ExptsSec: S2.d. Experiment Design to Extend the Scope to Wireless Nodes in Emulab and ProtoGENI

Xiaoyan Hong, Fei Hu, Yang Xiao, Dawei Li, Fnu Shalini
University of Alabama
Sept 24, 2010

I. INTRODUCTION

This document describes the design of the experiments to extend the scope of the security assessment of ProtoGENI. We targeted at new types of aggregates. After the initial screening, we picked the wireless nodes in Utah Emulab. ProtoGENI advertizes the Emulab nodes with wireless links. Through Rspec, an experimenter can request and obtain the wireless nodes after a prior validation about their availability.

We describe our initial investigations and the design of experiments that will explore potential security vulnerability.

II. INITIAL WIRELESS EXPERIMENTS

We have conducted several experiments to investigate the following issues in order to familiar ourselves to the testbed supports for experimentation.

(a) test the configured parameters and the achieved metrics over the wireless links in a LAN mode. The parameters and metrics include bandwidth, delay, protocol selection and general performance measurements.
(b) Test the channel interferences with two simultaneous experiments.
(c) Test multihop configuration in Emulab.
(d) Test the capability of wireless traffic capture in Emulab.
(e) Test the capability of wireless traffic capture in ProtoGENI.

The details of these experiments are presented in the Appendix.

III. EXPERIMENT DESIGN

There are inherited physical deployment limitations for Emulab wireless nodes. These limitation are stated for the experimenters on the web page. Security and privacy policies are clearly given to the experimenters as well. However, to an uninformed user or a purposeful user, the listed policy items are vulnerabilities. Our designs go beyond what are listed as policies. Our goal of the experiment design is to reveal the potential threats that come from the wireless link and wireless networks that could be formed using the available nodes. Different from a wired connection from an experimenter to the GENI testbed, the open nature wireless media makes it easier for one experimenter to intervene others' experiments. We outline the potential security issues and plans for our experiments.

3.1. Eavesdropping

A single wireless node can launch the eavesdropping attack. Many tools are available for this purpose. The data obtained by eavesdropping can lead to different uses. We plan to perform the experiments for the follow issues:

(a) The capabilities of capturing the wireless traffic using a few different tools. Though we have tested Wireshark, there are remaining issues with Wireshark, for example, potential errors in sequence of records. In addition, some tools can provide more information on the channel usage.
(b) Capture traffic from the channels that are used by other users.
(c) Try to analyze a few select protocols to see if obtaining useful information is possible. We understand that WPA provides strong security protection. An experimenter could set different configurations.

3.2. Explore wireless links

The wireless links make it easier for the nodes in vicinity to explore vulnerability. We plan to investigate the following protocols. The direct impact of these experiments would be on the experiment in question. But our main interest is to understand the impact on the large GENI and may be on the Internet.

(d) TCP is a potential protocol at risk. We plan to perform experiments to explore possible DOS attacks against TCP.
(e) A multihop wireless network requires nodes to work at ad hoc mode and run routing protocols. A few issues rise following the general wireless network security research. We plan to perform experiments to study the feasibility of selected vulnerabilities and their potential consequences that could lead to GENI.

IV. APPENDIX: REPORT ON INITIAL WIRELESS EXPERIMENTS

4.1 Test emulab wireless links of different bandwidth, delay, protocol and lan-mod

Purpose: Test configuration and achievable bandwidth and delay, general performance measurement in a LAN mod. Test if we can get what we configured.

Experiment Setup: In this experiment, we used the ns scripts provided by the advanced emulab tutorial. It has one AP

(connected to a host through wired link) and two wireless nodes. Connections details see Fig 1.

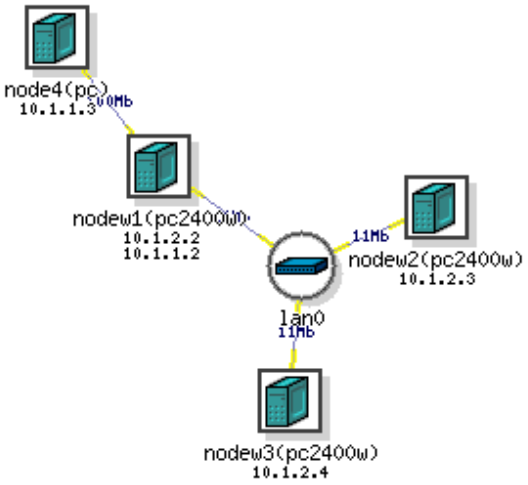

Fig 1

Physical Nodes and Virtual Nodes mapping:

Access point: pcwf146: nodew1,

AP associated host: pcwf150: node4

Two wireless nodes: pcwf147: nodew3, pcwf145: nodew2

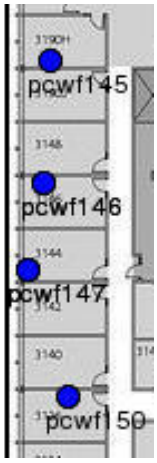

Fig 2

Experiments: The experiments used the "link test function" to test the maximum achievable bandwidth, delay and loss for various network configurations, including protocol, bandwidth, delay and LAN-mode. The link-test function produces performance measurement in terms of the metrics of latency, routing, loss and bandwidth. There are 4 linktest levels available:

Level 1 – Connectivity and Latency (using ping)

Level 2 – Plus Static Routing (ping all the other nodes)

Level 3 – Plus Link Loss (using Rude and Crude, a real-time packet emitter and collector)

Level 4 – Plus Bandwidth (using Iperf)

However, there are limitations for the linktest as mentioned in emulab document (http://users.emulab.net/trac/emulab/wiki/linktest). Not all bandwidths can be accurately measured.

For LAN protocols, two common used protocols: 802.11g and 802.11b are available.

Bandwidth test: Experiments are performed regarding to each protocol. For protocol 802.11b the 11Mb be the maximum bandwidth by standard and for 802.11g, 54Mb be the maximum bandwidth by standard. The experiments configured different bandwidths, which are listed in the Table I. They contain values below, equal or above the default one.

Delay test: Combined with the bandwidth configuration, different delays are specified on NS as well. The default delay is 0ms.

Moreover, the experiments also used different LAN-mode (access-point and ad-hoc). All used channel 14.

In each experiment, the four variables are set to one value discussed above in NS file. For each experiment, we record performance. All the above configurations are listed in Table 1 and results are listed in Table 2.

Experiment Name	**Network Setup**			
	Lan protocol	**bandwidth**	**delay**	**Lan-mode**
Test1	802.11g	11Mb	0ms	AP
Test2	802.11g	54Mb	0ms	AP
Test3	802.11g	25Mb	0ms	AP
Test4	802.11g	54Mb	30ms	AP
Test5	802.11g	100Mb	0ms	AP
Test6	802.11b	11Mb	0ms	AP
Test7	802.11b	54Mb	0ms	AP
Test8	802.11b	5Mb	0ms	AP
Test9	802.11b	11Mb	0ms	Adhoc
Test10	802.11b	5Mb	0ms	Adhoc
Test11	802.11g	54Mb	0ms	Adhoc

Table 1

Result analysis:

1. though the estimated maxmum bandwidthes for 802.11b and 802.11g are 11Mb and 54Mb, in reality, they can not be reached, as is well understood.
2. But for a bandwidth set smaller than the default one, the wireless nodes can set to a close bandwidth. This is a good design from emulab.

Experiment Name	**Link Test Result**			
	Latency test	**Routing test**	**Loss test**	**Band test**
Test1	OK	OK	OK	12.46Mb
Test2	OK	OK	OK	Vary[1]
Test3	OK	OK	OK	Vary[2]
Test4	Error	OK	OK	Vary[1]
Test5	Fail			
Test6	OK	OK	OK	Vary[3]
Test7	Fail			
Test8	OK	OK	OK	5.66Mb
Test9	OK	OK	OK	Vary[4]
Test10	OK	OK	OK	5.66Mb
Test11	OK	OK	Loss	Vary[5]

TABLE 2

3. Also, the bandwidth can not be set to a number larger than the default bandwidth.

4. Latency can not be added to wireless links,
5. Adhoc mod may lead to unexpected loss.

TABLE 2: CAPTION

[1] nodew1– nodew2: 39.11 nodew2- nodew1: 27.64 nodew1- nodew3: 32.06 nodew3- nodew1: 38.53
[2] nodew1- nodew2:28.35 nodew2- nodew1:28.01 nodew1- nodew3:27.65 nodew3- nodew1:28.35
[3] nodew1- nodew2:7.81 nodew2- nodew1:7.94 nodew1- nodew3:7.66 nodew3- nodew1:7.68
[4] nodew1- nodew2:6.55 nodew2- nodew1:6.97 nodew1- nodew3:6.51 nodew3- nodew1:6.92
[5] nodew1- nodew2:25.29 nodew2- nodew1:27.94 nodew1- nodew3:24.04 nodew3- nodew1:27.44

4.2 Channel Interference

Purpose: Testing channel interferences by two simultaneous experiments

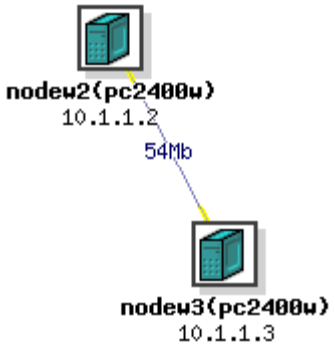

Fig 3

Scenario:

We use two experiments: test1 and test2, using the same topology as above. Both use 802.11g and communicate in the same channel 14. Each experiment uses two nodes, ad hoc mode, link protocol 802.11g. The physical locations of the four nodes are shown in Fig 4. When the two experiments send traffic at the same, they will interfere if the same channel is used. (We should have two cases). This experiment also uses link test function.

