCP	1440	[TCP segment of a reassembled PDU]
10.63.60.227	10.63.48.224	TCP	1440	[TCP segment of a reassembled PDU]
10.63.60.227	10.63.48.224	TCP	1440	[TCP segment of a reassembled PDU]
10.63.48.224	10.63.60.227	TCP	52	vcom-tunnel > 50019 [ACK] Seq=1 Ack=4749558
10.63.48.224	10.63.60.227	TCP	52	vcom-tunnel > 50019 [ACK] Seq=1 Ack=4752198
10.63.60.227	10.63.48.224	TCP	1304	[TCP segment of a reassembled PDU]

3G实验室upload rate

- 实验室搭建环境模拟sony实验室，使用usb tethering方式使用PC browser upload file来验证modem的转发能力以及是否browser AP或者mobile network stack导致速率慢
- 实验环境搭建如下：



3G实验室upload rate

■ Analysis

- 从实验结果来看，usb tethering时，PC browser upload file速率都能达到要求，但是与modem的极速（5.76Mbps）还是有差。所以多半是GB browser AP有导致速率慢的原因。

Rate (Mbps)	8960				
	1	2	3	4	5
Zte73v2 (GB.FDD2. MP)	3.114	2.975	3.703	3.808	3.562
6577(JB.MP)	2.693	3.82	2.894	3.771	2.524

3G实验室upload rate

■ Analysis

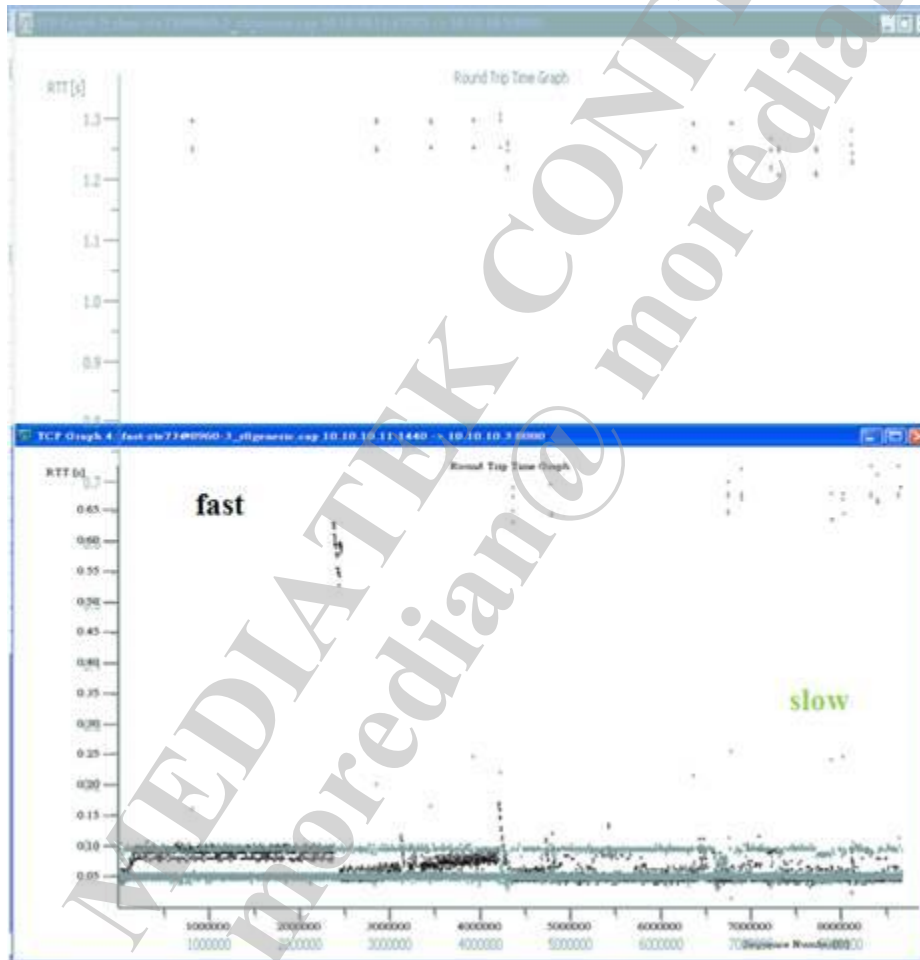
- 比较usb tethering upload file与mobile browser upload的tcpdump, 发现mobile接收server ACK比较慢（总是经过0.04s才可以继续发送new segments, 快的只经过0.01s）

