

A Hands on SIP based VoIP Experiments on:Call
Establishment, Busy Lines, Call on Hold and Conference
Calling

RYMA NAITAMARA

7 juillet 2018

PART - 1

OBJECTIVE -

To learn the implementation of Voice over IP (VoIP) using Session Initiation Protocol (SIP) over Ad-Hoc Network.

HARDWARE SETUP -

For Phase 1 and 2 , we need two clients and one server machine (a Linux machine). Among a total of three machines, we are using a Linux machine as Server and two Windows machine as our clients.

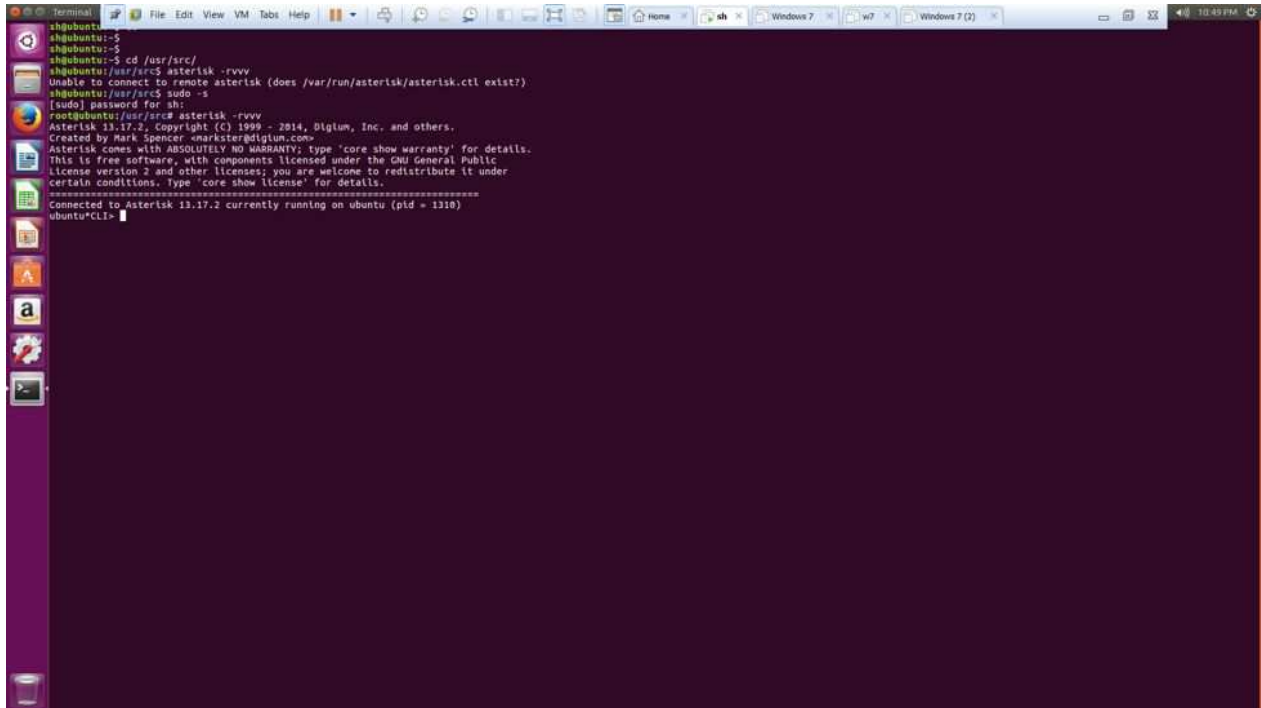
Server - The asterisk server is installed on the server machine with the sip.conf and extension.conf files already within them. These files are used for registration of sip clients on the server machine and the call set up.

For creating two clients, we added the following commands in the sip.conf files which we have downloaded along with the asterisk server. They are as below -

```
[general]
port=5060; Port to bind to (SIP is 5060)
bind addr = 10.42.0.1 = Asterisk server IP address allow =
Ulaw ; All all codecs
```

```
[100] username=100
type=friend
secret=password
Host=dynamic
context=from-sip
```

```
[200] username=200
type=friend
secret=password
hots=dynamic
context=from-sip
```

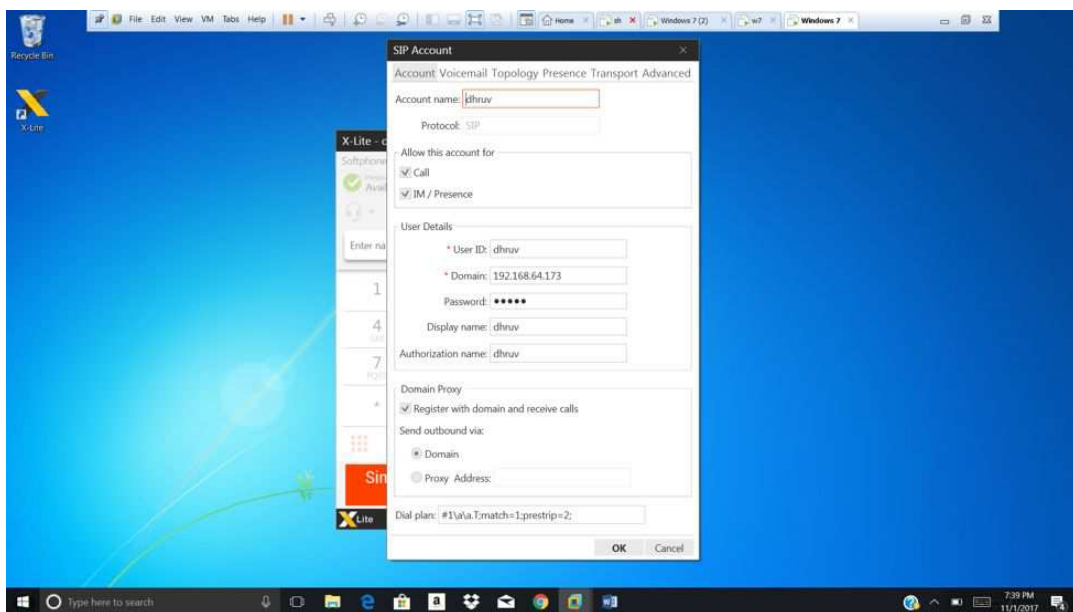
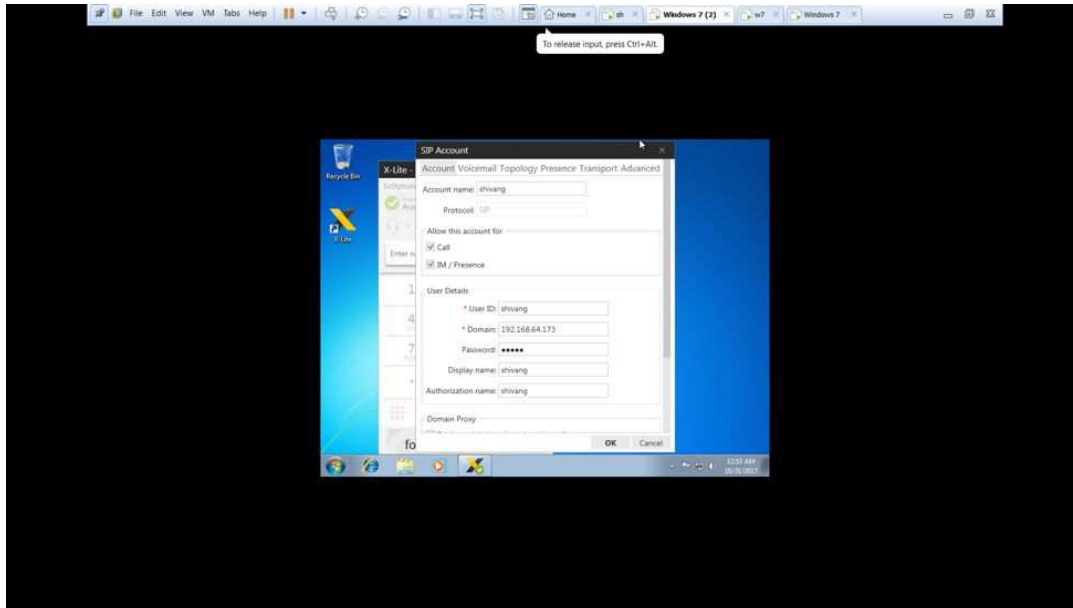


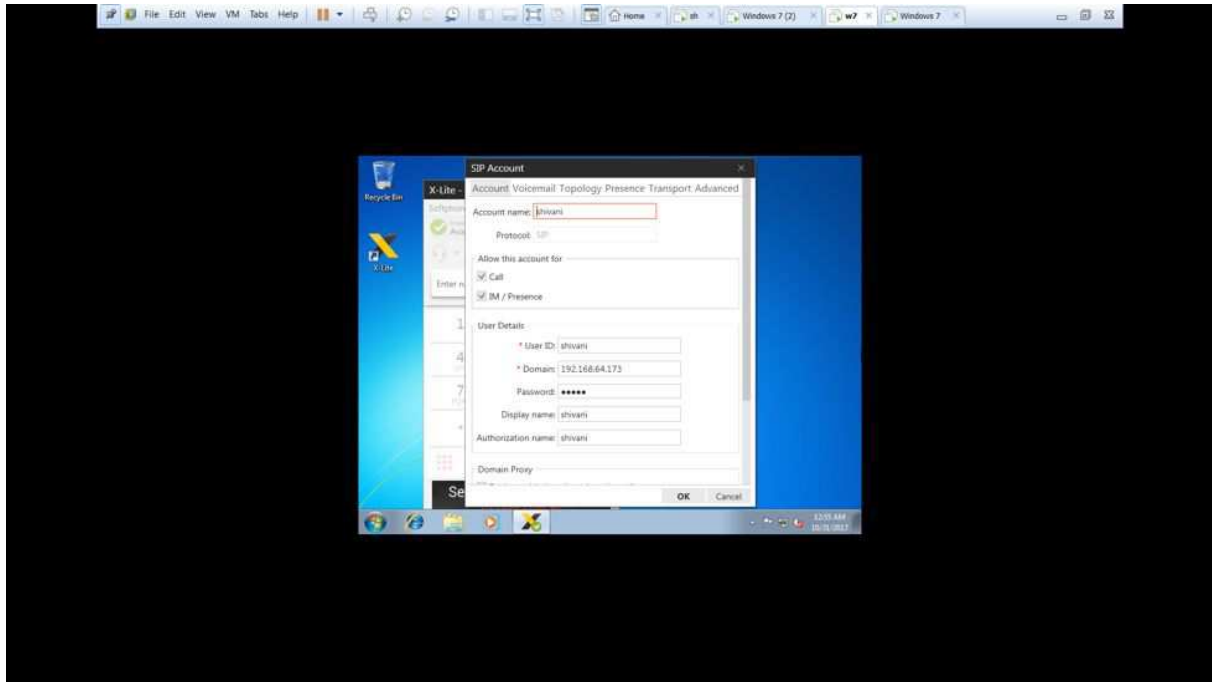
```
sh@ubuntu:~$ sudo apt-get install asterisk
sh@ubuntu:~$ cd /usr/src/
sh@ubuntu:~$ asterisk -rvvv
Unable to connect to remote asterisk (does /var/run/asterisk/asterisk.ctl exist?)
sh@ubuntu:~$ sudo -s
[sudo] password for sh:
root@ubuntu:~# asterisk -rvvv
Asterisk 13.17.2, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=====
Connected to Asterisk 13.17.2 currently running on ubuntu (pid = 1310)
ubuntu*CLI>
```

Clients - The X-Lite Softphone software is installed on the to client machines, which is VoIP smartphone and uses SIP (session initiation protocol). One client is assigned with an user ID of 100 and the other is done the same with ID 200.

We can check the IP addresses on all the machines, of both server and the two clients. Also the IP addresses on all the machines, will be in the same network.