Physical nodes mapping:

For test1:
nodew2: pcwf147
nodew3: pcwf148
For test2:
nodew2: pcwf146
nodew3: pcwf150

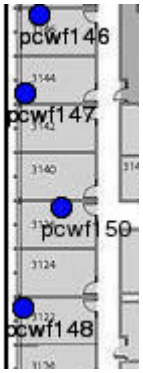

Fig 4

Test results are shown in Table 3.

TABLE 3

BANDWIDTH TEST (INTEGRAL BANDWIDTH, NULL MEANS NO LINK TEST IS RUNNING ON THE NODE):

Experiment Name	Nodew2->nodew3 for test1	Nodew3->nodew2 for test1	Nodew2->nodew3 for test2	Nodew3->nodew2 for test2
Exp1	6	6	Null	Null
Exp2	6	6	Null	Null
Exp3	6	6	Null	Null
Exp4	Null	Null	6	15
Exp5	Null	Null	6	14
Exp6	Null	Null	6	19
Exp7	6	0.4	1	13
Exp8	6	Could not find	1	10
Exp9	6	0.7	2	12

Packet lost happened for all the above scenarios.

Result analysis:

The above table shows the interference between two experiments at the same channel when doing link test at the same time.

4.3 Multihop topology

This experiment asks for a 4 node multihop network. It has not been successful because we were not able to find 4 available nodes that can be physically multihop. A best case we had was that three nodes are within each other's transmission range. We tested two NS configurations. In the first cast, using the ad-hoc mode, the routing protocol is set to Manual and the routing tables has later been modified. The wireless nodes can still reach each other directly without a multi-hop. In the second case, we set each link pair as a subnet. The multiple hop line topology is formed. Experiments are continuing with the attempts to find available physical nodes.

4.4 Design and Execution of Emulab Wireless traffic Experiment

As per Emulab wireless map and tabular views, we initiated an experiment to observe traffic between two wireless nodes. A ns file was created to acquire required PCs with some traffic

generation code for node and link monitoring through built-in Link Tracing/Monitoring tool in Emulab.We choose two nearby wireless nodes pc4 and pc5 as shown in fig. 2. Traffic was generated and monitored as per Emulab Link Tracing/Monitoring tool. There were some details initially through Emulab Link Tracing/Monitoring tool but then it always showed same patterenin all variations of experiment. We tried to capture the traffic between two wireless nodes in different network events. At first, we tried to observe network traffic while in-built link tracing/monitoring was paused. After this, we send ping from pc4 to pc5 then we restarted the Emulab link tracing/monitoring and observed the changes in Wireshark.

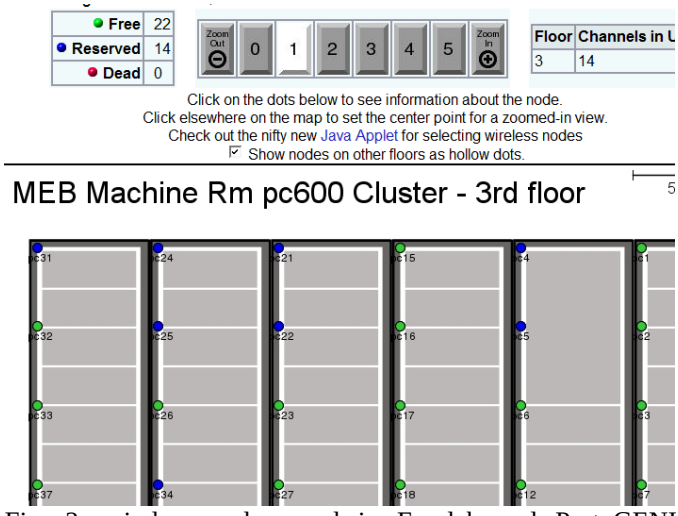

Fig. 2: wireless nodes used in Emulab and ProtoGENI experiments

Next we uploaded client server files to pc4 and pc5. Client file was modified to send and receive the messages in an indefinite loop to the server. pc4 was considered as client and pc5 was considered as client. We captured the traffic and analyzed it through summary, filter options and through expert info on captured traffic packets to have an idea about network traffic state (fig 1).