0.000003	10.10.10.3		10.10.10.11	TCP	56 http-alt > 44079 [ACK] Seq=1 Ack=47695 Win=64380 Len=0
0.000363	10.10.10.11	慢	10.10.10.3	TCP	684 [TCP segment of a reassembled PDU]
0.000375	10.10.10.11		10.10.10.3	TCP	1476 [TCP segment of a reassembled PDU]
0.000264	10.10.10.11		10.10.10.3	TCP	684 [TCP segment of a reassembled PDU]
0.000209	10.10.10.11		10.10.10.3	TCP	1476 [TCP segment of a reassembled PDU]
0.000302	10.10.10.11		10.10.10.3	TCP	684 [TCP segment of a reassembled PDU]
0.000696	10.10.10.11		10.10.10.3	TCP	1476 [TCP segment of a reassembled PDU]
0.002641	10.10.10.11		10.10.10.3	TCP	1476 [TCP Out-Of-Order] [TCP segment of a reassembled PDU]
0.000623	10.10.10.11		10.10.10.3	TCP	684 [TCP segment of a reassembled PDU]
0.000341	10.10.10.11		10.10.10.3	TCP	1476 [TCP segment of a reassembled PDU]
0.043787	10.10.10.3		10.10.10.11	TCP	56 http-alt > 44079 [ACK] Seq=1 Ack=49743 Win=64380 Len=0
0.000165	10.10.10.3		10.10.10.11	TCP	56 http-alt > 44079 [ACK] Seq=1 Ack=51791 Win=64380 Len=0
0.000222	10.10.10.3		10.10.10.11	TCP	56 http-alt > 44079 [ACK] Seq=1 Ack=53839 Win=64380 Len=0
0.000003	10.10.10.3		10.10.10.11	TCP	56 http-alt > 44079 [ACK] Seq=1 Ack=55887 Win=64380 Len=0
0.000304	10.10.10.11		10.10.10.3	TCP	684 [TCP segment of a reassembled PDU]
0.000313	10.10.10.3		10.10.10.11	TCP	56 http-alt > innosys [ACK] Seq=1 Ack=7175261 Win=64380 Len=0
0.000634	10.10.10.11		10.10.10.3	TCP	1476 [TCP segment of a reassembled PDU]
0.000183	10.10.10.11		10.10.10.3	TCP	1476 [TCP segment of a reassembled PDU]
0.000194	10.10.10.11		10.10.10.3	TCP	1476 [TCP segment of a reassembled PDU]
0.002574	10.10.10.11		10.10.10.3	TCP	1476 [TCP Out-Of-Order] [TCP segment of a reassembled PDU]
0.000024	10.10.10.11		10.10.10.3	TCP	1476 [TCP segment of a reassembled PDU]
0.000119	10.10.10.11		10.10.10.3	TCP	1476 [TCP segment of a reassembled PDU]
0.002502	10.10.10.11		10.10.10.3	TCP	1476 [TCP Out-Of-Order] [TCP segment of a reassembled PDU]
0.000022	10.10.10.11		10.10.10.3	TCP	1476 [TCP segment of a reassembled PDU]
0.015482	10.10.10.3		10.10.10.11	TCP	56 http-alt > innosys [ACK] Seq=1 Ack=7178101 Win=64380 Len=0
0.000005	10.10.10.3	快	10.10.10.11	TCP	56 http-alt > innosys [ACK] Seq=1 Ack=7180941 Win=64380 Len=0
0.000341	10.10.10.3		10.10.10.11	TCP	56 http-alt > innosys [ACK] Seq=1 Ack=7183781 Win=64380 Len=0
0.000004	10.10.10.3		10.10.10.11	TCP	56 http-alt > innosys [ACK] Seq=1 Ack=7186621 Win=64380 Len=0
0.000570	10.10.10.11		10.10.10.3	TCP	1476 [TCP segment of a reassembled PDU]