These are the images for the clients -





PHASE - 1 : Establishment and Analysis of calls between two clients

Call Establishment:

For establishing a call between our two sip clients with user IDs 100 and 200, we modified the extension.conf, which we downloaded along with the asterisk software. Here the user name will be the name as displayed on the X-Lite softphone when we connect it with the server. We are required to add the following commands in the extension. Conf file of the asterisk server.

```
[from-sip]
exten=> 100, 1, Dial (SIP/100,20)
exten=> 100, 2, Hangup
exten=> 100, 1, Dial (SIP/200,20) exten=> 2-1, 2, Hangup
```

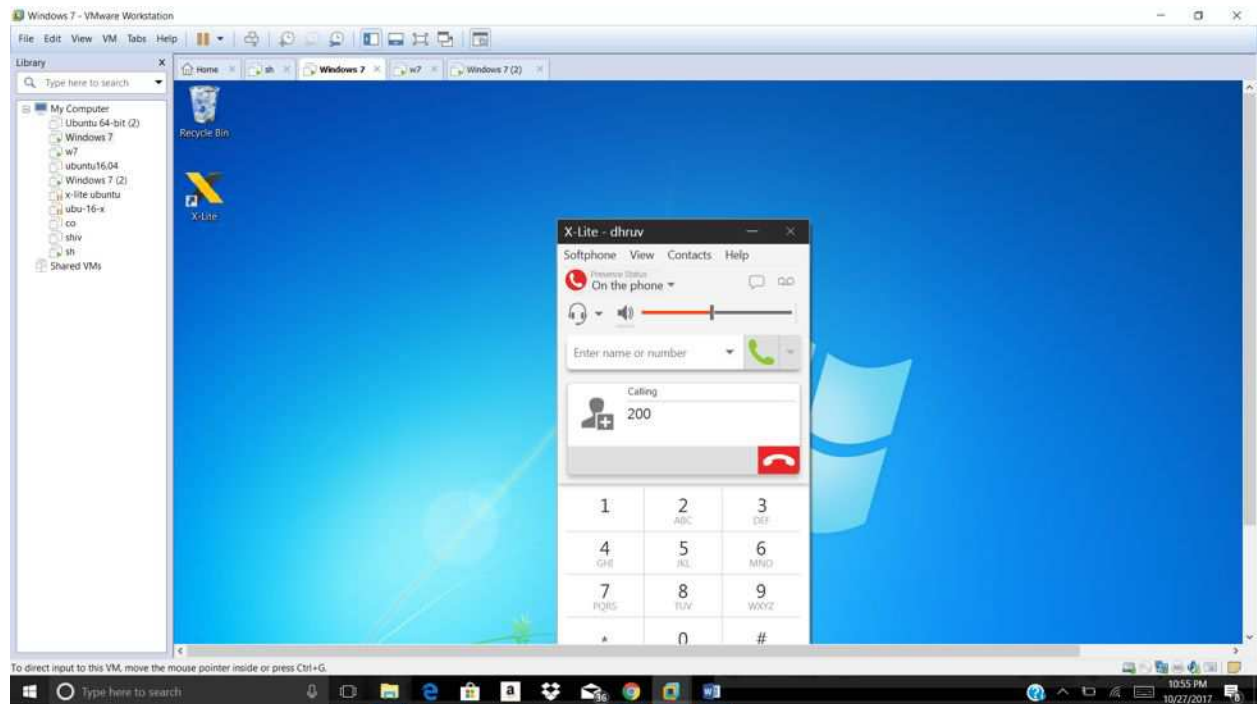
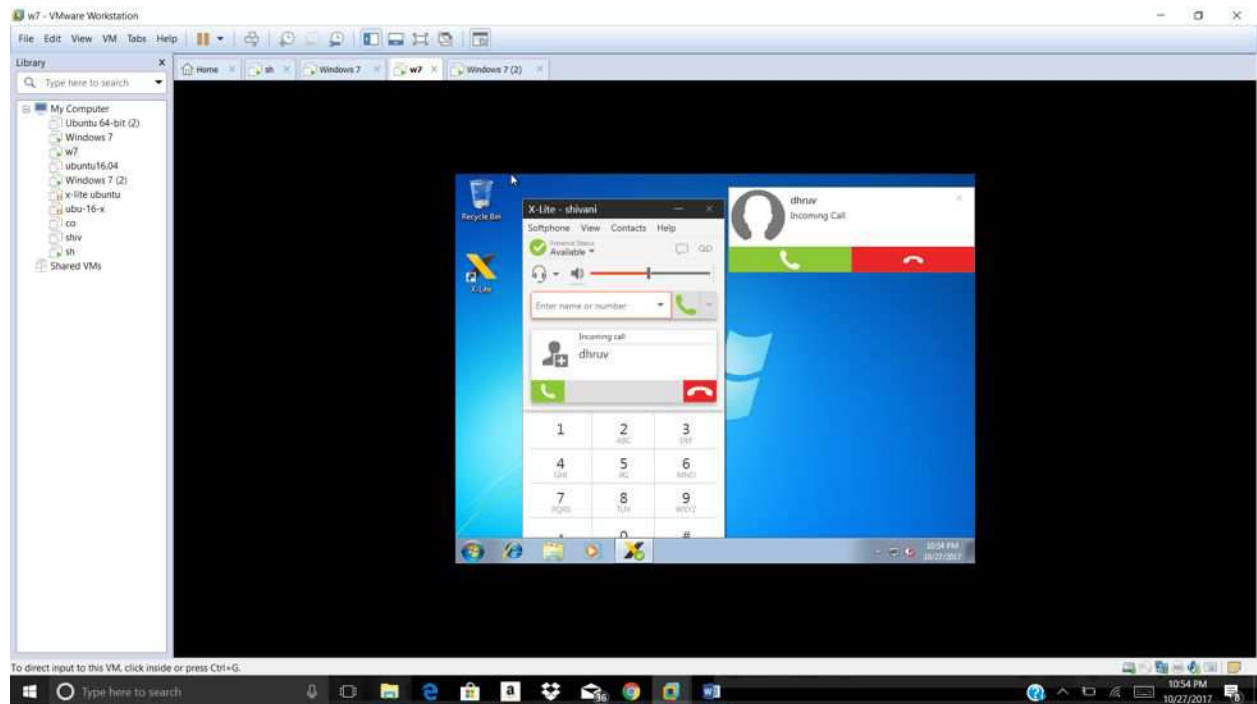
After modified the extension file, we reloaded the asterisk server to save the changes which we made. This is done by typing the reload command. We now call one client to another by dialing the user id and also capture the packets through Wireshark. Here we are calling from client with id 100 to 200.

Here in our project, we have,

<u>Name on X-Lite Phone</u>	<u>ID (in our project)</u>	<u>ID (in original file)</u>
Shivang	100	2000
Shivani	200	2010
Dhruv	300	2020

Experiment Result : Call was successfully established between two clients.

Calling 200 from 300-



```

No.      Time      Source      Destination      Protocol      Length      Info
1      21.8.137151000    192.168.84.173    192.168.84.180    TCP          60      Request: TWINTE sip-shivang192.168.84.190:61148;instance=f32c4f92180466, in-dialing
2      44.6.131047462    192.168.84.155    192.168.84.173    SIP          331      Status: 300 Trying
3      56.8.138880618    192.168.84.155    192.168.84.173    SIP          73      Status: 200 OK
4      57.6.138881742    192.168.84.173    192.168.84.155    SIP          468      Request: ACK sip-shivang192.168.84.155:51914;instance=90b623f2a005d39a
5      67.6.145727906    192.168.84.173    192.168.84.180    SIP/SDP     594      Request: TWINTE sip-shivang192.168.84.190:61148;instance=f32c4f92180466, in-dialing
6      69.4.400552618    192.168.84.155    192.168.84.173    SIP          521      Status: 200 OK
7      70.6.488084544    192.168.84.173    192.168.84.180    SIP          508      Request: ACK sip-shivang192.168.84.155:61148;instance=f32c4f92180466
8      132.2.398134848    192.168.84.173    192.168.84.155    SIP/SDP     594      Request: TWINTE sip-shivang192.168.84.173
9      150.2.388806472    192.168.84.173    192.168.84.155    SIP          611      Status: 401 Unauthorized
10     180.2.388806472    192.168.84.155    192.168.84.173    SIP          380      Request: ACK sip-1000212.168.84.173
11     202.12.378612614    192.168.84.155    192.168.84.173    SIP/SDP     594      Request: TWINTE sip-1000212.168.84.173
12     242.2.382112510    192.168.84.173    192.168.84.155    SIP          554      Status: 300 Trying
13     257.12.464549187    192.168.84.155    192.168.84.180    SIP/SDP     594      Request: TWINTE sip-shivang192.168.84.190:61148;instance=f32c4f92180466
14     268.12.462595616    192.168.84.173    192.168.84.180    SIP/SDP     594      Request: TWINTE sip-shivang192.168.84.190:61148;instance=f32c4f92180466
15     278.12.592758077    192.168.84.155    192.168.84.173    SIP          521      Status: 200 OK
16     282.12.617918053    192.168.84.180    192.168.84.173    SIP          520      Status: 180 Ringing
17     322.12.6180966    192.168.84.180    192.168.84.173    SIP          520      Status: 180 Ringing
18     387.12.621133911    192.168.84.173    192.168.84.155    SIP          570      Status: 300 Ringing
19     1363.19.395694568    192.168.84.180    192.168.84.173    SIP/SDP     597      Request: TWINTE sip-shivang192.168.84.173:5060, in-dialing
20     1384.19.391458677    192.168.84.173    192.168.84.180    SIP          600      Status: 300 Trying
21     1397.19.397744607    192.168.84.173    192.168.84.180    SIP          927      Status: 200 OK
22     1398.19.398134809    192.168.84.173    192.168.84.155    SIP          468      Request: TWINTE sip-shivang192.168.84.155:51914;instance=90b623f2a005d39a, in-dialing
23     1399.19.400169503    192.168.84.180    192.168.84.173    SIP          825      Status: 200 OK
24     1371.19.401407172    192.168.84.180    192.168.84.180    SIP          508      Request: ACK sip-shivang192.168.84.190:61148;instance=f32c4f92180466
25     1372.19.410945134    192.168.84.173    192.168.84.155    SIP/SDP     594      Request: 200 OK
26     1375.19.435136166    192.168.84.180    192.168.84.173    SIP          543      Request: ACK sip-shivang192.168.84.173:5060
27     1386.19.463340667    192.168.84.155    192.168.84.180    SIP          562      Request: TWINTE sip-shivang192.168.84.155:51914;instance=f32c4f92180466, in-dialing
28     1387.19.476434489    192.168.84.155    192.168.84.173    SIP          583      Request: ACK sip-1000212.168.84.173
29     1388.19.476830677    192.168.84.173    192.168.84.155    SIP          873      Request: TWINTE sip-shivang192.168.84.155:51716;instance=537b6e1550bdc, in-dialing
30     1389.19.482326868    192.168.84.155    192.168.84.173    SIP/SDP     793      Status: 200 OK
31     1389.19.482326868    192.168.84.173    192.168.84.155    SIP          468      Request: ACK sip-shivang192.168.84.155:51914;instance=537b6e1550bdc, in-dialing
32     1410.19.564550972    192.168.84.173    192.168.84.180    SIP/SDP     594      Request: TWINTE sip-shivang192.168.84.155:51914;instance=f32c4f92180466, in-dialing
33     1410.19.574540167    192.168.84.173    192.168.84.155    SIP/SDP     873      Request: TWINTE sip-shivang192.168.84.155:51716;instance=537b6e1550bdc, in-dialing
34     1410.19.645439567    192.168.84.155    192.168.84.173    SIP          335      Status: 300 Trying
35     1410.19.645390509    192.168.84.155    192.168.84.173    SIP/SDP     793      Status: 200 OK
36     1447.19.648791726    192.168.84.173    192.168.84.155    SIP          472      Request: ACK sip-shivang192.168.84.155:51716;instance=537b6e1550bdc, in-dialing
37     1474.19.760330814    192.168.84.173    192.168.84.180    SIP/SDP     594      Request: TWINTE sip-shivang192.168.84.190:61148;instance=f32c4f92180466, in-dialing
38     1500.19.812330623    192.168.84.155    192.168.84.173    SIP          565      Status: 200 OK
39     1514.19.835448462    192.168.84.180    192.168.84.173    SIP/SDP     825      Status: 200 OK
40     1515.19.836151443    192.168.84.180    192.168.84.173    SIP          520      Status: 200 OK
41     1518.19.836398246    192.168.84.173    192.168.84.180    SIP          512      Request: ACK sip-shivang192.168.84.190:61148;instance=f32c4f92180466
42     1519.19.837154688    192.168.84.173    192.168.84.180    SIP          512      Request: ACK sip-shivang192.168.84.190:61148;instance=f32c4f92180466
43     1520.19.182326122    192.168.84.155    192.168.84.180    SIP          607      Request: OPTIONS sip-shivang192.168.84.190:61148;instance=f32c4f92180466
44     1599.19.194511210    192.168.84.180    192.168.84.173    SIP          824      Status: 200 OK
45     1599.19.195650087    192.168.84.173    192.168.84.155    SIP          607      Request: OPTIONS sip-shivang192.168.84.155:51716;instance=537b6e1550bdc, in-dialing
46     1614.19.322580548    192.168.84.155    192.168.84.173    SIP          624      Status: 200 OK
47     1614.19
```


