TUTORIAL

Development of a framework to gather the data from mobile devices for the purpose of user activity / behavior classification

December 12, 2017

Department of CSIS
BITS Pilani

Team:

Devamalya Hazra [2016H1120169P]
Mohit Agarwal [2016H1120161P]
Rishabh Sharma [2016H1120154P]
Shubham Bansal [2016H1120157P]

Experiment setup

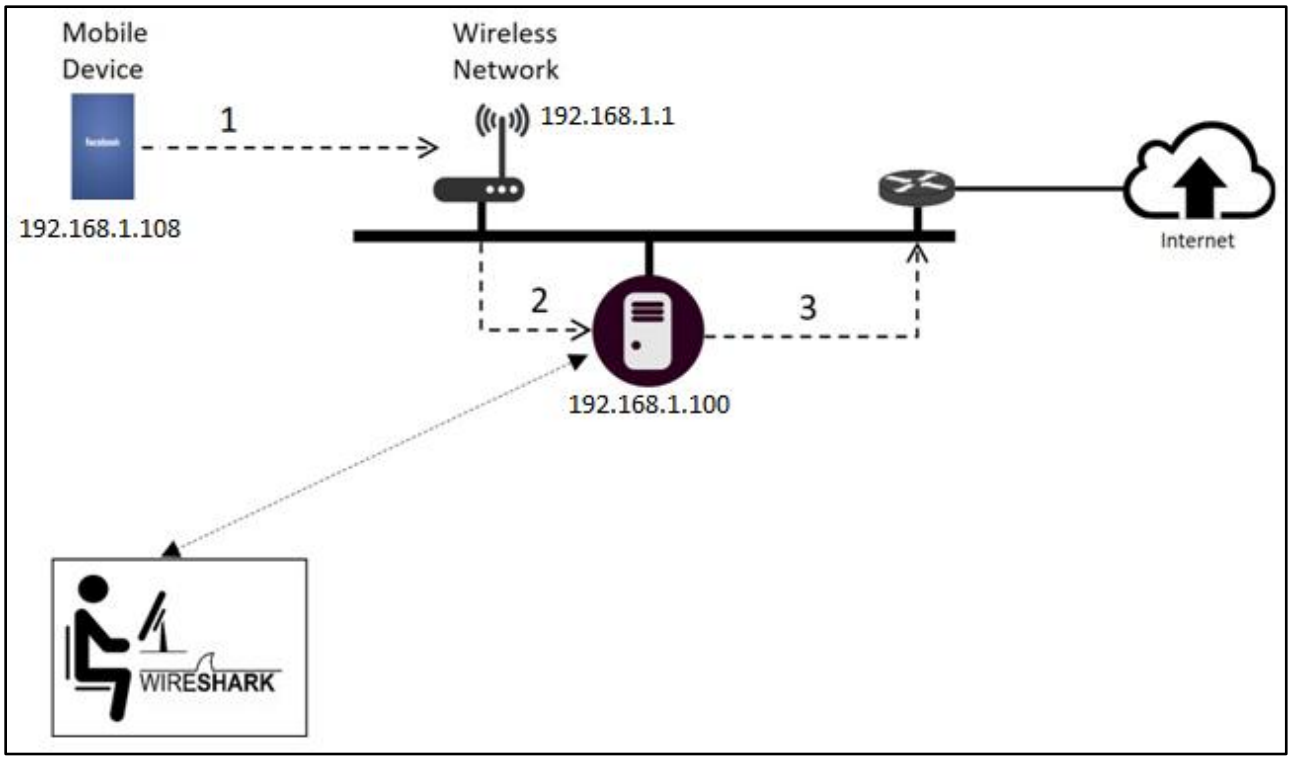


Figure 1: Experiment setup

Above figure shows the experimental setup for our tutorial.
In this tutorial, we are going to use a wifi router, an android phone and a laptop with ubuntu OS.