3G实验室upload rate

- Analysis

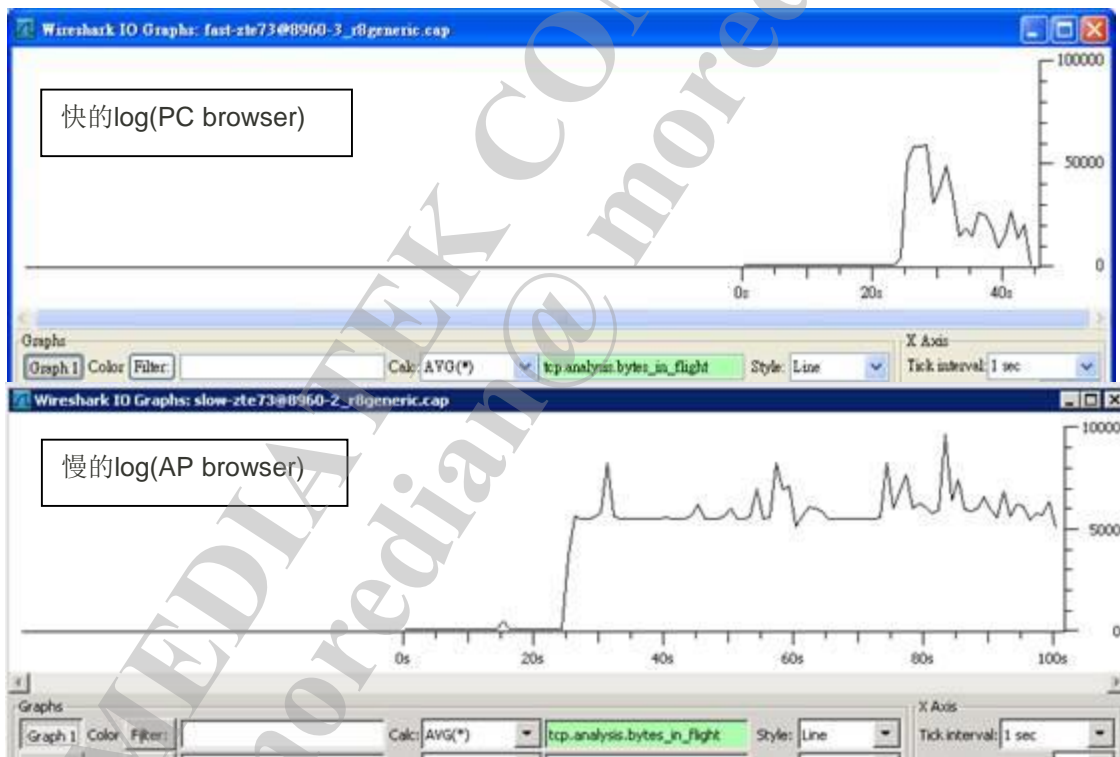
- 但是从RTT来看，慢的RTT并不是很长（在实验室环境，route path应该都是一样的，所以RTT不会差别太大）



3G实验室upload rate

■ Analysis

- 比较usb tethering upload file与mobile browser upload的tcpdump，发现AP browser的bytes in flight看起来比较小，PC browser一开始就可以衝到60000bytes，AP browser最多不到10000Bytes，为了提高bytes_in_flight,可以增大sendbuffer



3G实验室upload rate

■ Analysis

- 根据buffer size与BDP之间关系，估算upload rate为5.76Mbps所需要buffer size，： rtt为0.05s

- $2 * (5.76 / 8 * 0.05) = 0.072M$

Optimal RCV buffer size is considered to be $2 \times BDP$, where BDP is the **Bandwidth * Delay Product**

ex) RTT is 20 ms, and connection speed is 10 Mbps.
 $2 \times (10Mbps / 8 * .020s) = 50Kbytes$

