Running udpmon on the eduPERT VMs and mininet

This note describes how to use udpmon to make the network tests described in the eduPERT workshop using: either (1) the path between two VMs in the cloud hosted by SWITCH, or (2) using a mininet topology running on either one VM, or (3) on your laptop.

You can find the udpmon code and excel files on

<http://www.hep.man.ac.uk/u/rich/net/index.html>

# Using two VMs in the SWITCH Cloud

udpmon and the command files have already been installed on the Cloud VMs.

It is recommended to work in pairs, each person logged into one VM using the username “ptraining” and one running the udpmon server e.g. udpmon\_resp, the other the udpmon client e.g. udpmon\_bw\_mon.

For illustration purposes, let user A login to pert1 and user B login to pert2; user B will run the udpmon server udpmon\_resp in these examples.

## Simple udpmon test to show effect of socket buffer size

### User B on pert2

1. cd to the udpmon directory
cd udpmon-1.4.1
2. Use ifconfig to obtain the IP address of the host
let’s say it is 10.0.0.20
3. Start udpmon\_resp

./udpmon\_resp

###  User A on pert1

1. cd to the udpmon directory
cd udpmon-1.4.1
2. Run udpmon with a 300 Mbit/s UDP flow
./udpmon\_bw\_mon -d 10.0.0.20 -p1472 -w40 -l1000

 Fri Jun 12 15:09:16 2015 : UDPmon Version 1.4.1 CPUs Command line: ./udpmon\_bw\_mon -d 10.0.0.20 -p1472 -w40 -l1000

Default affinity: 00000003 Set affinity: 00000003

 Initialising real-time clock.

CPU speed = 2593.6738126 +- 0.0188770 MHz (2593.6288266 - 2593.7177908)

StopWatch\_Stop calibrated: 0.0108 us minimum : StopWatch\_Delay calibrated: 0.0177 us minimum

clock ticks for 1 sec = 1000168 us

Defaults: RCVBUF 229376 SNDBUF 229376 IP\_TOS 0x0 IP prec 0 IP tos 0

Settings: RCVBUF 65535 SNDBUF 65535 IP\_TOS 0x0 IP prec 0 IP tos 0

 The destination IP name: 10.0.0.20 IP address: 10.0.0.20

 The destination UDP port is 14233 3799

 1472 bytes

 pkt len; num\_sent; inter-pkt\_time us; send\_user\_data\_rate Mbit; num\_recv; num\_lost; num\_badorder; %lost; num\_lost\_innet; %lost\_innet; recv\_user\_data\_rate Mbit; recv\_wire\_rate Mbit;

 1472; 1000; 40; 293.446; 851; 149; 0; **15**; 59; **5.9**; 283.772; 296.495;

Notice the difference between the percentage losses at the application and losses in the network. Try with a large socket buffer on the receiver.

###  User B on pert2

1. Stop udpmon\_resp with control-c and restart with 1MByte buffer

./udpmon\_resp –S 1000000

###  User A on pert1

1. Run udpmon with a 300 Mbit/s UDP flow
./udpmon\_bw\_mon -d 10.0.0.20 -p1472 -w40 -l1000

 Fri Jun 12 15:09:52 2015 : UDPmon Version 1.4.1 CPUs Command line: ./udpmon\_bw\_mon -d 10.0.0.20 -p1472 -w40 -l1000

Default affinity: 00000003 Set affinity: 00000003

 Initialising real-time clock.

CPU speed = 2593.6390264 +- 0.0179016 MHz (2593.5967215 -2593.6813250)

StopWatch\_Stop calibrated: 0.0108 us minimum : StopWatch\_Delay calibrated: 0.0216 us minimum

clock ticks for 1 sec = 1000169 us

Defaults: RCVBUF 229376 SNDBUF 229376 IP\_TOS 0x0 IP prec 0 IP tos 0

Settings: RCVBUF 65535 SNDBUF 65535 IP\_TOS 0x0 IP prec 0 IP tos 0

 The destination IP name: 10.0.0.20 IP address: 10.0.0.20

 The destination UDP port is 14233 3799

 1472 bytes

 pkt len; num\_sent; inter-pkt\_time us; send\_user\_data\_rate Mbit; num\_recv; num\_lost; num\_badorder; %lost; num\_lost\_innet; %lost\_innet; recv\_user\_data\_rate Mbit; recv\_wire\_rate Mbit;

 1472; 1000; 40; 293.067; 948; 52; 0; **5.2**; 52; **5.2**; 281.607; 294.234;

The large socket buffer on the receiver reduces losses in the IP stack of the VM.