PHASE-2 : Busy User

In this mode of operation, one user tries to call the other user, but the other user is busy. The time duration for which the user wants to set the busy mode is sent as an argument in the extension.conf file.

Steps -

1. Configure the two SIP users in “sip.conf” file as done in Phase 1 of the experiment.
2. Configure the extension.conf file for the 2 SIP users as given below -

[from-sip]

exten=> 100,1, Dial(SIP/100,20)

exten=> 100,2, Hangup exten => 200,1, Answer() exten => 200,2, Busy (10)

exten=> 200,3, Hangup

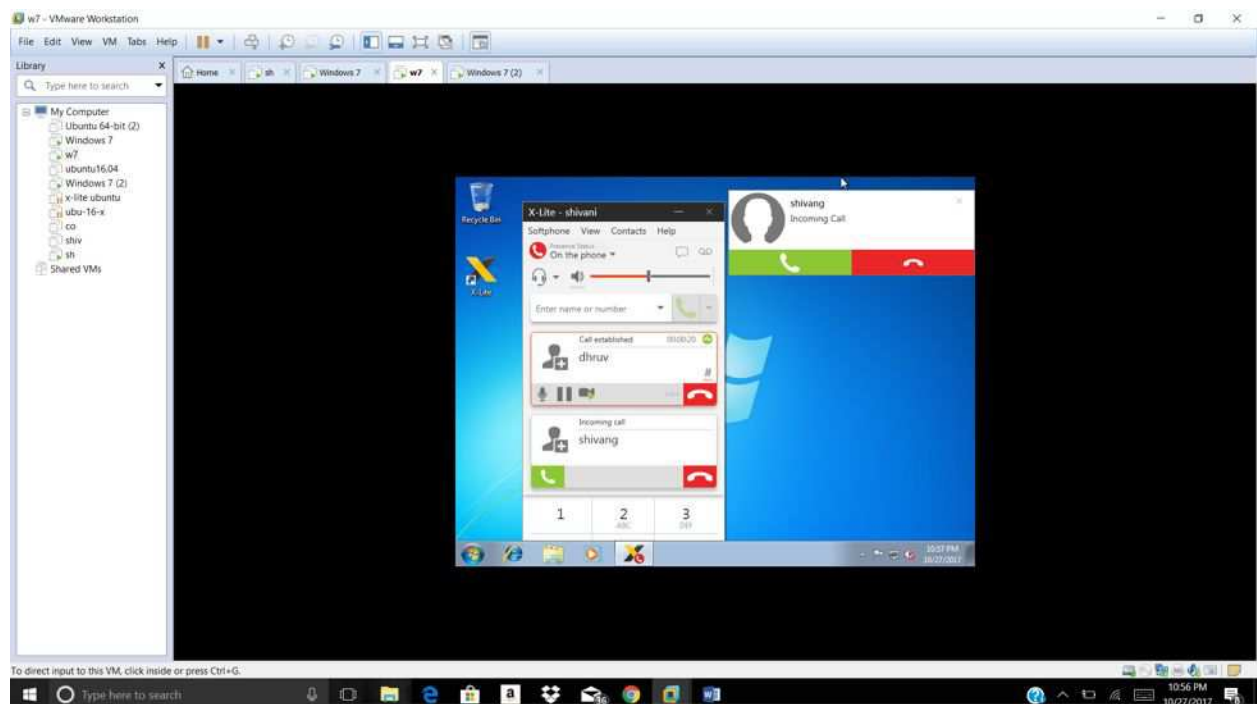
Now, here in this one client with ID 200 is made busy with the busy period of the time mentioned in the brackets; here in our case we have considered the time to be busy as of 10 seconds.

exten=> 200,1, Answer

exten=> 200,2, Busy(10) exten=>

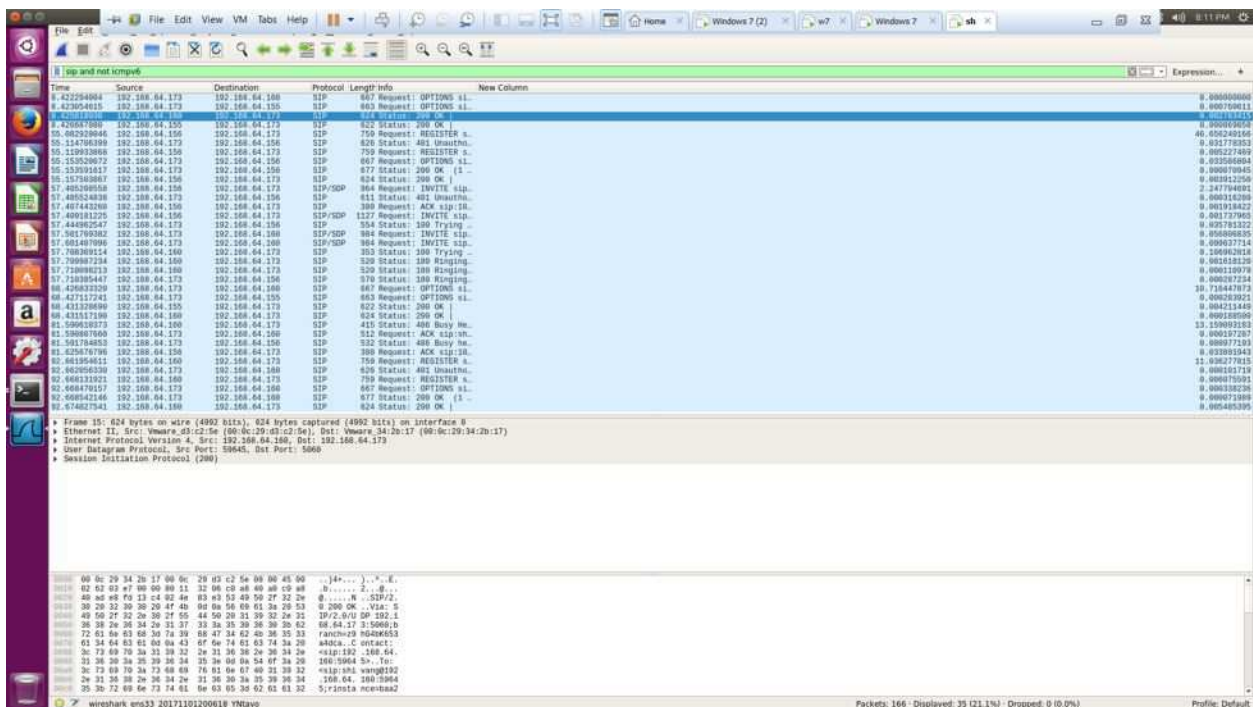
200,3, Hangup

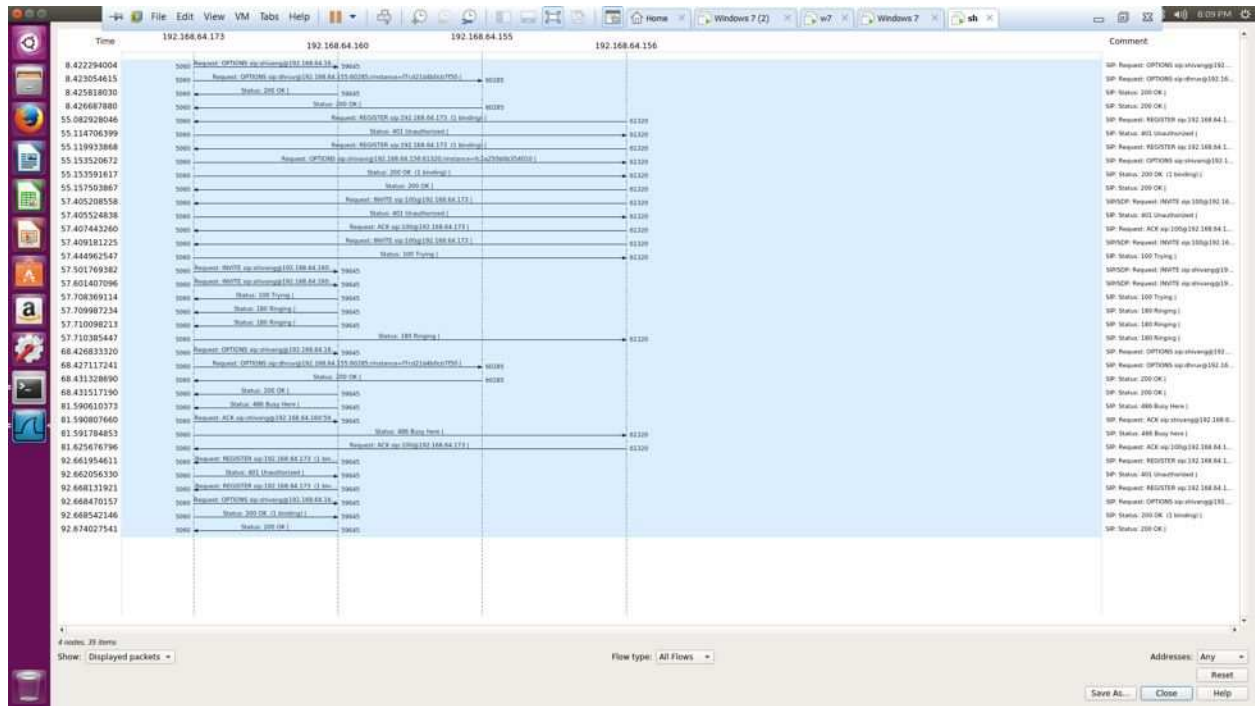
200=> Busy



```
sh@ubuntu:~$ cd /usr/src/
sh@ubuntu:~$ asterisk -rvvv
Unable to connect to remote asterisk (does /var/run/asterisk/asteriskctl exist?)
sh@ubuntu:~$ sudo -s
[sudo] password for sh:
root@ubuntu:~# asterisk -rvvv
Asterisk 13.17.2, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=====
Connected to Asterisk 13.17.2 currently running on ubuntu (pid = 1310)
-- Registered SIP 'dhruv' at 192.168.64.155:50348
-- Unregistered SIP 'dhruv'
-- Registered SIP 'dhruv' at 192.168.64.155:50348
[Oct 27 22:51:49] NOTICE[1404]: chan_sip.c:24586 handle_request_subscribe: Received SIP subscribe for peer without mailbox: dhruv
-- Using SIP RTP cos mark 5
-- Executing [100phones:1] NoOp("SIP/dhruv-00000000", "Call for shivang") in new stack
-- Executing [100phones:2] Dial("SIP/dhruv-00000000", "SIP/shivang") in new stack
[Oct 27 22:51:51] WARNING[2285][c-00000000]: app_dial.c:2125 dial_exec_full: Unable to create channel of type 'SIP' (cause 28 - Subscriber absent)
-- Everyone is busy/congested at this time (1:0/0/1)
-- Executing [100phones:3] Hangup("SIP/dhruv-00000000", "") in new stack
-- Span extension (phones, 100, 2) exited non-zero on 'SIP/dhruv-00000000'
-- Registered SIP 'shivang' at 192.168.64.160:61044
[Oct 27 22:53:58] NOTICE[1404]: chan_sip.c:24586 handle_request_subscribe: Peer 'shivang' is now Reachable. (8ms / 2000ms)
-- Unregistered SIP 'shivang'
-- Registered SIP 'shivang' at 192.168.64.160:61044
[Oct 27 22:54:00] NOTICE[1404]: chan_sip.c:28404 handle_request_subscribe: Received SIP subscribe for peer without mailbox: shivang
-- Registered SIP 'shivant' at 192.168.64.156:61475
[Oct 27 22:54:02] NOTICE[1404]: chan_sip.c:24586 handle_request_subscribe: Peer 'shivant' is now Reachable. (8ms / 2000ms)
-- Unregistered SIP 'shivant'
-- Registered SIP 'shivant' at 192.168.64.156:61475
[Oct 27 22:54:05] NOTICE[1404]: chan_sip.c:28404 handle_request_subscribe: Received SIP subscribe for peer without mailbox: shivant
-- Using SIP RTP cos mark 5
-- Executing [100phones:1] NoOp("SIP/dhruv-00000001", "Call for shivang") in new stack
-- Executing [100phones:2] Dial("SIP/dhruv-00000001", "SIP/shivang") in new stack
-- Using SIP RTP cos mark 5
-- Called SIP/shivang
-- SIP/shivang-00000002 is ringing
-- Span extension (phones, 100, 2) exited non-zero on 'SIP/dhruv-00000001'
-- Using SIP RTP cos mark 5
-- Executing [200phones:1] NoOp("SIP/dhruv-00000003", "Call for shivant") in new stack
-- Executing [200phones:2] Dial("SIP/dhruv-00000003", "SIP/shivant") in new stack
-- Using SIP RTP cos mark 5
-- Called SIP/shivant
-- SIP/shivant-00000004 is ringing
-- SIP/shivant-00000004 is ringing
-- Got SIP response 480 "Busy Here" back from 192.168.64.156:61475
-- SIP/shivant-00000004 is busy
-- Everyone is busy/congested at this time (1:1/0/0)
-- Executing [200phones:3] Hangup("SIP/dhruv-00000003", "") in new stack
-- Span extension (phones, 200, 2) exited non-zero on 'SIP/dhruv-00000003'
-- Using SIP RTP cos mark 5
-- Executing [200phones:1] NoOp("SIP/shivang-00000005", "Call for shivant") in new stack
-- Executing [200phones:2] Dial("SIP/shivang-00000005", "SIP/shivant") in new stack
-- Using SIP RTP cos mark 5
```