- browser AP采用SO_SNDBUF option为8K，根据上面计算结果，将SO_SNDBUF option修改为32k

3G实验室upload rate

- Analysis

- 使用修改sendbuffer的load测试browser upload rate，达到要求速



Rate (Mbps)	1	2	3	4	5
Zte73v2	4.22	5.01	5.01	5.0	4.96

3G实验室upload rate

■ Solution

- 增大socket send buffer不失为提高upload rate的有效方法，对于使用SO_SNDBUF的APP来说，需要增大SO_SNDBUF的值，未使用socket option的APP来说，只需增大init.rc/DataConnection.java中send buffer值。
- 增大APP的temporary send buffer，可以有效减小包的分组和数量，提高upload rate



MEDIATEK

PC RNDIS performance check

MEDIATEK CONFIDENTIAL
moremedian@moremedian.com US

1. phone的环境设置

- 将iperf push到手机的/data/local/tmp下面,并做相应的权限设置
 - Adb push iperf /data/local/tmp/
 - Adb shell chmod 777 /data/local/tmp/iperf
- 开机后进入->设置->开发人员选项->勾选保持唤醒状态（充电时屏幕不会休眠）这样当设置CPU的频率后，频率就不会降回来
- 采用Iperf 2.05

3. PC环境设置

- 开机后应尽量关闭Kaspersky等杀毒软件
 - 因为如果开着的话会对performance有很大的影响,如果有豌豆夹之类的手机相关的软件也最好关闭。
- 在测试开始前, 断开PC的其他网络, 以免影响测试
- Notebook, 请链接电源测试, 如果用电池, 请调节到效能最高mode

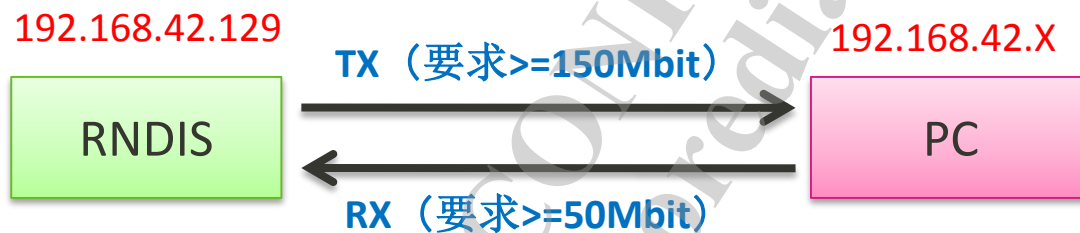
4. PC环境设置

- 将rndis网卡上的Kaspersky勾选去掉



5. 测试的项描述

- 5.1 PC 和 RNDIS 之间的RX和TX及双向



5.1.1 测试RX



phone 端 server: `/data/local/tmp/iperf -s -p 5001 -u -i 1 -l 1440 -w 1M`

PC 端 client: `iperf.exe -c 192.168.42.129 -p 5001 -i 1 -u -t 3000 -f m -b 200M -l 1440`

记录server的速度

5.1.2 测试TX



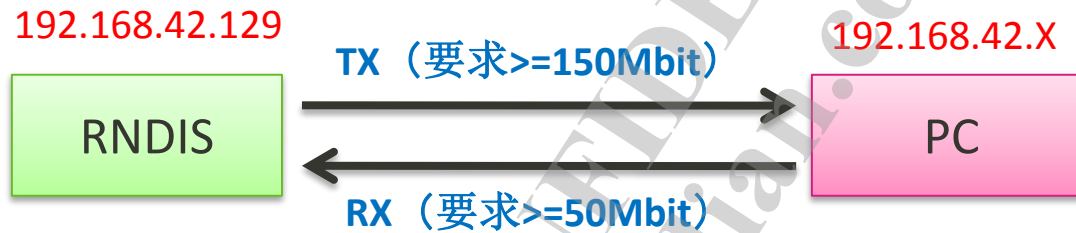
iperf 命令: **192.168.42.X** 表示 tethering PC IP

phone 端 client: `/data/local/tmp/iperf -c 192.168.42.X -p 5001 -i 1 -u -t 3000 -f m -b 200M -l 1440`

PC 端server: `iperf.exe -s -p 5001 -u -i 1 -l 1440 -w 1M`

记录server的速度

5.1.3 测试UDP双向



iperf 命令: 192.168.42.X 表示 tethering PC IP

phone 端 client: /data/local/tmp/iperf -c 192.168.42.X -p 5001 -i 1 -u -t 3000 -f m -b 200M -l 1440