Router is configured with following DHCP ip range.
IP range: 192.168.1.1 to 192.168.1.254
Subnet: 192.168.1.0/24

So, all the devices - mobile phones, laptop and router are connected in same network. Whole mobile traffic which is generated from mobile devices is routed through laptop (proxy server) and that can be intercepted at this point.

We will start the tutorial with configuration of laptop and mobile devices. First we will setup proxy-server on ubuntu-laptop and then we will use this proxy on our phones for internet.

1. PROXY-SERVER Setup (TinyProxy)

We will use TinyProxy for proxy-server setup. Tinyproxy is a HTTP proxy server daemon for linux operating systems which is designed to be small and fast solution for proxy-server. For our prototype, this is viable solution.

1.>We can install it via apt-get command. Execute following command on terminal:

```
sudo apt-get install tinyproxy
```

2.>We need to configure ip-permissions so that proxy server can allow connections from android phones.

open **/etc/tinyproxy.conf** with root user or [sudo permission] in text editor. Scroll down in file where the default IP rules are mentioned.

Add the IP list (whitelist)

//Allow 127.0.0.1
//Allow 192.168.1.113 //
//Allow 192.168.1.110 //
Allow 192.168.1.0/24 // this will include whole network.

3.>Now, we can start the proxy Server (background process)

```
sudo /etc/init.d/tinyproxy restart
```

4.>Proxy server's maintains all logs in default location (/var/log/tinyproxy/tinyproxy.log)
We can see the logs related to connection-requests, requested urls, etc.

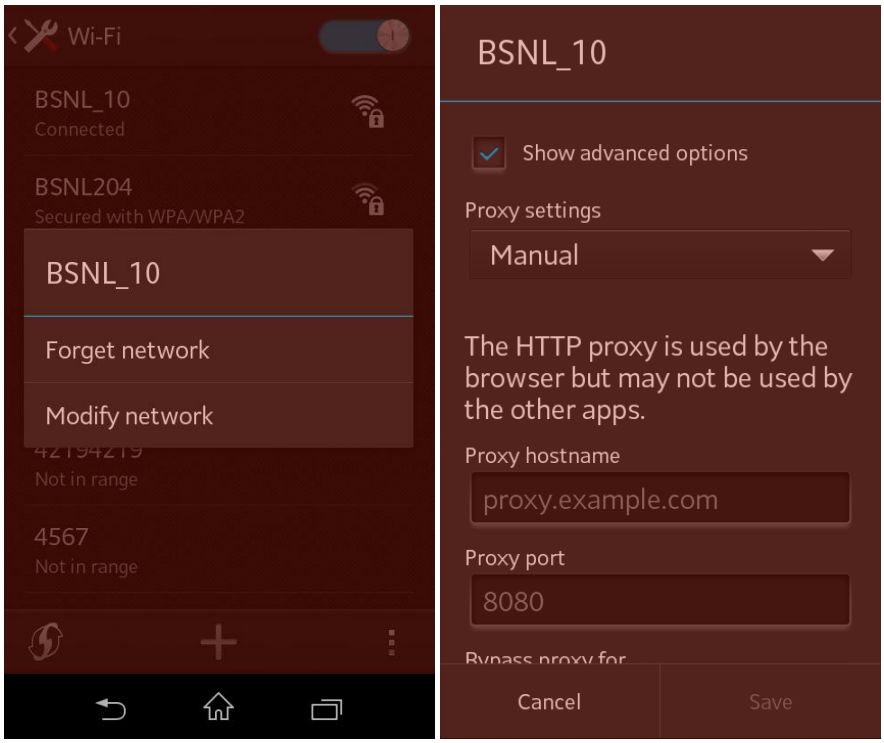
```
sudo cat /var/log/tinyproxy/tinyproxy.log
```

2. Android phone configuration

We will configure proxy and restrict background settings.