We can see in the following Wireshark capture, the of user being BUSY.
From 100-





PHASE-3: Call on Hold

Here the third client, with the client ID 300, calls to the user with ID 100, which is already in a connection with user 200. User 100 then keeps the 200 user on HOLD and establishes a call with ID 300. After completing the call with 300, he (ID#100) continues the call with user 200.

For this part, we register the third client (ID#300) in sip.conf file as described below -

[300]

username=300

type=friend

secret=password

host=dynamic

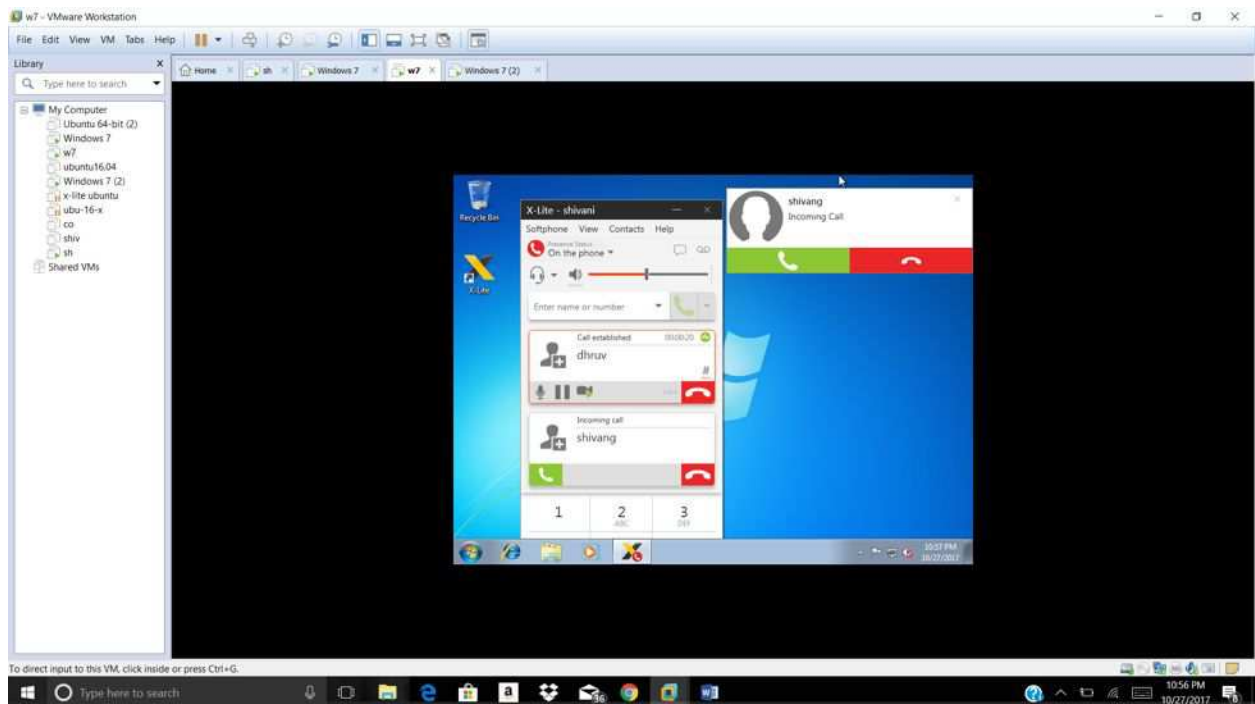
context =from-sip

Also, since we want to establish the call with user (ID#300) we have to add the following commands in the extension.conf file -

Exten => 300,1,Dial (SIP/300,20)

Exten => 300,2,Hangup

In this image we can see that user 300 is calling user 100 and then the user 100 holds call of user 200 and continues call with 100 once it gets completed.



The screenshot shows a VMware Workstation window titled 'w7 - VMware Workstation'. The main area displays a Windows 7 virtual machine named 'w7'. The desktop background is blue with a large white 'X' logo. An application window titled 'X-Lite - shivani' is open, showing a softphone interface. The interface includes a menu bar (Softphone, View, Contacts, Help), a status bar (On the phone), a volume slider, and a call log. The call log shows a call to 'shivang' (0000007) that is 'Call established'. The VMware interface also shows a library on the left with various VMs and a taskbar at the bottom with various icons.

```
root@ubuntu:/usr# ./File Edit View VM Tabs Help
[Oct 27 22:54:00] NOTICE[1404]: chan_sip.c:28404 handle_request_subscribe: Received SIP subscribe for peer without mailbox: shivang
Registered SIP 'shivang' at 192.168.64.150:61044
[Oct 27 22:54:02] NOTICE[1404]: chan_sip.c:24566 handle_response_peerpoke: Peer 'shivant' is now Reachable. (6ms / 2000ms)
Unregistered SIP 'shivant'
Registered SIP 'shivant' at 192.168.64.150:61475
[Oct 27 22:54:03] NOTICE[1404]: chan_sip.c:28404 handle_request_subscribe: Received SIP subscribe for peer without mailbox: shivant
Using SIP RTP CoS mark 5
Executing [100@phones:1] NoOp("SIP/dhruv-00000001", "Call for shivang") in new stack
Executing [100@phones:2] Dial("SIP/dhruv-00000001", "SIP/shivang") in new stack
Using SIP RTP CoS mark 5
Called SIP/shivang
SIP/shivang-00000002 is ringing
Spawn extension (phones, 100, 2) exited non-zero on 'SIP/dhruv-00000001'
Using SIP RTP CoS mark 5
Executing [200@phones:1] NoOp("SIP/dhruv-00000003", "call for shivant") in new stack
Executing [200@phones:2] Dial("SIP/dhruv-00000003", "SIP/shivant") in new stack
Using SIP RTP CoS mark 5
Called SIP/shivant
SIP/shivant-00000004 is ringing
SIP/shivant-00000004 is ringing
Got SIP response 486 "Busy here" back from 192.168.64.150:61475
SIP/shivant-00000004 is busy
Everyone is busy/congested at this time (11/0/0)
Executing [200@phones:3] Hangup("SIP/dhruv-00000003", "") in new stack
Spawn extension (phones, 200, 3) exited non-zero on 'SIP/dhruv-00000003'
Using SIP RTP CoS mark 5
Executing [200@phones:1] NoOp("SIP/shivang-00000005", "call for shivant") in new stack
Executing [200@phones:2] Dial("SIP/dhruv-00000005", "SIP/shivant") in new stack
Using SIP RTP CoS mark 5
Called SIP/shivant
SIP/shivant-00000006 is ringing
SIP/shivant-00000006 is ringing
Got SIP response 486 "Busy here" back from 192.168.64.150:61475
SIP/shivant-00000006 is busy
Everyone is busy/congested at this time (11/0/0)
Executing [200@phones:3] Hangup("SIP/shivang-00000005", "") in new stack
Spawn extension (phones, 200, 3) exited non-zero on 'SIP/shivang-00000005'
Using SIP RTP CoS mark 5
Executing [200@phones:1] NoOp("SIP/dhruv-00000007", "call for shivant") in new stack
Executing [200@phones:2] Dial("SIP/dhruv-00000007", "SIP/shivant") in new stack
Using SIP RTP CoS mark 5
Called SIP/shivant
SIP/shivant-00000008 is ringing
SIP/shivant-00000008 is ringing
SIP/shivant-00000008 answered SIP/dhruv-00000007
Channel SIP/shivant-00000008 joined 'simple_bridge' basic-bridge <9908b078-60c4-4fd6-b2e6-8db3bc787cde>
Channel SIP/dhruv-00000007 joined 'simple_bridge' basic-bridge <9908b078-60c4-4fd6-b2e6-8db3bc787cde>
Using SIP RTP CoS mark 5
Executing [200@phones:1] NoOp("SIP/shivang-00000009", "call for shivant") in new stack
Executing [200@phones:2] Dial("SIP/shivang-00000009", "SIP/shivant") in new stack
Using SIP RTP CoS mark 5
Called SIP/shivant
SIP/shivant-00000009 is ringing
SIP/shivant-00000009 is ringing
SIP/shivant-00000009 is ringing
Started music on hold, class 'default', on channel 'SIP/dhruv-00000007'
SIP/shivant-00000009 answered SIP/shivang-00000009
Channel SIP/shivant-00000009 joined 'simple_bridge' basic-bridge <5b5444d-0fe4-44bb-9642-721fa49e61f>
Channel SIP/shivang-00000009 joined 'simple_bridge' basic-bridge <5b5444d-0fe4-44bb-9642-721fa49e61f>
Stopped music on hold id SIP/dhruv-00000007
```