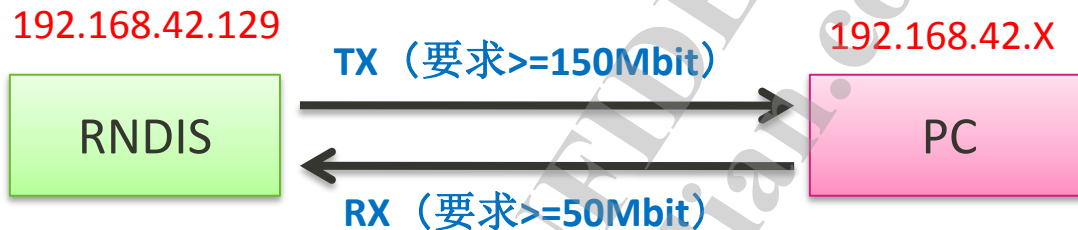
PC 端 server: iperf.exe -s -p 5001 -u -i 1 -l 1440 -w 1M

phone 端 server: /data/local/tmp/iperf -s -p 5001 -u -i 1 -l 1440 -w 800k

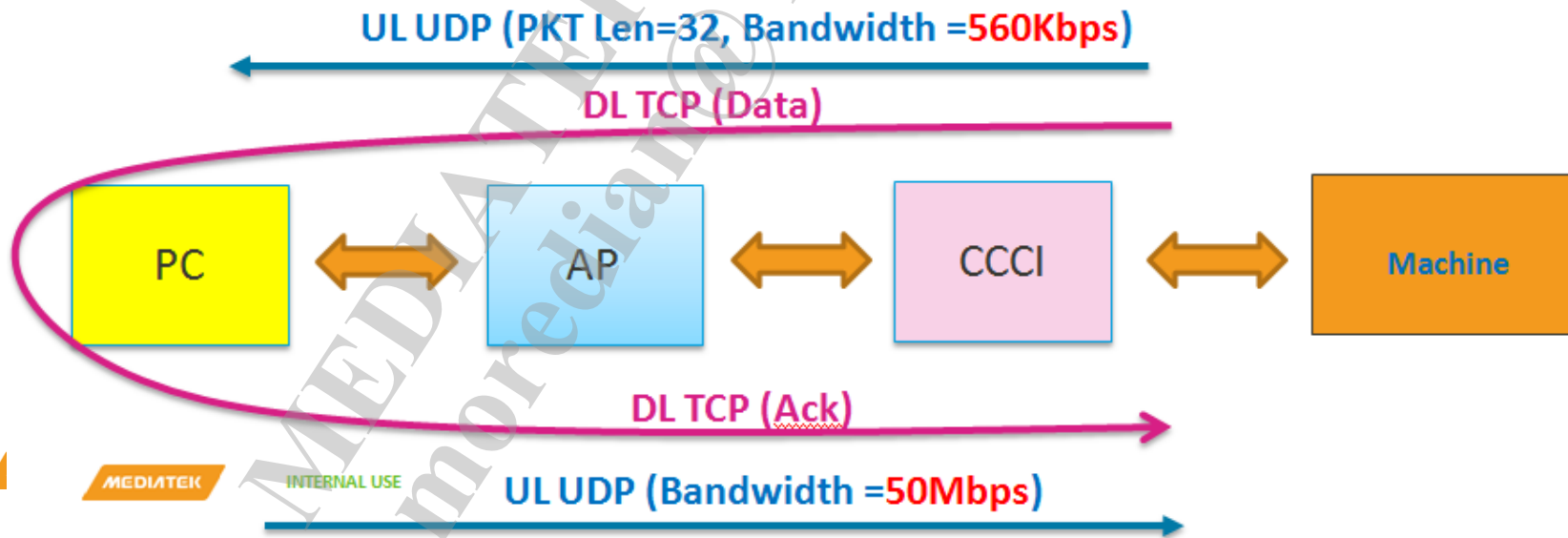
PC 端 client: iperf.exe -c 192.168.42.129 -p 5001 -i 1 -u -t 3000 -f m -b 80M -l 1440