## Measure packet jitter with udpmon

###  User B on pert2

1. Start udpmon\_resp with 1MByte buffer

./udpmon\_resp –S 1000000

###  User A on pert1

Run udpmon sending a UDP flow with 100 μs packet spacing and histograms
./udpmon\_bw\_mon -d 10.0.0.20 -p1472 –w100 -l1000 -H

## Running a set of udpmon tests using the command files

The command files provide a simple way to make a set of tests and record the results of each test in a separate “;” delimited text file. These text files are named with a prefix that you give, the date, and a postfix that is a test number. They may be imported into the excel files using the loadall macro.

cmd\_jitter.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

cmd\_1waydelay.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

cmd\_throughput\_lite.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

cmd\_all\_lite.pl runs all the three above files

###  User B on pert2

1. Start udpmon\_resp with 1MByte buffer

./udpmon\_resp –S 1000000

###  User A on pert1

1. Edit the file cmd\_all\_lite to set the file prefix and the destination IP address. For example

DEST=10.0.0.20

FILEPREFIX=Rlap\_h1-2

./cmd\_jitter.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

./cmd\_1waydelay.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

./cmd\_throughput\_lite.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

1. Start the test by entering

./cmd\_all\_lite.pl

The text files will be put in the current directory which should be the directory with the udpmon distribution.

1. Copy the files to your laptop and plot them with the udpmon excel file udpmon\_all-lite.xlsm by importing using the “loadall” macro

## Investigate packet loss distributions with udpmon -L

###  User B on pert2

1. Start udpmon\_resp with 1MByte buffer

./udpmon\_resp –S 1000000

###  User A on pert1

1. Run udpmon sending a UDP flow with 300 Mbit UDP flow recording lost packets. This may be done to a file with:

./udpmon\_bw\_mon -d 10.0.0.20 -p1472 –w40 -l1000 –L50 > pert1-2\_13Jun15\_01.txt

Or to the terminal with:
./udpmon\_bw\_mon -d 10.0.0.20 -p1472 –w40 -l1000 –L50

Sat Jun 13 08:07:19 2015 : UDPmon Version 1.4.1 CPUs Command line: ./udpmon\_bw\_mon -d 10.0.0.4 -p1472 -w40 -l1000 -L 50

Default affinity: 00000003 Set affinity: 00000003

 Initialising real-time clock.

CPU speed = 2593.6480171 +- 0.0172132 MHz (2593.6069770 - 2593.6872577)

StopWatch\_Stop calibrated: 0.0108 us minimum : StopWatch\_Delay calibrated: 0.0197 us minimum

clock ticks for 1 sec = 1000148 us

Defaults: RCVBUF 229376 SNDBUF 229376 IP\_TOS 0x0 IP prec 0 IP tos 0

Settings: RCVBUF 65535 SNDBUF 65535 IP\_TOS 0x0 IP prec 0 IP tos 0

 The destination IP name: 10.0.0.20 IP address: 10.0.0.20

 The destination UDP port is 14233 3799

 1472 bytes

 pkt len; num\_sent; inter-pkt\_time us; send\_user\_data\_rate Mbit; num\_recv; num\_lost; num\_badorder; %lost; num\_lost\_innet; %lost\_innet; recv\_user\_data\_rate Mbit; recv\_wire\_rate Mbit;

Error: No response for TSYNC from remote host 10.0.0.20 : Success

 1472; 1000; 40; 292.848; 951; 49; 0; 4.9; 49; 4.9; 292.209; 305.311;

 num packets skipped =; 0

lost event; recv\_time 0.1us; send\_time 0.1us; diff 0.1us; one\_way time us; lost packet num; ;delta recv\_time us; delta send\_time us; num packets between losses;