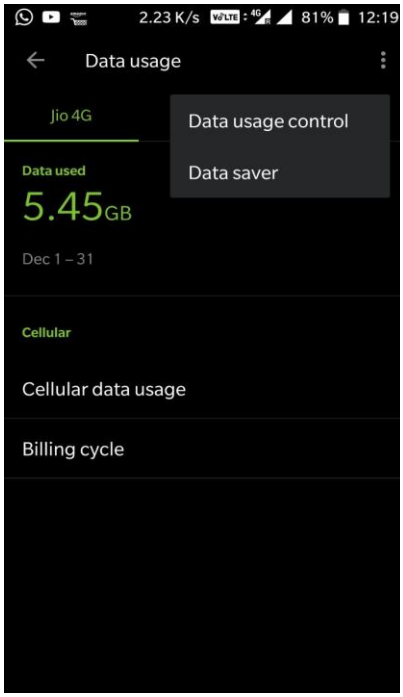
- **Proxy Settings**
  Open wifi from settings.
  Set proxy setting which is available under advanced options of wifi-connection:

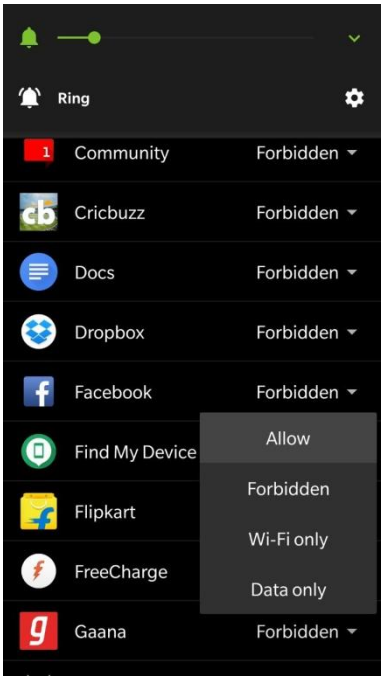
Proxy settings: Manual
Proxy hostname: 192.168.1.100
Proxy port: 8888