记录server的速度

5.1.4 测试TCP双向



原理： 由于TCP 有flow control，如果2边都打TCP stream，最后会变成DL/UL相同T-put结果。所以我们在UL方向先打50Mbps的UDP stream（因为UDP能设定bandwidth），然后在DL打TCP，观察这条TCP能否达到150Mbps. 注意真正的UL是TCP，它在DL方向还有ACK PACKET，所以我们还要在DL方向加一条 UDP 来模拟UL方向的ACK PACKET



5.1.4 测试TCP双向

AUTO TEST : 测试比较复杂, 推荐用工具测试:

<http://wiki/display/oss3ss5/RNDIS+TCP+UT+TOOL>

手动命令:

USB TX/DL 测试命令:

phone 端 client: /data/local/tmp/iperf -c 192.168.42.X -p 5003 -i 1 -f m -t 3000

PC 端 server: iperf.exe -s -p 5003 -i 1 -f m -w 1M

phone 端 client : /data/local/tmp/iperf -c 192.168.42.X -p 5005 -u -i 1 -l 60 -f m -b 1M -t 9999

PC 端 server : iperf.exe -s -u -i 1 -l 60 -p 5005 -f m

USB RX/UL 测试命令:

phone 端 server : /data/local/tmp/iperf -s -u -i 1 -p 5004 -f m -w 800k

PC 端 client : iperf.exe -c 192.168.42.129 -p 5004 -u -i 1 -f m -b 55M -t 9999

记录5003 server的速度

6. 测试结果稳定性

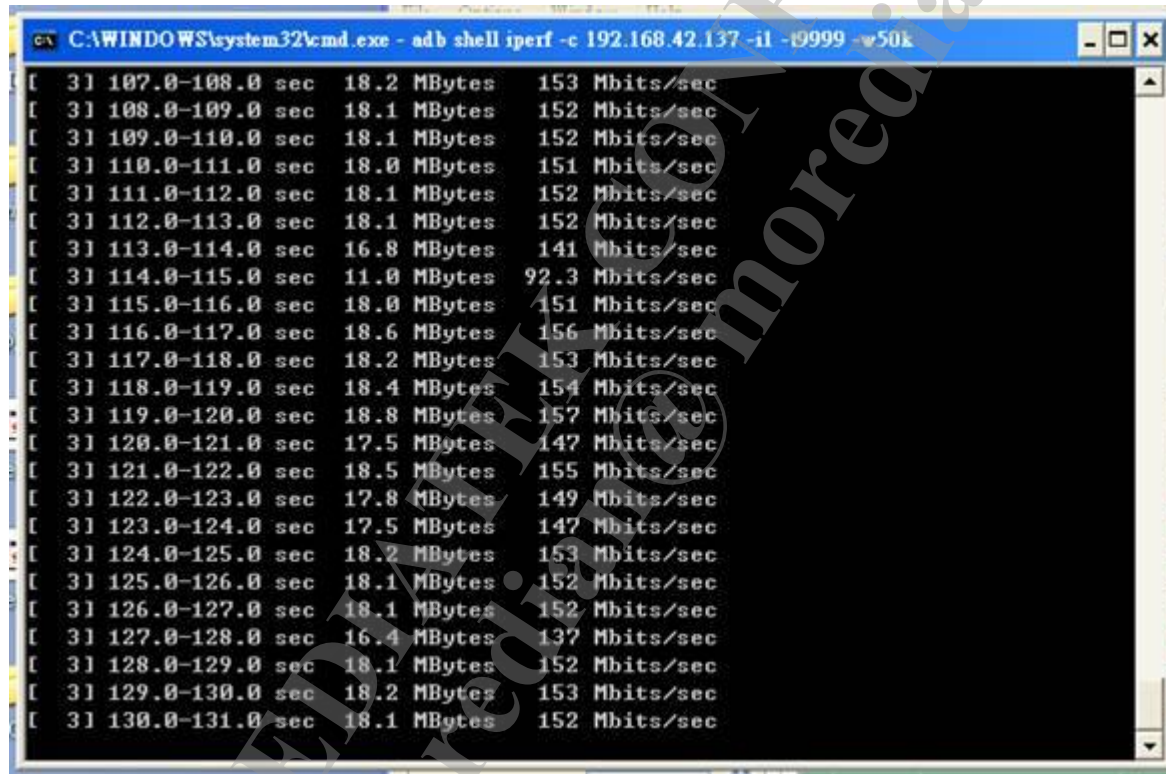
本次测试结果， T-Put 保持大于150Mbps/50Mbps 合格

[41	67.0-68.0	sec	20.6	MBytes	173	Mbits/sec				
[41	68.0-69.0	sec	21.5	MBytes	180	Mbits/sec				