Packets: 8325 · Displayed: 86 (1.0%) Profile: Default

Packets: 6325 · Displayed: 86 (1.0%) Profile: Default

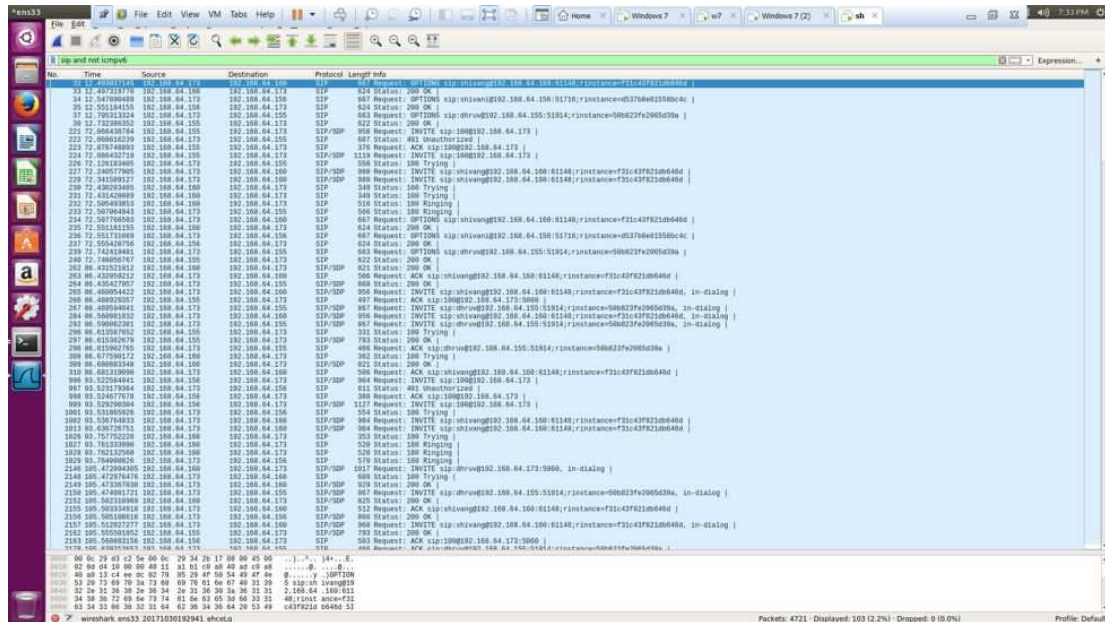
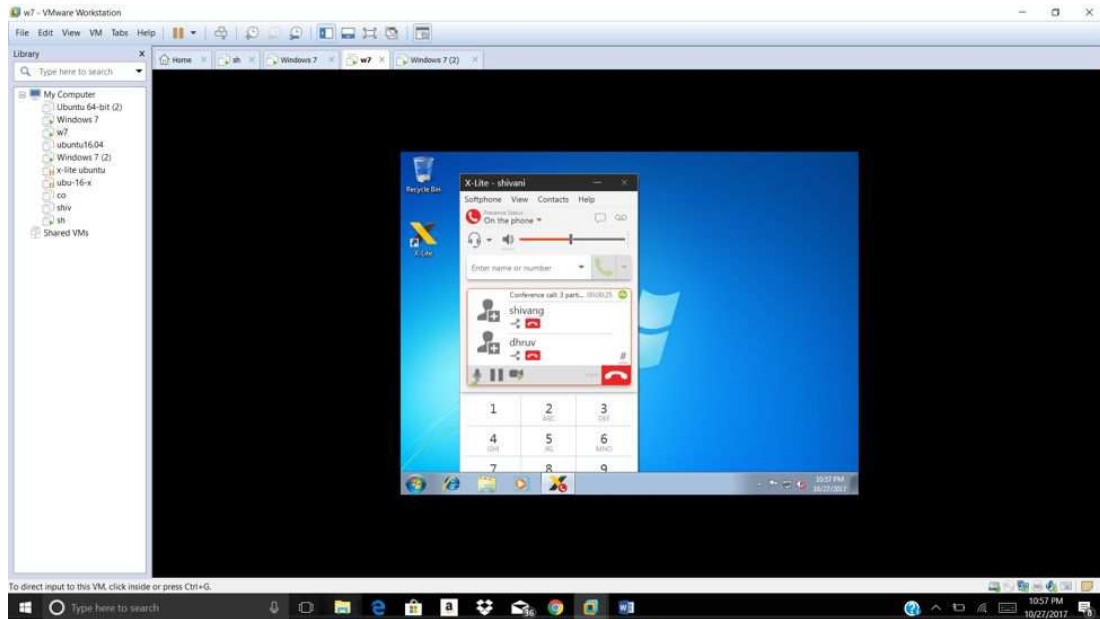
[illegible][illegible]

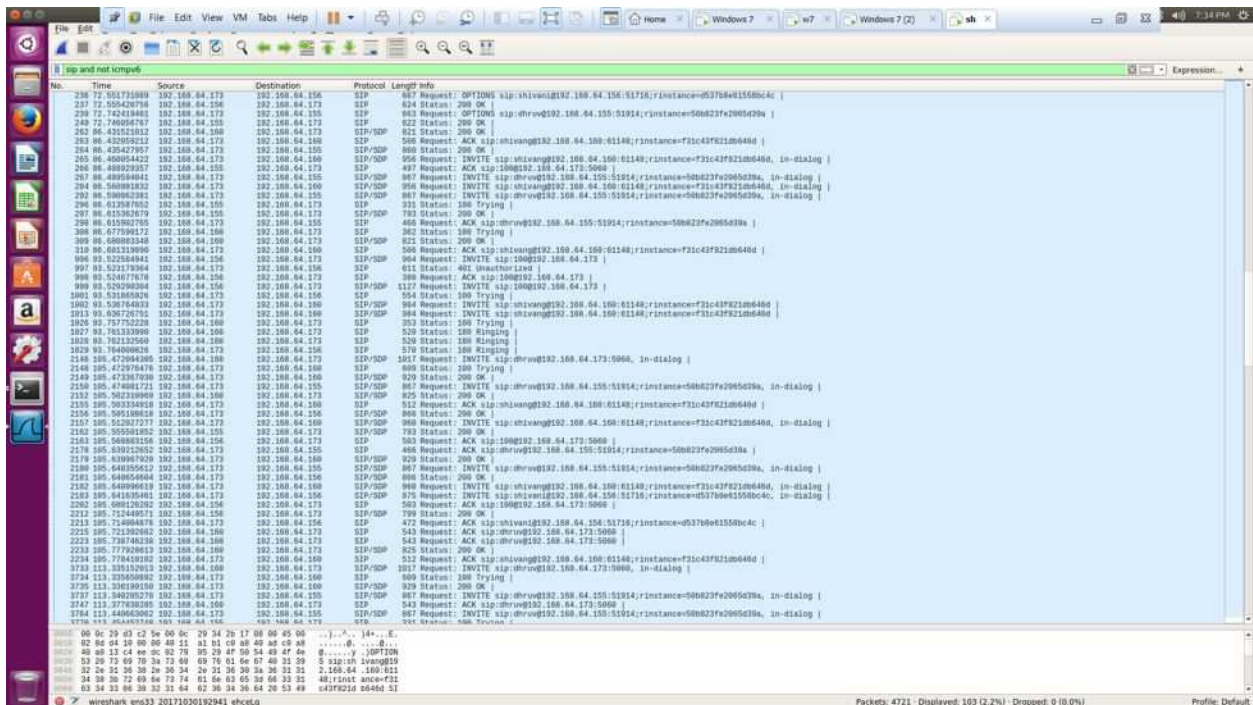
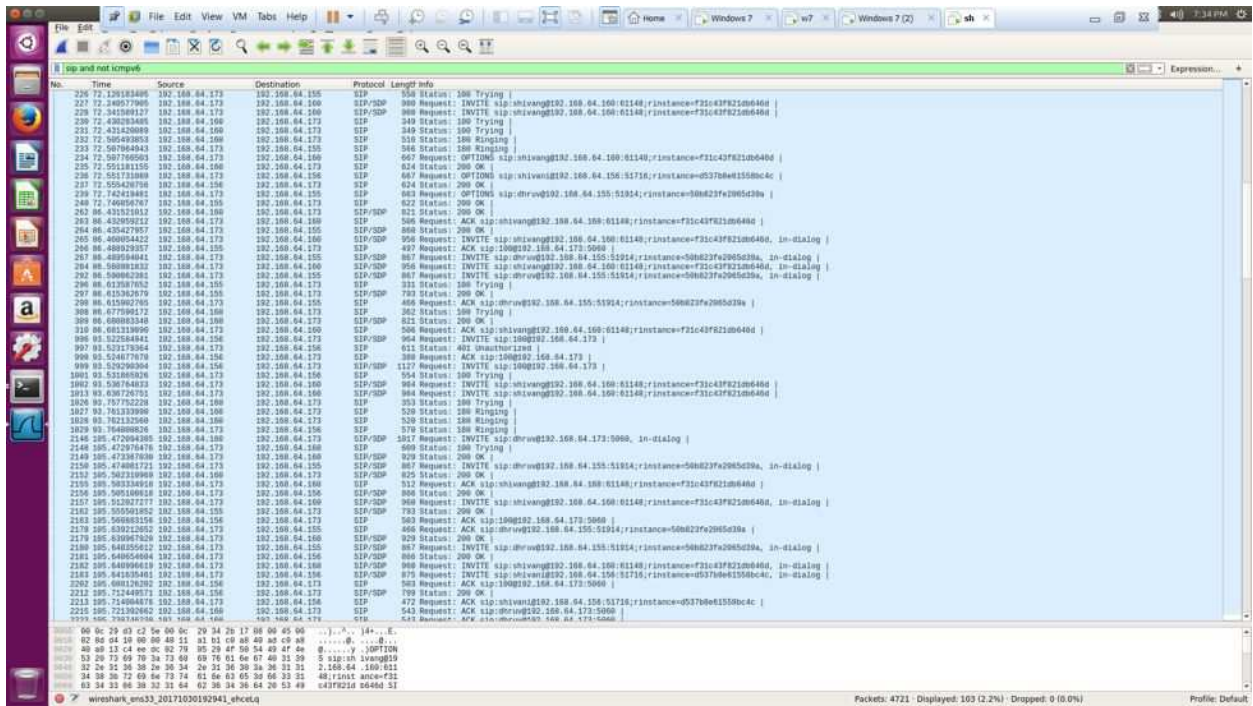
PHASE - 4: Call Conference

In this Call Conferencing scenario, the user with ID 300 calls user (ID#100), whereas a call is already in progress with two users (IDs#100 and 200). The user 2000, puts the call by 200 on Hold and establishes the call with user 300. Once the call has been set up between them, user again invites the other user (ID#200) to join the call. So, a conferencing call starts in between the users with IDs # 100, 200,300.