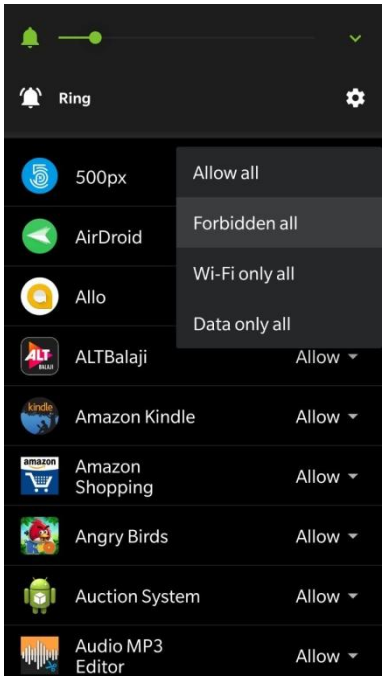


- **Restrict background**

  Go to **Settings => data usage => on the top right of the screen**







3. Wireshark (Packet capture)

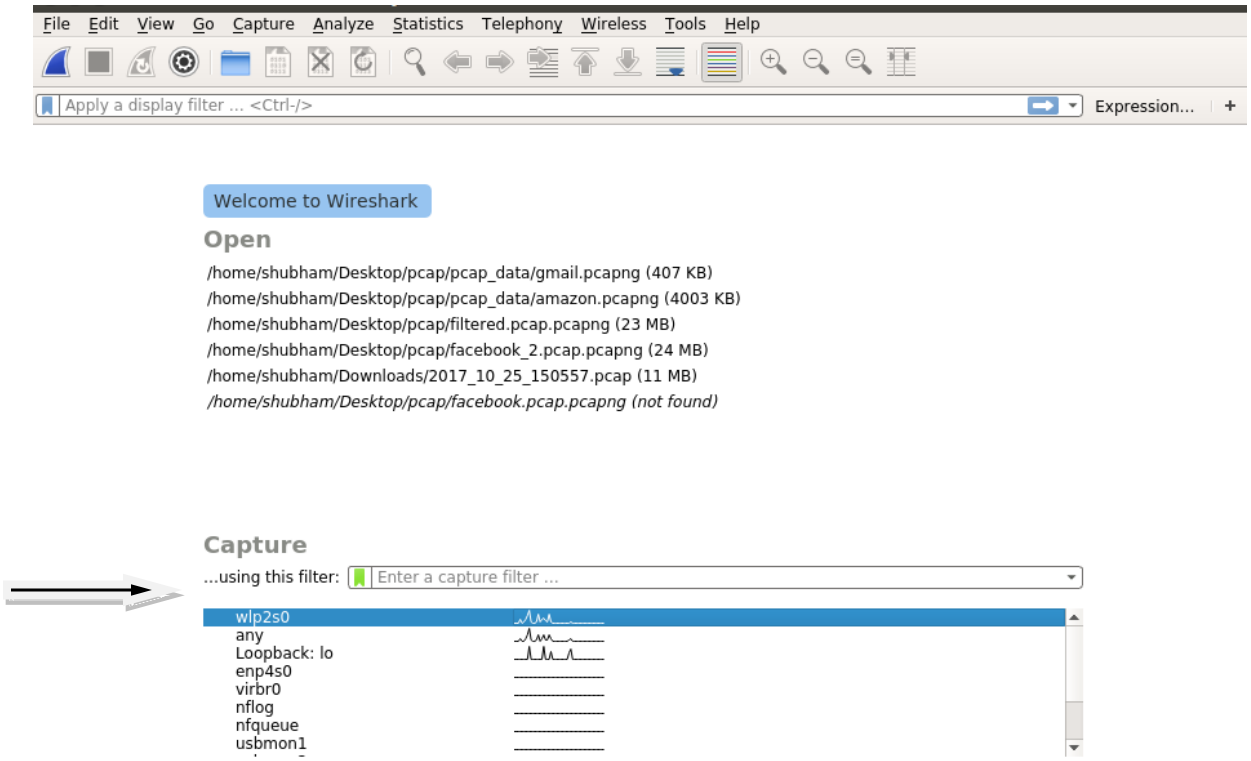
We have configured laptop with proxy-server and android phone with proxy setting and background data restrictions. Now. We are ready for the packet capturing part. We will use wireshark (Network sniffer tool).

Install Wireshark via apt-get.