[41	69.0-70.0	sec	21.6	MBytes	181	Mbits/sec				
[41	70.0-71.0	sec	21.5	MBytes	181	Mbits/sec				
[41	71.0-72.0	sec	21.5	MBytes	180	Mbits/sec				
[41	72.0-73.0	sec	21.1	MBytes	177	Mbits/sec				
[41	73.0-74.0	sec	21.2	MBytes	178	Mbits/sec				
[41	74.0-75.0	sec	21.5	MBytes	181	Mbits/sec				
[41	75.0-76.0	sec	21.5	MBytes	180	Mbits/sec				
[41	76.0-77.0	sec	20.3	MBytes	171	Mbits/sec				
[41	77.0-78.0	sec	21.5	MBytes	180	Mbits/sec				
[41	78.0-79.0	sec	21.5	MBytes	180	Mbits/sec				
[41	79.0-80.0	sec	21.5	MBytes	181	Mbits/sec				
[41	80.0-81.0	sec	21.6	MBytes	181	Mbits/sec				
[41	81.0-82.0	sec	21.6	MBytes	181	Mbits/sec				
[41	82.0-83.0	sec	20.9	MBytes	175	Mbits/sec				
[41	83.0-84.0	sec	21.5	MBytes	181	Mbits/sec				
[41	84.0-85.0	sec	21.6	MBytes	181	Mbits/sec				
[41	85.0-86.0	sec	21.6	MBytes	181	Mbits/sec				
[41	86.0-87.0	sec	21.4	MBytes	180	Mbits/sec				
[41	87.0-88.0	sec	21.0	MBytes	176	Mbits/sec				
[41	88.0-89.0	sec	21.0	MBytes	176	Mbits/sec				
[41	89.0-90.0	sec	21.3	MBytes	179	Mbits/sec				
[41	90.0-91.0	sec	21.5	MBytes	180	Mbits/sec				