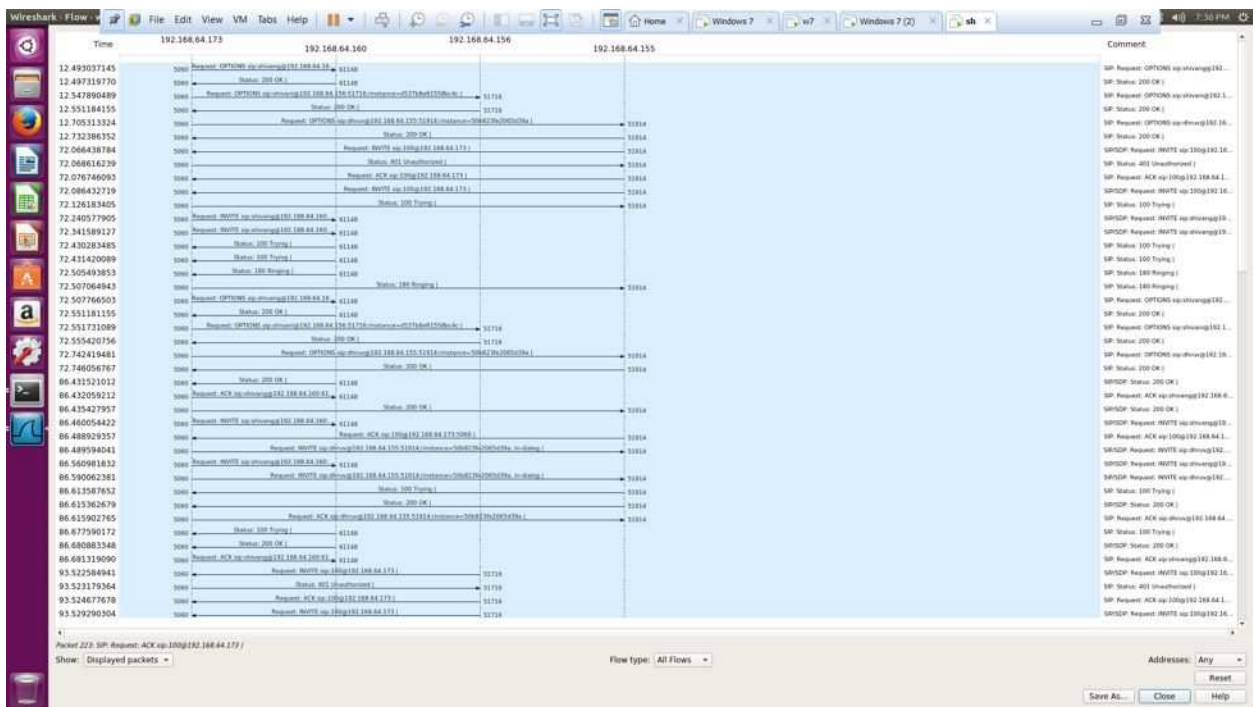
```
root@ubuntu:/usr# [Oct 27]
-- Using SIP RTP Ccs mark 5
-- Executing [100gphones:1] NoOp("SIP/dhruv-00000001", "Call for shivang") in new stack
-- Executing [100gphones:2] Dial("SIP/dhruv-00000001", "SIP/shivang") in new stack
-- Using SIP RTP Ccs mark 5
-- Called SIP/shivang
-- SIP/shivang-00000002 is ringing
-- SIP/shivang-00000002 exited non-zero on 'SIP/dhruv-00000001'
-- Spawn extension (phones, 100, 2) exited non-zero on 'SIP/dhruv-00000001'
-- Using SIP RTP Ccs mark 5
-- Executing [200gphones:1] NoOp("SIP/dhruv-00000003", "call for shivani") in new stack
-- Executing [200gphones:2] Dial("SIP/dhruv-00000003", "SIP/shivani") in new stack
-- Using SIP RTP Ccs mark 5
-- Called SIP/shivani
-- SIP/shivani-00000004 is ringing
-- SIP/shivani-00000004 is ringing
-- Got SIP response 486 "Busy Here" back from 192.168.64.156:61475
-- SIP/shivani-00000004 is busy
-- Everyone is busy/congested at this time (1:1/0/0)
-- Executing [200gphones:3] Hangup("SIP/dhruv-00000003", "") in new stack
-- Spawn extension (phones, 200, 3) exited non-zero on 'SIP/dhruv-00000003'
-- Using SIP RTP Ccs mark 5
-- Executing [200gphones:1] NoOp("SIP/shivang-00000005", "call for shivani") in new stack
-- Executing [200gphones:2] Dial("SIP/shivang-00000005", "SIP/shivani") in new stack
-- Using SIP RTP Ccs mark 5
-- Called SIP/shivani
-- SIP/shivani-00000006 is ringing
-- SIP/shivani-00000006 is ringing
-- Got SIP response 486 "Busy Here" back from 192.168.64.156:61475
-- SIP/shivani-00000006 is busy
-- Everyone is busy/congested at this time (1:1/0/0)
-- Executing [200gphones:3] Hangup("SIP/shivang-00000005", "") in new stack
-- Spawn extension (phones, 200, 3) exited non-zero on 'SIP/shivang-00000005'
-- Using SIP RTP Ccs mark 5
-- Executing [200gphones:1] NoOp("SIP/dhruv-00000007", "call for shivani") in new stack
-- Executing [200gphones:2] Dial("SIP/dhruv-00000007", "SIP/shivani") in new stack
-- Using SIP RTP Ccs mark 5
-- Called SIP/shivani
-- SIP/shivani-00000008 is ringing
-- SIP/shivani-00000008 is ringing
-- SIP/shivani-00000008 answered SIP/dhruv-00000007
-- Channel SIP/shivani-00000008 joined 'simple_bridge' basic-bridge <0908b070-60c4-4fd6-b2e6-8d8b3e787c8e>
-- Channel SIP/dhruv-00000007 joined 'simple_bridge' basic-bridge <0908b070-60c4-4fd6-b2e6-8d8b3e787c8e>
-- Using SIP RTP Ccs mark 5
-- Executing [200gphones:1] NoOp("SIP/shivang-00000009", "call for shivani") in new stack
-- Executing [200gphones:2] Dial("SIP/shivang-00000009", "SIP/shivani") in new stack
-- Using SIP RTP Ccs mark 5
-- Called SIP/shivani
-- SIP/shivani-00000009 is ringing
-- SIP/shivani-00000009 is ringing
-- SIP/shivani-00000009 is ringing
-- Started music on hold, class 'default', on channel 'SIP/dhruv-00000007'
-- SIP/shivani-00000009 answered SIP/shivang-00000009
-- Channel SIP/shivani-00000009 joined 'simple_bridge' basic-bridge <5bb5444d-0fe4-44bb-9642-721fa49e01f>
-- Channel SIP/shivang-00000009 joined 'simple_bridge' basic-bridge <5bb5444d-0fe4-44bb-9642-721fa49e01f>
-- Stopped music on hold on SIP/dhruv-00000007
-- Channel SIP/shivani-00000009 left 'native_rtp' basic-bridge <5bb5444d-0fe4-44bb-9642-721fa49e01f>
-- Channel SIP/shivang-00000009 left 'native_rtp' basic-bridge <5bb5444d-0fe4-44bb-9642-721fa49e01f>
-- Spawn extension (phones, 100, 2) exited non-zero on 'SIP/shivang-00000009'
-- Channel SIP/shivani-00000008 left 'native_rtp' basic-bridge <0908b070-60c4-4fd6-b2e6-8d8b3e787c8e>
-- Channel SIP/dhruv-00000007 left 'native_rtp' basic-bridge <0908b070-60c4-4fd6-b2e6-8d8b3e787c8e>
-- Spawn extension (phones, 200, 2) exited non-zero on 'SIP/dhruv-00000007'
ubuntu@cli:~$
```