 1; 190960513; 189918713; 1041800; -1.89919e+07; 4; ; 1.90961e+07; 1.89919e+07; 4

 2; 190960975; 189932882; 1028093; -1.89933e+07; 39; ; 46.2; 1416.9; 35

 3; 190970797; 189936246; 1034551; -1.89936e+07; 47; ; 982.2; 336.4; 8

 4; 190978901; 189948259; 1030642; -1.89948e+07; 77; ; 810.4; 1201.3; 30

 5; 190982076; 189957069; 1025007; -1.89957e+07; 99; ; 317.5; 881; 22

 6; 190997464; 189971484; 1025980; -1.89971e+07; 135; ; 1538.8; 1441.5; 36

 7; 191007570; 189978065; 1029505; -1.89978e+07; 151; ; 1010.6; 658.1; 16

 8; 191011895; 189985273; 1026622; -1.89985e+07; 169; ; 432.5; 720.8; 18

 9; 191025316; 189999285; 1026031; -1.89999e+07; 204; ; 1342.1; 1401.2; 35

 10; 191026711; 190001687; 1025024; -1.90002e+07; 210; ; 139.5; 240.2; 6

 11; 191039589; 190012497; 1027092; -1.90012e+07; 237; ; 1287.8; 1081; 27

 12; 191039629; 190014099; 1025530; -1.90014e+07; 241; ; 4; 160.2; 4

 13; 191046122; 190019429; 1026693; -1.90019e+07; 254; ; 649.3; 533; 13

 14; 191047680; 190021831; 1025849; -1.90022e+07; 260; ; 155.8; 240.2; 6

 15; 191052594; 190024277; 1028317; -1.90024e+07; 266; ; 491.4; 244.6; 6

 16; 191066951; 190040641; 1026310; -1.90041e+07; 306; ; 1435.7; 1636.4; 40

 17; 191066951; 190040641; 1026310; -1.90041e+07; 307; ; 0; 0; 1

 18; 191082713; 190057202; 1025511; -1.90057e+07; 346; ; 1576.2; 1656.1; 39

 19; 191097328; 190070804; 1026524; -1.90071e+07; 380; ; 1461.5; 1360.2; 34

 20; 191097341; 190071605; 1025736; -1.90072e+07; 382; ; 1.3; 80.1; 2

 21; 191106691; 190082007; 1024684; -1.90082e+07; 408; ; 935; 1040.2; 26

 22; 191106691; 190082007; 1024684; -1.90082e+07; 409; ; 0; 0; 1

 ...

 47; 191322300; 190298811; 1023489; -1.90299e+07; 949; ; 244.1; 280.1; 7

 48; 191333844; 190310013; 1023831; -1.9031e+07; 977; ; 1154.4; 1120.2; 28

 49; 191341745; 190317614; 1024131; -1.90318e+07; 996; ; 790.1; 760.1; 19

 50; 0; 0; 0; 0; 0; ; -1.91342e+07; -1.90318e+07; -996

1. Copy the files to your laptop and plot them with the udpmon excel file udpmon\_LossDist.xlsm by importing using the “loadall” macro.

# Using mininet on a VM in the SWITCH Cloud

mininet, the topology file, udpmon and the command files have already been installed on the Cloud VMs, so you can just start mininet and run the udpmon tests.

## Simple udpmon test

1. ssh to the mininet-vm
cd to udpmon-1.4.1
2. Start mininet and load the topology- you need to give the full location of the topology file, given here as ~/TNC2015/

sudo mn --custom ~/TNC2015/topo-ps-tutorial.py --topo mytopo
3. Start udpmon\_resp in the background on h2

mininet> h2 ../bin/udpmon\_resp &
4. Start udpmon\_bw\_mon to send a stream of spaced UDP packets
(-w 123 gives about 100 Mbit/s)

mininet> h1 ../bin/udpmon\_bw\_mon -d 10.0.0.2 -p1472 -w 123 -l 1000

The output is given on the terminal something like the following giving the number of packets sent, received, lost, and out of order, as well as the percentage lost and the received data rates.

Sun May 31 00:40:27 2015 : UDPmon Version 1.4.1 CPUs Command line: ../bin/udpmon\_bw\_mon -d 10.0.0.2 -p1472 -w 123 -l 1000

Default affinity: 00000001 Set affinity: 00000001

 Initialising real-time clock.