[31	102.0-103.0	sec	5.97	MBytes	50.0	Mbits/sec	0.129	ms	0/	4255 (<0%)
[31	103.0-104.0	sec	5.97	MBytes	50.1	Mbits/sec	0.157	ms	0/	4256 (<0%)
[31	104.0-105.0	sec	5.97	MBytes	50.0	Mbits/sec	0.153	ms	0/	4255 (<0%)
[31	105.0-106.0	sec	5.97	MBytes	50.0	Mbits/sec	0.184	ms	0/	4255 (<0%)
[31	106.0-107.0	sec	5.97	MBytes	50.1	Mbits/sec	0.190	ms	0/	4256 (<0%)
[31	107.0-108.0	sec	5.97	MBytes	50.0	Mbits/sec	0.161	ms	0/	4255 (<0%)
[31	108.0-109.0	sec	5.97	MBytes	50.0	Mbits/sec	0.181	ms	0/	4255 (<0%)
[31	109.0-110.0	sec	5.97	MBytes	50.1	Mbits/sec	0.113	ms	0/	4256 (<0%)
[31	110.0-111.0	sec	5.97	MBytes	50.0	Mbits/sec	0.132	ms	0/	4255 (<0%)
[31	111.0-112.0	sec	5.97	MBytes	50.0	Mbits/sec	0.131	ms	0/	4255 (<0%)
[31	112.0-113.0	sec	5.97	MBytes	50.1	Mbits/sec	0.119	ms	0/	4256 (<0%)
[31	113.0-114.0	sec	5.97	MBytes	50.0	Mbits/sec	0.145	ms	0/	4255 (<0%)
[31	114.0-115.0	sec	5.97	MBytes	50.0	Mbits/sec	0.175	ms	0/	4255 (<0%)
[31	115.0-116.0	sec	5.97	MBytes	50.1	Mbits/sec	0.184	ms	0/	4256 (<0%)
[31	116.0-117.0	sec	5.97	MBytes	50.0	Mbits/sec	0.187	ms	0/	4255 (<0%)
[31	117.0-118.0	sec	5.97	MBytes	50.0	Mbits/sec	0.134	ms	0/	4255 (<0%)
[31	118.0-119.0	sec	5.97	MBytes	50.0	Mbits/sec	0.191	ms	0/	4255 (<0%)
[31	119.0-120.0	sec	5.97	MBytes	50.1	Mbits/sec	0.128	ms	0/	4256 (<0%)
[31	120.0-121.0	sec	5.97	MBytes	50.0	Mbits/sec	0.148	ms	0/	4255 (<0%)
[31	121.0-122.0	sec	5.97	MBytes	50.0	Mbits/sec	0.191	ms	0/	4255 (<0%)
[31	122.0-123.0	sec	5.97	MBytes	50.1	Mbits/sec	0.135	ms	0/	4256 (<0%)
[31	123.0-124.0	sec	5.97	MBytes	50.0	Mbits/sec	0.171	ms	0/	4255 (<0%)
[31	124.0-125.0	sec	5.97	MBytes	50.0	Mbits/sec	0.141	ms	0/	4255 (<0%)
[31	125.0-126.0	sec	5.97	MBytes	50.1	Mbits/sec	0.118	ms	0/	4256 (<0%)

6. 测试结果稳定性

测试结果要保证稳定性，下面测试结果，T-Put 突然下降，不合格



```
C:\WINDOWS\system32\cmd.exe - adb shell iperf -c 192.168.42.137 -il -t9999 -w50k
[ 3] 107.0-108.0 sec 18.2 MBytes 153 Mbits/sec
[ 3] 108.0-109.0 sec 18.1 MBytes 152 Mbits/sec
[ 3] 109.0-110.0 sec 18.1 MBytes 152 Mbits/sec
[ 3] 110.0-111.0 sec 18.0 MBytes 151 Mbits/sec
[ 3] 111.0-112.0 sec 18.1 MBytes 152 Mbits/sec
[ 3] 112.0-113.0 sec 18.1 MBytes 152 Mbits/sec
[ 3] 113.0-114.0 sec 16.8 MBytes 141 Mbits/sec
[ 3] 114.0-115.0 sec 11.0 MBytes 92.3 Mbits/sec
[ 3] 115.0-116.0 sec 18.0 MBytes 151 Mbits/sec
[ 3] 116.0-117.0 sec 18.6 MBytes 156 Mbits/sec
[ 3] 117.0-118.0 sec 18.2 MBytes 153 Mbits/sec
[ 3] 118.0-119.0 sec 18.4 MBytes 154 Mbits/sec
[ 3] 119.0-120.0 sec 18.8 MBytes 157 Mbits/sec
[ 3] 120.0-121.0 sec 17.5 MBytes 147 Mbits/sec
[ 3] 121.0-122.0 sec 18.5 MBytes 155 Mbits/sec
[ 3] 122.0-123.0 sec 17.8 MBytes 149 Mbits/sec
[ 3] 123.0-124.0 sec 17.5 MBytes 147 Mbits/sec
[ 3] 124.0-125.0 sec 18.2 MBytes 153 Mbits/sec
[ 3] 125.0-126.0 sec 18.1 MBytes 152 Mbits/sec
[ 3] 126.0-127.0 sec 18.1 MBytes 152 Mbits/sec
[ 3] 127.0-128.0 sec 16.4 MBytes 137 Mbits/sec
[ 3] 128.0-129.0 sec 18.1 MBytes 152 Mbits/sec
[ 3] 129.0-130.0 sec 18.2 MBytes 153 Mbits/sec
[ 3] 130.0-131.0 sec 18.1 MBytes 152 Mbits/sec
```

MEDIATEK

everyday genius