Call conferencing using all three users -

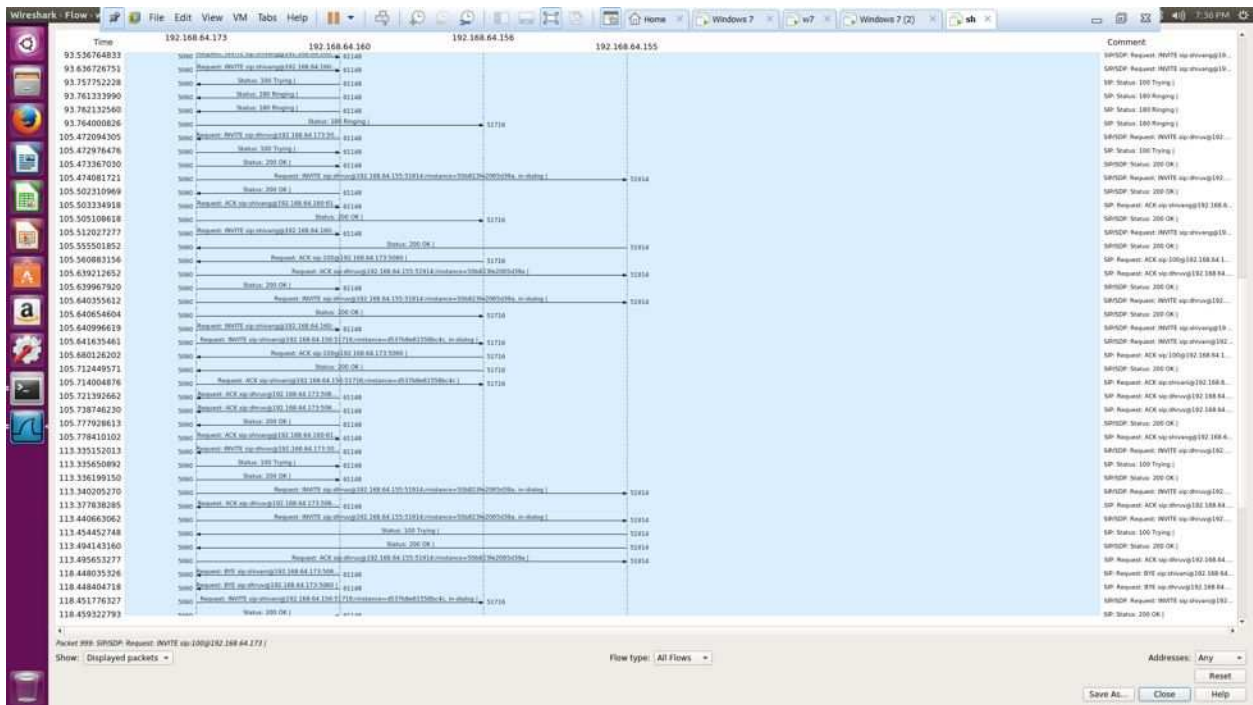




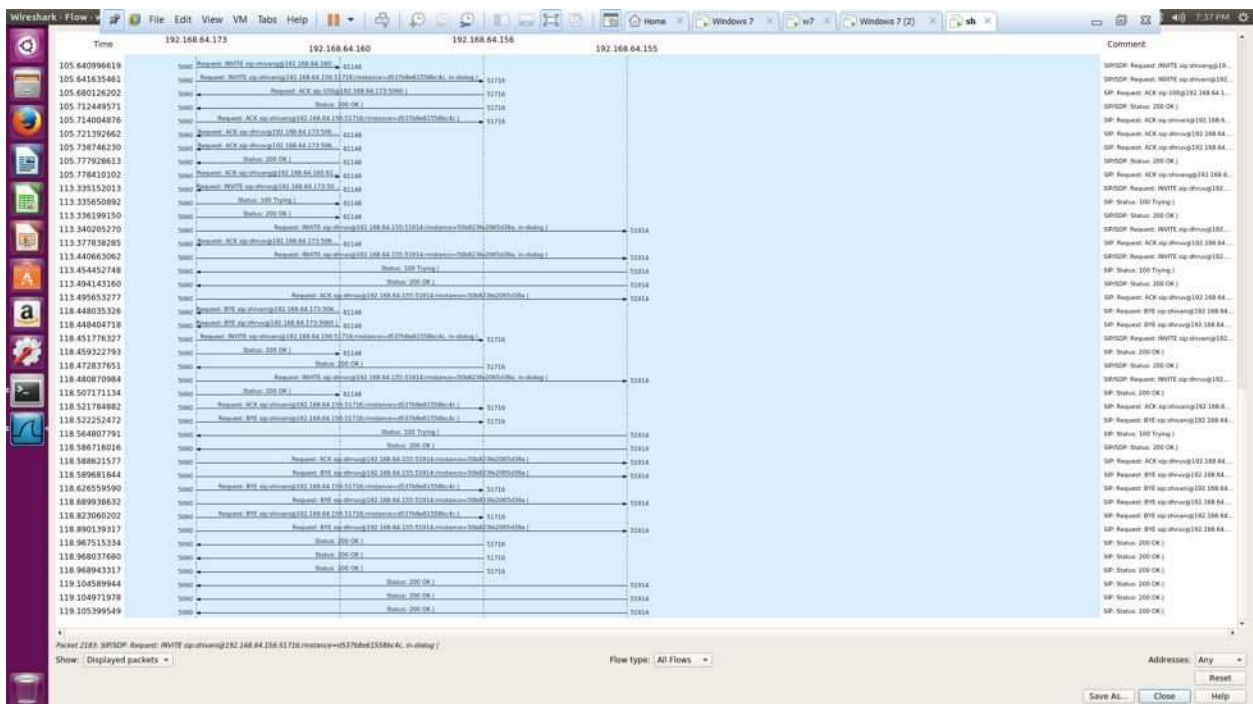
Only SIP files-



192.168.64.173 multicasts to all



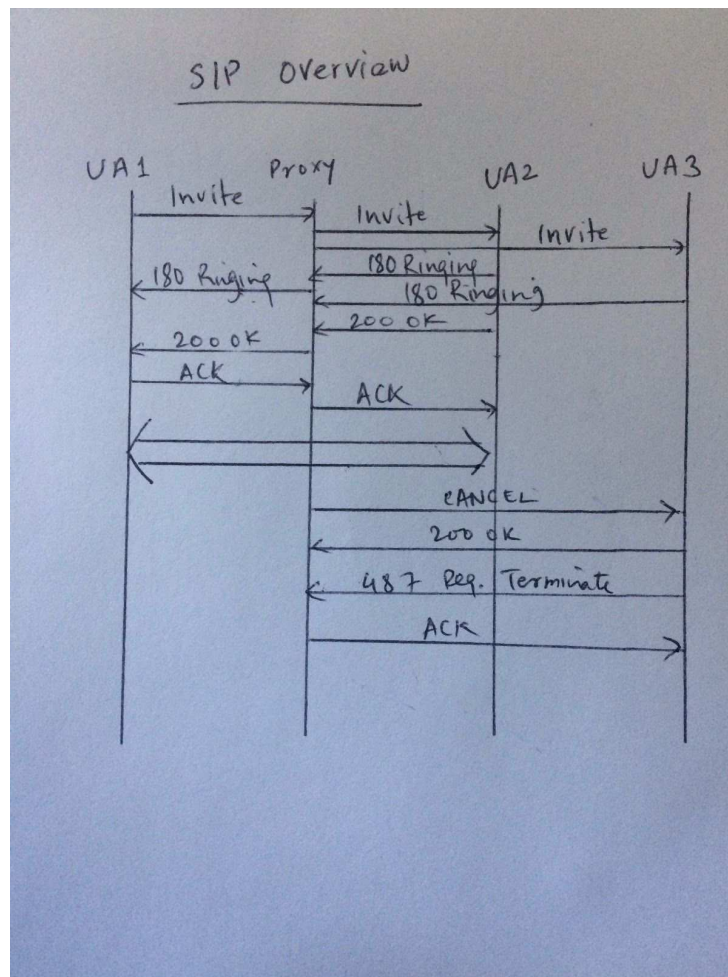
192.168.64.173 to 192.168.64.156 via 192.168.64.160



PART - 2

SIP Overview-

Session Initiation Protocol (SIP) is used to establish the session between two parties. It is a communication and media exchange protocol. It is an application as well as session layer protocol which varies with the functionality it performs at an instance. It works in corporate with other application layer protocols like H323. Three way handshaking can be seen here in the following figure -



PJSIP -

- It is a free and open source multimedia communication library which is written in C language implementing standard based protocol such as SIP, SDP, RTP, ICE and others.
- It is both compact and feature-full. It supports all video, audio and instant messaging facilities.

The parts of our script, requiring special attention as described as below-

1. Lib Class-

- This is the most most library class. It needs to be initialized only one single time in the entire program from the point we start the library.
- It is used to create other important objects like transport and accounts.
- We have created the callback class as “log_cb”.

```
# Creating the library instance of Lib Class
lib = pj.Lib()
# Instantiate library with default configuration
lib.init(log_cfg = pj.LogConfig(level=3, callback=log_cb))
```

2. Call Class-

- A call class is created to do two things here -
 - a. To Answer a call
 - b. For hanging up the call

After successful registration, the PJSIP client attempts to send the INVITE message to the server to call the client in the argument. We can notice that if “call” instance is created successfully then the INVITE is sent to the server with which it is registered. “SRDialCallBack()” is called here to get the notifications on the change of state of events of call.

Here ahead of that the program will wait for the user to exit on the input line and if user gives the ENTER command, program will be exited and main class instance will be destroyed and hence successful call will be completed.

```
# Start calling process
b=raw_input("Enter destination URI: ")
call = acc.make_call(b, SRDialCallback())
```

3. Callback Classes -

We use Callback classes to get the notification. There are various callback classes such as Account Callback class, call callback class and many more to handle the relevant methods and install the necessary callback instance to the object.

```

# To retrieve events from the call
class SRDialCallback(pj.DialCallback):
    def __init__(self, call=None):
        pj.DialCallback.__init__(self, call)

    def on_state(self):
        print("Call is ON :", self.call.info().state_text),
        print ("Last code is :", self.call.info().last_code),
        print ("(" + self.call.info().last_reason + ")")

# Notification whenever there is a change in media state
def on_media_state(self):
    global lib
    if self.call.info().media_state == pj.MediaState.ACTIVE:

        #Connecting the call to a different sound device
        call_slot = self.call.info().conf_slot
        lib.conf_connect(call_slot, 0)
        lib.conf_connect(0, call_slot)
        print ("Hi. How are you doing today???)
        print (lib)

# Main Part

```

Python-Client Code :

```

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help
For 284 Project - untitled3 - [~/PycharmProjects/untitled3]
No Python interpreter configured for the project
1 import sys
2 import pjsua as pj
3
4 LOG_LEVEL=3
5 current_call = None
6
7 # For printing the log
8 def log_callback(level, string, len):
9     print string
10
11 # To get the notifications for Account Registration
12 class MyAccCallback(pj.AccountCallback):
13     def __init__(self, acc=None):
14         pj.AccountCallback.__init__(self, acc)
15     def on_reg_state(self):
16         if self.state == pj.AccountState.REGISTRATION_SUCCESS:
17             self.call.release()
18
19 # Incoming call
20 def on_incoming_call(self, call):
21     global current_call
22     if current_call:
23         call.answer(486, "Busy")
24     return
25
26 print "Incoming call from: ", call.info().remote_uri
27 print "Type to answer"
28
29 current_call = call
30
31 call_cb = MyAccCallback(current_call)
32 current_call.set_callback(call_cb)
33
34 # For Ringing
35 current_call.answer(180)
36
37 # To retrieve events from the call
38 class SRDialCallback(pj.DialCallback):
39     def __init__(self, call=None):
40         pj.DialCallback.__init__(self, call)
41
42     def on_state(self):
43         print("Call is ON :", self.call.info().state_text),
44         print ("Last code is :", self.call.info().last_code),
45         print ("(" + self.call.info().last_reason + ")")
46
47 # Notification whenever there is a change in media state
48 def on_media_state(self):
49     global lib
50     if self.call.info().media_state == pj.MediaState.ACTIVE:
51
52         #Connecting the call to a different sound device
53         call_slot = self.call.info().conf_slot
54         lib.conf_connect(call_slot, 0)
55         lib.conf_connect(0, call_slot)
56         print ("Hi. How are you doing today???)
57         print (lib)
58
59 # Main Part
60 on_incoming_cal...

```

PyCharm Console: Error running 'scratch_11': @NoNull method com.intellij.executor.configurations.GeneralCommandLine.getExePath must not return null (a minute ago)

22:33 LFI UTF-8


```
untitled3 For 284 Project scratch_11
No Python interpreter configured for the project
Configure Python Interpreter

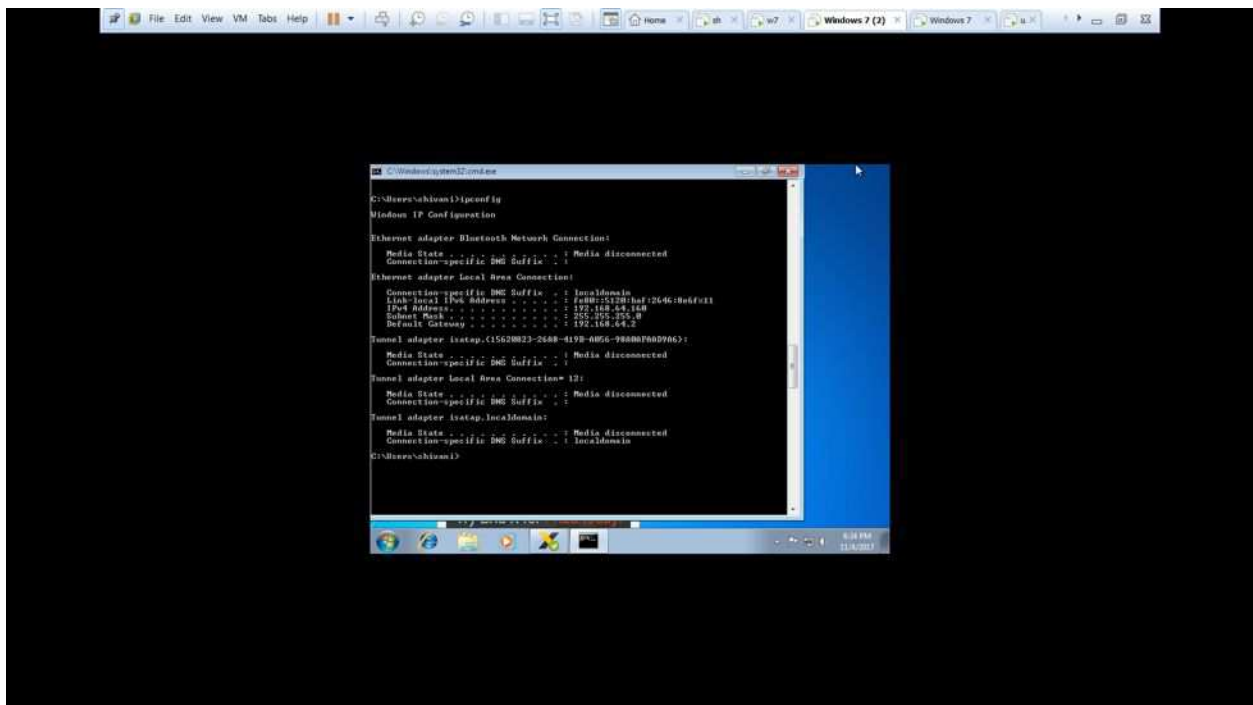
61 # Main Part
62 def make_call(uri):
63     try:
64         print "making call to", uri
65         return acc.make_call(uri, cb=MyAccCallback())
66     except pj.Error, e:
67         print "exception: " + str(e)
68         return None
69
70 # Creating the library instance of Lib Class
71 lib = pj.Lib()
72
73 # Instantiate library with default configuration
74 lib.init(log_cfg = pj.LogConfig(level=LOG_LEVEL, callback=log_cb))
75
76 # Configure one Transport Object and fixing it for listening to 5060 port and UDP protocol
77 trans_conf = pj.TransportConfig()
78
79
80 print "-----REGISTRATION BELOW-----"
81 print "\n\n"
82
83 trans_conf.port = 5060
84 raw_input("Enter the IP Address of your client : ")
85 print "we have our Default port number 5060 for SIP"
86
87 trans_conf.bound_addr = a
88 transport = lib.create_transport(pj.TransportType.UDP, trans_conf)
89
90 # Initiate the instance for Library Class
91 lib.start()
92
93 #when no sound card is found
94 lib.set_null_snd_dev()
95
96 #Configuring account class to register with registrar server
97 #Giving info to create header of REGISTER SIP message
98
99 ab4=raw_input("Enter the IP address of your server: ")
100 ab=raw_input("Enter username: ")
101 ab1=raw_input("Enter password: ")
102 ab2=raw_input("Do you want to use same Display name and username ? Enter valid entry Y/N T")
103 if ab2=="Y" or ab2=="y":
104     ab3=ab
105 else:
106     ab3=raw_input("Enter display name: ")
107 acc_conf = pj.AccountConfig(domain = ab4, username = ab, password = ab1, display = ab3)
108
109 # registrar = 'sip:' + ab4 + ':5060', proxy = 'sip:' + ab4 + ':5060'
110
111 acc_conf.id = "sip" + ab
112 acc_conf.reg_uri = 'sip:' + ab4 + ':5060'
113 acc_callback = MyAccCallback(acc_conf)
114 acc = lib.create_account(acc_conf, cb=acc_callback)
115
116 #Creating instance of AccCallback Class
117 acc.set_callback(acc_callback)
118 print "\n\n"
119 print "Registration is completed"
120 print("Status: ", acc.info().reg_status, "\n")
121 print("Reason: ", acc.info().reg_reason, "\n")
122
123 on_incoming_call...
```