CPU speed = 2120.7192521 +- 0.0122813 MHz (2120.6072815 - 2120.7461674)

StopWatch\_Stop calibrated: 0.0113 us minimum : StopWatch\_Delay calibrated: 0.0384 us minimum

clock ticks for 1 sec = 1000359 us

Defaults: RCVBUF 163840 SNDBUF 163840 IP\_TOS 0x0 IP prec 0 IP tos 0

Settings: RCVBUF 65535 SNDBUF 65535 IP\_TOS 0x0 IP prec 0 IP tos 0

 The destination IP name: 10.0.0.2 IP address: 10.0.0.2

 The destination UDP port is 14233 3799

 1472 bytes

 pkt len; num\_sent; inter-pkt\_time us; send\_user\_data\_rate Mbit; num\_recv; num\_lost; num\_badorder; %lost; num\_lost\_innet; %lost\_innet; recv\_user\_data\_rate Mbit; recv\_wire\_rate Mbit;

 1472; 1000; 123; 57.3597; 914; 239; 153; 8.6; 160; 16; 63.1047; 65.9342;

## Simple udpmon test with large 1 Mbyte socket buffer

We can vary this by giving a large socket buffer to udpmon\_resp. If udpmon\_resp is already running stop it with

mininet> h2 killall udpmon\_resp

A 1 Mbyte buffer is specified with

mininet> h2 ../bin/udpmon\_resp –S 1000000 &

## Running a set of udpmon tests using the command files

The command files provide a simple way to make a set of tests and record the results of each test in a separate “;” delimited text file. These text files are named with a prefix that you give, the date, and a postfix that is a test number. They may be imported into the excel files using the loadall macro.

cmd\_jitter.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

cmd\_1waydelay.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

cmd\_throughput\_lite.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

cmd\_all\_lite.pl runs all the three above files

Proceed with steps 1-3 as given in section 2.1, unless mininet and udpmon\_resp are already running. Then:

1. Edit the file cmd\_all\_lite to set the file prefix and the destination IP address. For example

DEST=10.0.0.2

FILEPREFIX=Rlap\_h1-2

./cmd\_jitter.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

./cmd\_1waydelay.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

./cmd\_throughput\_lite.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

1. Start the test by entering

mininet> h1 ./cmd\_all\_lite.pl

The text files will be put in the current directory which should be the directory with the udpmon distribution.

1. Copy the files to your laptop and plot them with the udpmon excel file udpmon\_all-lite.xlsm by importing using the “loadall” macro

# Using mininet on your laptop

## Installation

1. Install mininet on a VM on your laptop. I installed mininet by downloading and importing the mininet VM into VirtualBox.
2. Start the mininet-vm
3. Install udpmon
* Download udpmon-1.4.1 and libhj-4.5.0 tarballs from
<http://www.hep.man.ac.uk/u/rich/net/index.html>
* Login to the mininet-vm
* Get the IP address and ssh into the mininet-vm
* scp the tarballs to the mininet-vm
* ssh into the mininet-vm to unpack and build udpmon

## Making UDP tests

### Topology

Get a copy of the topology file topo-ps-tutorial.py and put it onto the mininet-vm with scp.

The topology is made up of:

host --- switch --- switch -- switch --- host

### Simple udpmon test

1. ssh to the mininet-vm
cd to udpmon-1.4.1
2. Start mininet and load the topology- you need to give the full location of the topology file, given here as ~/TNC2015/
sudo mn --custom ~/TNC2015/topo-ps-tutorial.py --topo mytopo
3. Start udpmon\_resp in the background on h2
mininet> h2 ../bin/udpmon\_resp &
4. Start udpmon\_bw\_mon to send a stream of spaced UDP packets
(-w 123 gives about 100 Mbit/s)
mininet> h1 ../bin/udpmon\_bw\_mon -d 10.0.0.2 -p1472 -w 123 -l 1000

The output is given on the terminal something like the following giving the number of packets sent, received, lost, and out of order, as well as the percentage lost and the received data rates.

Sun May 31 00:40:27 2015 : UDPmon Version 1.4.1 CPUs Command line: ../bin/udpmon\_bw\_mon -d 10.0.0.2 -p1472 -w 123 -l 1000

Default affinity: 00000001 Set affinity: 00000001

 Initialising real-time clock.