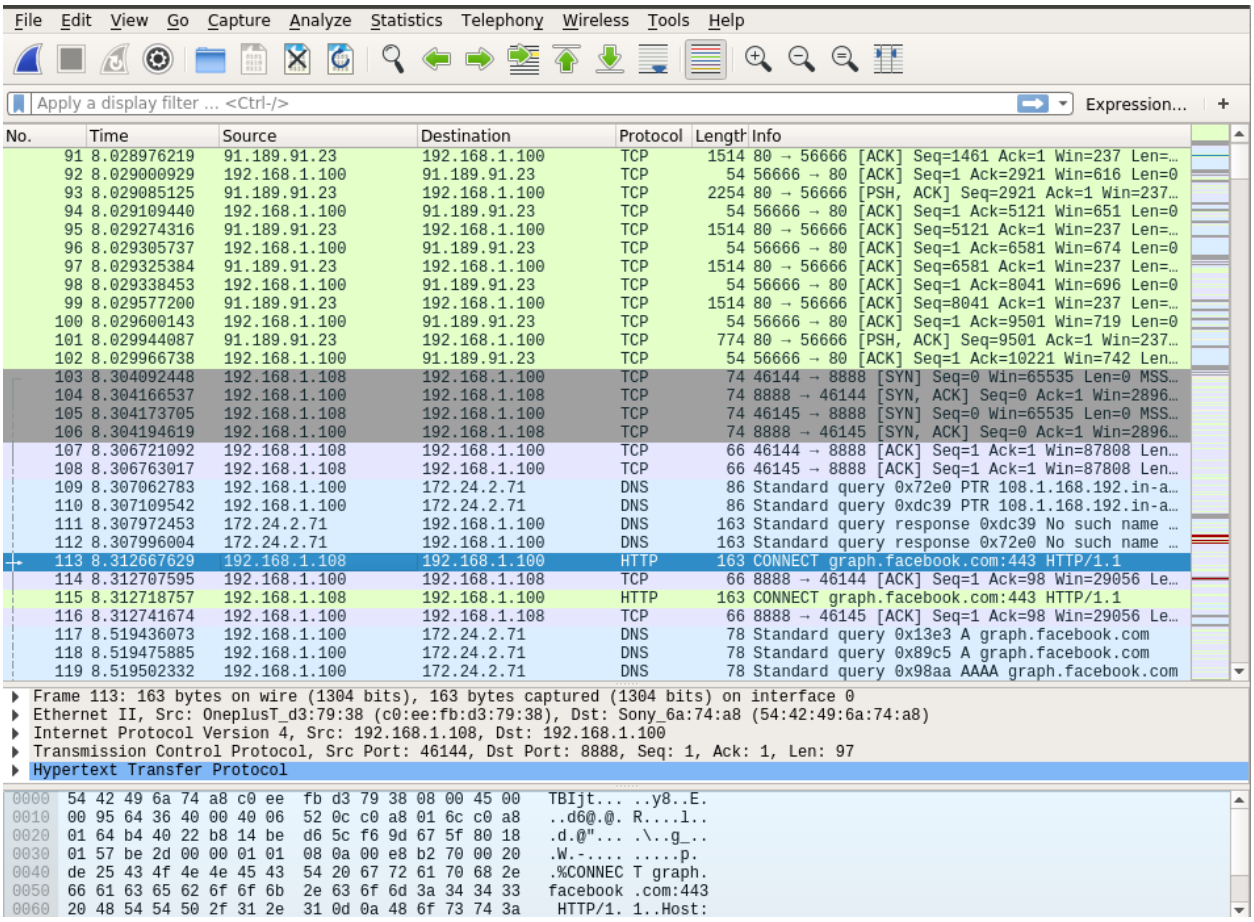
sudo apt-get *install wireshark*

Open wireshark with sudo permission.

sudo *wireshark*



Select wifi-interface (wlp2s0)



Now, in above picture we can see all the wifi network traffic. We can easily track packets related to the mobile application. In this case we are using facebook on phone and packet number 113 confirms the application (CONNECT graph.facebook.com).

This captured data contains all communication between android phone and facebook. It consists packets of http, dns, TLSv1.2, tcp, etc.

We will be capturing the n/w traffic for few minutes then we will stop and export data to pcap file. In next steps, we will use this pcap file for the analysis.

4. Packet Filteration

This is data pre-processing step. In this step, we will remove unwanted packets (like: non-IP, proxy-server background data)
Ex:

non-IP: SSDP, ARP
ubuntu repository data: us.archive.ubuntu.com, security.ubuntu.com,etc.
We will use following python code ({Project_dir}/**filter_packet_code.py**) for this step.

```
from scapy.allimport*


#reading from pcap file

packets = rdpcap('facebook_2.pcap.pcapng')

filtered_packets =  PacketList()


# Filter-Rules:

# 91.189.91.13, 91.189.91.14, 91.189.91.15, more ==> us.archive.ubuntu.com

# 91.189.88.161, 91.189.92.181, 91.189.92.200  ==> security.ubuntu.com

# 91.189.95.83 ==> ppa.launchpad.net

# 91.189.92.152 ==> extras.ubuntu.com


ignore_list
=['239.255.255.250','91.189.88.161','91.189.92.181','91.189.91.13','91.189.91.14','91.189.91.15','91.189
.95.83','91.189.92.152']


num =-1

for pk in packets:


num = num +1

ifnot pk.haslayer(IP):

print'packet without IP: ', num

continue


ifnot(pk['IP'].src in ignore_list or pk['IP'].dst in ignore_list):

filtered_packets.append(pk)
```

```
#writing to pcap file
wrpcap('filtered.pcap.pcapng',filtered_packets)

print'Packets count in original pcap =  ',len(packets)
print'Packets count in filtered pcap =  ',len(filtered_packets)
```

this will generate a filtered pcap file.