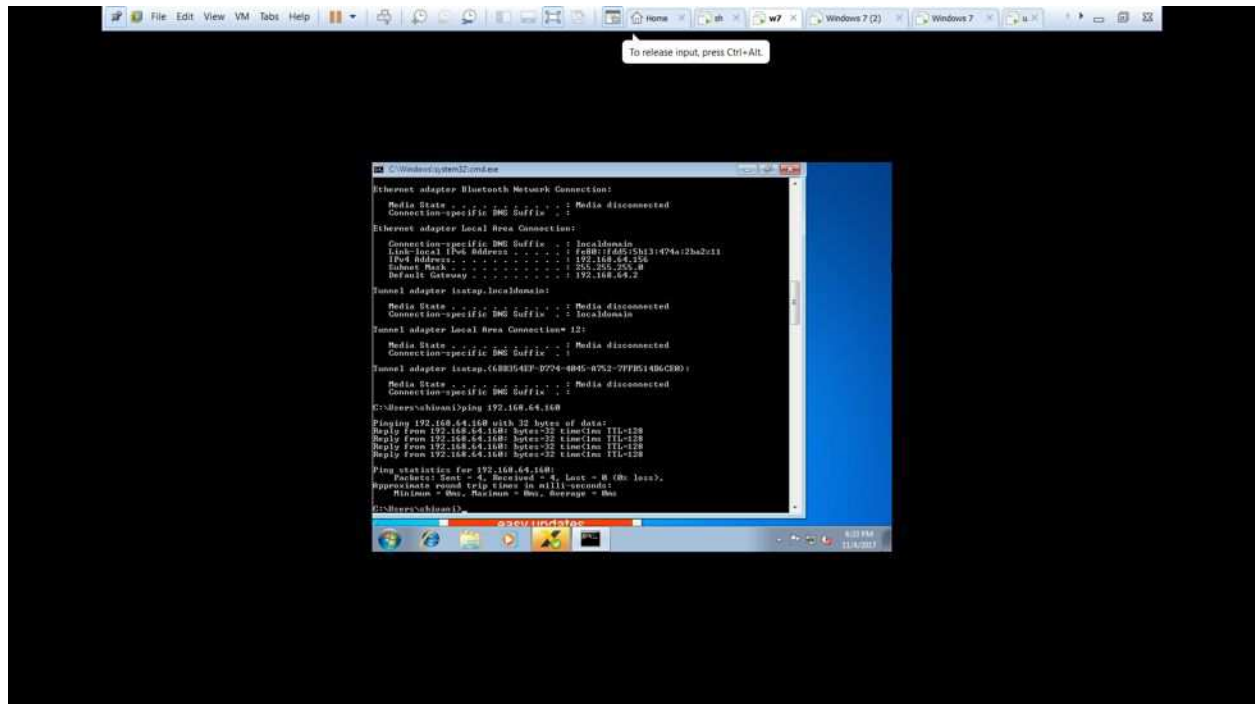
```
untitled3 For 284 Project scratch_11
No Python interpreter configured for the project
Configure Python Interpreter

123 on_incoming_call...
124
125 ab5=raw_input("Do you want to make a call now ?? Y/N\n")
126 print "\n\n"
127
128 if ab5=="Y" or ab5=="y":
129     # Start calling process
130     b=raw_input("Enter destination URI: ")
131     call = acc.make_call(b, SRDialCallback())
132
133 #Client Side Waiting for ENTER Command to exit
134 print ("Press <ENTER> to exit and destroy library")
135 input = sys.stdin.readline().rstrip("\r\n")
136
137 # To shut down the library
138 lib.destroy()
139 lib = None
140
141 else:
142     print "Unregistering"
143     time.sleep(2)
144     print "Destroying the libraries"
145     time.sleep(2)
146     lib.destroy()
147     lib = None
148     sys.exit(1)
149
150 except pj.Error, e:
151     print ("Exception: " + str(e))
152     lib.destroy()
153     lib = None
154     sys.exit(1)
```

MoS VALUE CALCULATION

- Mean Opinion Score (MoS) is a measure which is used representing the overall quality of a system.
 - MoS value is the result of underlying network attributes and is extremely useful in accessing the call quality.
 - It is a commonly used measure for the quality of audio, video and audiovisual entities.
- In our experiment we have pinged two users and then calculated the MoS value.
- Also since we are pinging through a LAN connection, so the packet loss is Zero or near to zero, which means a perfect transmission over the connection to the two entities.





- Here we have pinged 192.168.64.168 with 32 bytes of data and we see that-

Total number of packets sent = 4

The number of packet received = 4

LOSS = 0 (0% loss)

So the MoS becomes= **5** (max.) and the label is **EXCELLENT**

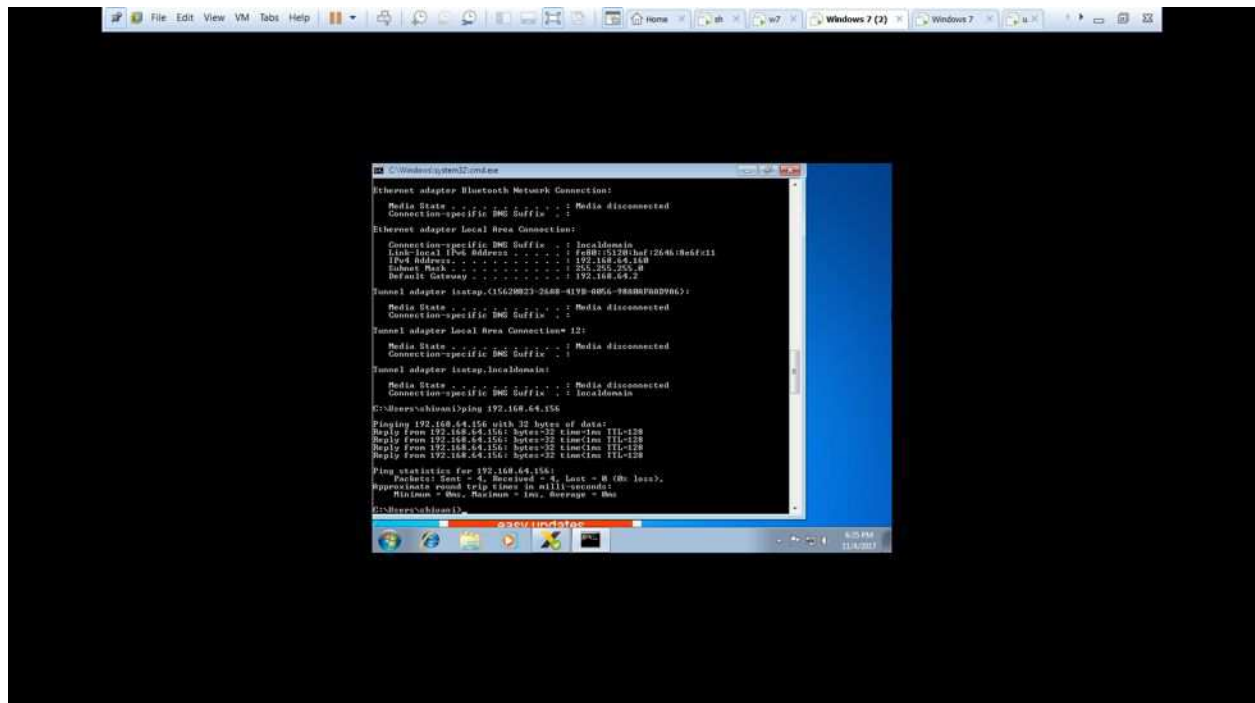
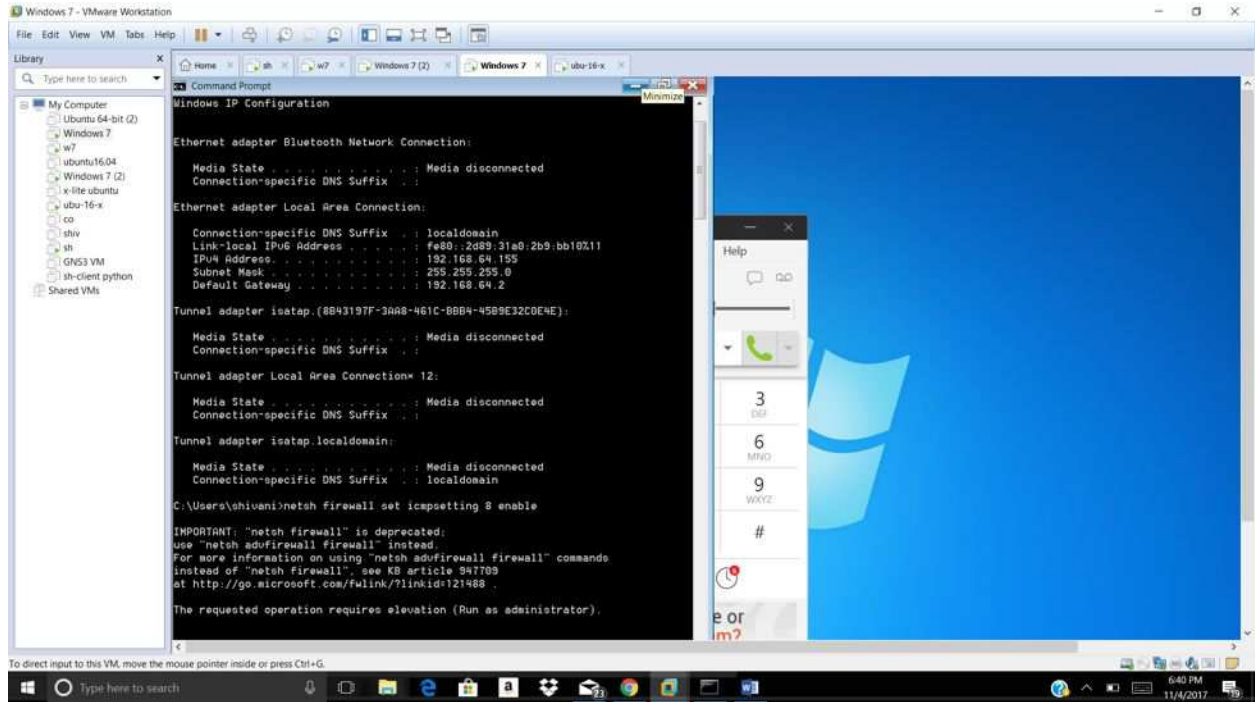
- In this one, I have pinged 192.168.64.156 and see that -

Total number of packets sent = 4

The number of packet received = 4

LOSS = 0 (0% loss)

So the MoS becomes= **5** (max.) and the label is **EXCELLENT**



This is how we pinged and calculated the MoS value