CPU speed = 2120.7192521 +- 0.0122813 MHz (2120.6072815 - 2120.7461674)

StopWatch\_Stop calibrated: 0.0113 us minimum : StopWatch\_Delay calibrated: 0.0384 us minimum

clock ticks for 1 sec = 1000359 us

Defaults: RCVBUF 163840 SNDBUF 163840 IP\_TOS 0x0 IP prec 0 IP tos 0

Settings: RCVBUF 65535 SNDBUF 65535 IP\_TOS 0x0 IP prec 0 IP tos 0

 The destination IP name: 10.0.0.2 IP address: 10.0.0.2

 The destination UDP port is 14233 3799

 1472 bytes

 pkt len; num\_sent; inter-pkt\_time us; send\_user\_data\_rate Mbit; num\_recv; num\_lost; num\_badorder; %lost; num\_lost\_innet; %lost\_innet; recv\_user\_data\_rate Mbit; recv\_wire\_rate Mbit;

 1472; 1000; 123; 57.3597; 914; 239; 153; 8.6; 160; 16; 63.1047; 65.9342;

We can vary this by giving a large socket buffer to udpmon\_resp

A 1 Mbyte buffer is specified with

mininet> h2 ../bin/udpmon\_resp –S 1000000&

### Running a set of udpmon tests using the command files

The command files provide a simple way to make a set of tests and record the results of each test in a separate “;” delimited text file. These text files are named with a prefix that you give, the date, and a postfix that is a test number. They may be imported into the excel files using the loadall macro.

cmd\_jitter.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

cmd\_1waydelay.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

cmd\_throughput\_lite.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

cmd\_all\_lite.pl runs all the three above files

Proceed with steps 1-3 as given in the above section, unless mininet and udpmon\_resp are already running. Then:

1. Edit the file cmd\_all\_lite to set the file prefix and the destination IP address. For example

DEST=10.0.0.2

FILEPREFIX=Rlap\_h1-2

./cmd\_jitter.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

./cmd\_1waydelay.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

./cmd\_throughput\_lite.pl -o ${FILEPREFIX} -d ${DEST} -l 10000

1. Start the test by entering

mininet> h1 ./cmd\_all\_lite.pl

The text files will be put in the current directory which should be the directory with the udpmon distribution.

1. Copy the files to your laptop and plot them with the udpmon excel file udpmon\_all-lite.xlsm by importing using the “loadall” macro

### Examining the packet loss distribution

1. Edit the topo-ps-tutorial.py file to set some packet loss on the link, the example below uses 2%

"""Simple topology for perfSONAR training

Three directly connected switches plus a host for each switch:

 host --- switch --- switch -- switch --- host

"""

from mininet.topo import Topo

from mininet.link import TCLink

class MyTopo( Topo ):

 def \_\_init\_\_( self ):

 "Create custom topo."

 # Initialize topology

 Topo.\_\_init\_\_( self )

 my\_bw = 1000 # note: max mininet bw = 1000

 # for no loss use this

 #my\_loss = 0

 # for 1% loss use this

 **my\_loss = 2**

 switch1 = self.addSwitch('s1')

 switch2 = self.addSwitch('s2')

 switch3 = self.addSwitch('s3')

 host1 = self.addHost('h1')

 host2 = self.addHost('h2')

 self.addLink(switch1, switch2, cls=TCLink,

 bw=my\_bw,delay='30ms',**loss=my\_loss**,use\_htb=True)

 self.addLink(switch2, switch3, cls=TCLink,

 bw=my\_bw, delay='5ms', loss=0, use\_htb=True)

 self.addLink(host1, switch1, cls=TCLink, bw=my\_bw)

 self.addLink(switch3, host2, cls=TCLink, bw=my\_bw)

topos = { 'mytopo': ( lambda: MyTopo() ) }

1. Start mininet and load the topology- you need to give the full location of the topology file, given here as ~/TNC2015/
sudo mn --custom ~/TNC2015/topo-ps-tutorial.py --topo mytopo
2. Start udpmon\_resp with a 1 Mbyte socket buffer in the background on h2
mininet> h2 ../bin/udpmon\_resp –S 1000000&
3. Start udpmon\_bw\_mon to send a stream of 2000 spaced UDP packets and record 30 lost packet events.
(-w 1230 gives about 10 Mbit/s)
mininet> h1 ../bin/udpmon\_bw\_mon -d 10.0.0.2 -p1472 -w 1230 -l 2000 –L30

The output looks something like

Sun May 31 00:47:52 2015 : UDPmon Version 1.4.1 CPUs Command line: ../bin/udpmon\_bw\_mon -d 10.0.0.2 -p1472 -w 1230 -l 2000 -L30

Default affinity: 00000001 Set affinity: 00000001

 Initialising real-time clock.