5. Feature extraction

Now, we will use above filtered pcap file for feature extraction.
We will use following python code ({Project_dir}/**flow_code.py**) for this step.

First we extract flow.

```
'''
@Parameter 1: packetList object
@Parameter 2: output pickleFile path, this is optional parameter. If valid path exists then function will return flowList by de-serialising the pickle file data

return: flowList which is list of list object..  ==> [ [srcIP, dstIP], [srcPort, dstPort], [packetIndex1, packetIndex2,.....]]
'''
def extractFlows(packets, outputPickleFile=None):

if outputPickleFile !=Noneand  os.path.exists(outputPickleFile):
fp =open('flows.pik',"rb")
flowList = pickle.load(fp)
fp.close()
```

```
return flowList


num =0
flowList =[]


for pk in packets:
if pk.haslayer(IP)and pk.haslayer(TCP):
src = pk['IP'].src
dst = pk['IP'].dst
sport = pk['IP'].sport
dport = pk['IP'].dport


#rules
#Ex: filter protocol, remove ack, threshold on minimum packets,



#aggregating packets into flow
iflen(flowList)==0:
flowList.append([[src,dst],[sport,dport],[num]])
else:
flag =False
for flow in flowList:
if src in flow[0]and dst in flow[0]and sport in flow[1]and dport in flow[1]:
flow[2].append(num)
flag =True
break
ifnot flag:
flowList.append([[src,dst],[sport,dport],[num]])
```

```
num +=1



#writing flowList into pickle

if outputPickleFile !=None:

fout =open(outputPickleFile,"wb")

pickle.dump(flowList, fout )

fout.close()


return flowList
```

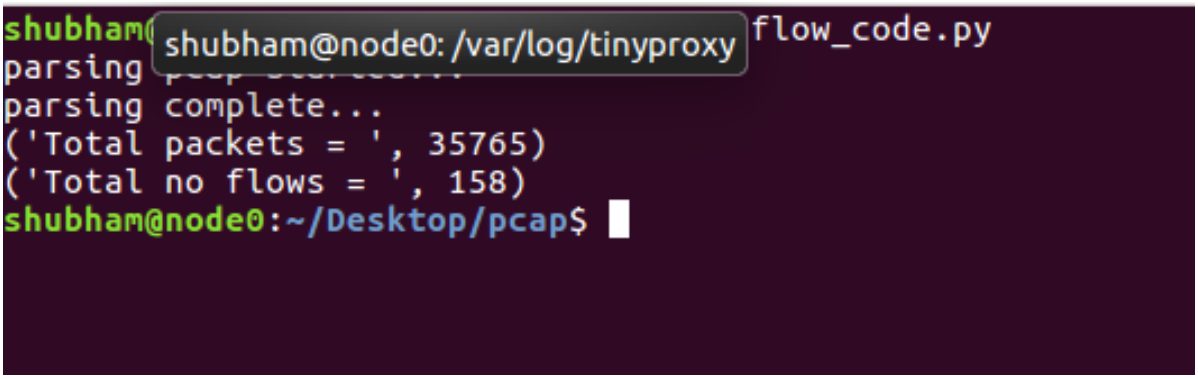
Now, we extract 22 features for each flow.

```
'''
@Parameter 1: packetList object

@Parameter 2: flowList which is list of list object..  ==> [ [srcIP, dstIP], [srcPort, dstPort], [packetIndex1, packetIndex2,.....]]


return: function return the feature vector which is list of numeric values

'''


def extractFeatures(packets, flow):


#creating featureVector list

featureVector =[]


#Adding features to feature vector


#1

totalPacketCount = feature_TotalPackets(packets, flow)
```

```
featureVector.append(totalPacketCount)


#2
avgPacketSize = feature_AveragePacketSize(packets, flow)
featureVector.append(avgPacketSize)



#3
minfpktl, maxfpktl, meanfpktl, stdfpktl, fpackets,  fbytes,  minfiat, maxfiat, meanfiat, stdfiat = DirectionFeatures(packets, flow,"Forward")
featureVector.append(minfpktl)
featureVector.append(maxfpktl)
featureVector.append(meanfpktl)
featureVector.append(stdfpktl)
featureVector.append(fpackets)
featureVector.append(fbytes)
featureVector.append(minfiat)
featureVector.append(maxfiat)
featureVector.append(meanfiat)
featureVector.append(stdfiat)

minbpktl, maxbpktl, meanbpktl, stdbpktl, bpackets,  bbytes,  minbiat, maxbiat, meanbiat, stdbiat = DirectionFeatures(packets, flow,"Backward")
featureVector.append(minbpktl)
featureVector.append(maxbpktl)
featureVector.append(meanbpktl)
featureVector.append(stdbpktl)
featureVector.append(bpackets)
featureVector.append(bbytes)
```

```
featureVector.append(minbiat)

featureVector.append(maxbiat)

featureVector.append(meanbiat)

featureVector.append(stdbiat)


return featureVector
```

Output:



>> **cat {Project_folder}/out_features.csv**

```
faceboook,26,156.115384615,52,298,168.0,121.028095912,15,2533,-1.93347574198,2.7658
faceboook,4,113.0,52,320,186.0,134.0,2,372,5.80230355263e-05,5.80230355263e-05,5.80
faceboook,4,100.5,52,270,161.0,109.0,2,322,3.65759730339e-05,3.65759730339e-05,3.65
faceboook,24,142.25,40,286,145.0,121.739182564,14,2036,-1.93458503199,2.76874491102
faceboook,14,98.9285714286,52,270,117.0,96.7699776347,7,820,4.4243991375e-05,1.5713
faceboook,14,106.071428571,52,320,131.0,119.340449375,7,920,2.47520208359e-05,1.474
faceboook,13,67.3076923077,40,221,77.0,62.0886463051,7,540,0.000900973021984,0.0640
faceboook,13,69.2307692308,40,221,80.0,63.9385866058,7,565,0.00076639598608,0.12792
faceboook,47,723.404255319,40,2960,1436.0,568.251628281,23,33040,-0.896489958048,2.
faceboook,18,101.666666667,40,467,111.0,122.716747023,10,1110,-0.748293097019,0.434
faceboook,16,102.8125,40,467,117.0,127.684071921,9,1058,-1.19041515601,0.3707019959
faceboook,16,87.25,40,455,97.0,127.580040236,9,877,-0.755388317049,0.433376551986,0
faceboook,13,98.1538461538,40,455,113.0,140.411639729,7,797,-1.19568071997,0.292060
faceboook,18,101.666666667,40,467,111.0,122.716747023,10,1110,-2.28213872403,0.6270
faceboook,16,102.8125,40,467,117.0,127.684071921,9,1058,-3.09156250399,0.6082511360
faceboook,1447,669.525224603,52,2948,407.0,675.16828114,789,321655,-3.93343553001,2
faceboook,21,98.2380952381,52,467,106.0,117.674435317,11,1174,-2.29226303399,0.8199
faceboook,16,87.25,40,455,97.0,127.580040236,9,877,-2.33087508005,0.425637515008,-0
faceboook,1275,742.249411765,40,14640,480.0,1384.6625431,642,308178,-4.10595768899,
```