CPU speed = 2120.7181818 +- 0.0151021 MHz (2120.4392392 - 2120.7452960)

StopWatch\_Stop calibrated: 0.0099 us minimum : StopWatch\_Delay calibrated: 0.0409 us minimum

clock ticks for 1 sec = 1000144 us

Defaults: RCVBUF 163840 SNDBUF 163840 IP\_TOS 0x0 IP prec 0 IP tos 0

Settings: RCVBUF 65535 SNDBUF 65535 IP\_TOS 0x0 IP prec 0 IP tos 0

 The destination IP name: 10.0.0.2 IP address: 10.0.0.2

 The destination UDP port is 14233 3799

 1472 bytes

 pkt len; num\_sent; inter-pkt\_time us; send\_user\_data\_rate Mbit; num\_recv; num\_lost; num\_badorder; %lost; num\_lost\_innet; %lost\_innet; recv\_user\_data\_rate Mbit; recv\_wire\_rate Mbit;

 1472; 2000; 1230; 9.14628; 1968; 32; 0; 1.6; 32; 1.6; 9.00581; 9.4096;

 num packets skipped =; 0

lost event; recv\_time 0.1us; send\_time 0.1us; diff 0.1us; one\_way time us; lost packet num; ;delta recv\_time us; delta send\_time us; num packets between losses;

 1; 227706127; 226975956; 730171; 35593; 43; ; 2.27706e+07; 2.26976e+07; 43

 2; 228298379; 227572453; 725926; 35167.5; 90; ; 59225.2; 59649.7; 47

 3; 228889704; 228160615; 729089; 35482.7; 136; ; 59132.5; 58816.2; 46

 4; 230691523; 229956006; 735517; 36122.3; 276; ; 180182; 179539; 140

 5; 231739224; 231008803; 730421; 35610.9; 356; ; 104770; 105280; 80

 6; 231790905; 231061227; 729678; 35536.5; 360; ; 5168.1; 5242.4; 4

 7; 231939126; 231204144; 734982; 36066.6; 371; ; 14822.1; 14291.7; 11

 8; 232278593; 231548964; 729629; 35530.7; 398; ; 33946.7; 34482; 27

 9; 232960277; 232229251; 731026; 35669.2; 450; ; 68168.4; 68028.7; 52

 10; 233322037; 232587398; 734639; 36029.8; 478; ; 36176; 35814.7; 28

 11; 234427045; 233696702; 730343; 35598.3; 564; ; 110501; 110930; 86

 12; 236035566; 235303367; 732199; 35781; 690; ; 160852; 160666; 126

 13; 236607831; 235877099; 730732; 35633.3; 735; ; 57226.5; 57373.2; 45

 14; 237073779; 236346097; 727682; 35327.5; 771; ; 46594.8; 46899.8; 36

 15; 237817891; 237088370; 729521; 35510.1; 829; ; 74411.2; 74227.3; 58

 16; 240528595; 239792372; 736223; 36175.5; 1038; ; 271070; 270400; 209

 17; 240968369; 240236508; 731861; 35738.5; 1072; ; 43977.4; 44413.6; 34

 18; 244007174; 243281535; 725639; 35110.9; 1310; ; 303880; 304503; 238

 19; 244007174; 243281535; 725639; 35110.9; 1311; ; 0; 0; 1

 20; 245553570; 244822759; 730811; 35625.4; 1429; ; 154640; 154122; 118

 21; 245715343; 244988944; 726399; 35183.9; 1442; ; 16177.3; 16618.5; 13

 22; 246224733; 245495803; 728930; 35436.1; 1480; ; 50939; 50685.9; 38

 23; 247248904; 246514337; 734567; 35998; 1559; ; 102417; 101853; 79

 24; 247340304; 246604780; 735524; 36093.5; 1566; ; 9140; 9044.3; 7

 25; 247397627; 246667947; 729680; 35509; 1571; ; 5732.3; 6316.7; 5

 26; 247675045; 246943185; 731860; 35726.5; 1592; ; 27741.8; 27523.8; 21

 27; 248178453; 247442661; 735792; 36118.8; 1632; ; 50340.8; 49947.6; 40

 28; 250563275; 249825294; 737981; 36333.5; 1818; ; 238482; 238263; 186

 29; 251189456; 250463791; 725665; 35100.8; 1867; ; 62618.1; 63849.7; 49

 30; 251513049; 250787610; 725439; 35077.6; 1892; ; 32359.3; 32381.9; 25