Emulab experiment's details are shown as fig. 3. Details of used ns file is attached as Annexure-A.

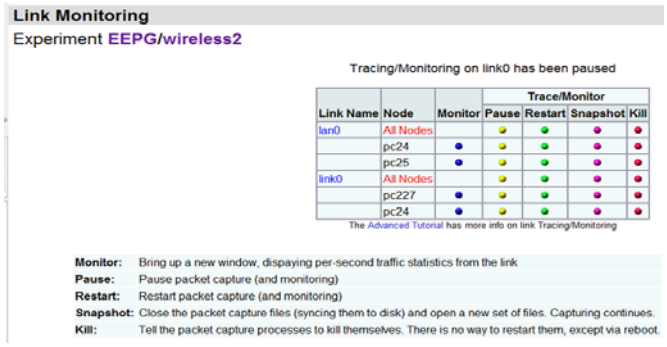

Fig. 3: Link Tracing/Monitoring options in Emulab

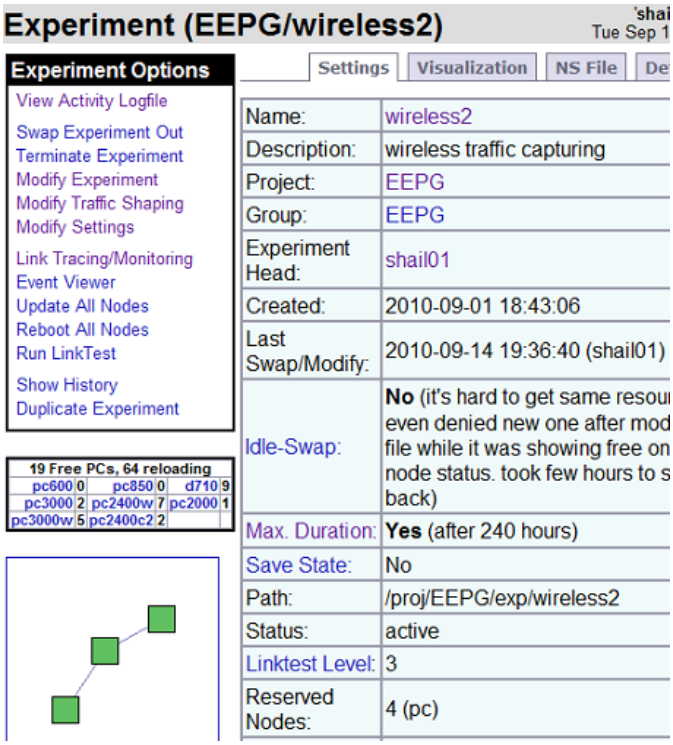

Fig. 4: Emulab experiment wireless2

Next we uploaded client server files to pc4 and pc5. Client file was modified to send and receive the messages in an indefinite loop to the server. pc4 was considered as client and pc5 was considered as client. We captured the traffic and analyzed it through summary, filter options and through expert info on captured traffic packets to have an idea about network traffic state (fig 1).

Emulab experiment's details are shown as fig. 3. Details of used ns file is attached as Annexure-A.

Next we uploaded client server files to pc4 and pc5. Client file was modified to send and receive the messages in an indefinite loop to the server. pc4 was considered as client and pc5 was considered as client. We captured the traffic and analyzed it through summary, filter options and through expert info on captured traffic packets to have an idea about network traffic state (fig 1).

Emulab experiment's details are shown as fig. 3. Details of used ns file is attached as Annexure-B.

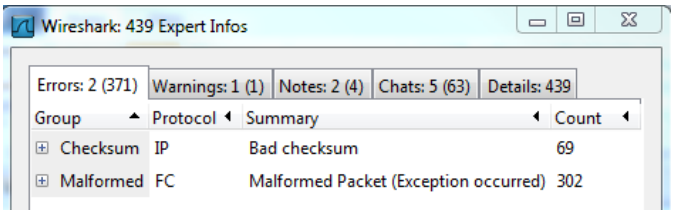

Fig. 5: Wireshark summary for pc24 when pc24 pings pc25

1. No traffic monitoring- no other active communication between pc24 and pc25
2. Traffic monitoring through Emulab in-built tracing/monitoring option: it didn't show any traffic

and we also couldn't see any traffic in wireshark captured traffic related to all involved PC ip addresses. Though delay node showed some files captured there for PC 24 and PC 227 but we couldn't download the file to open in wireshark as showing denial as per fig. 6.

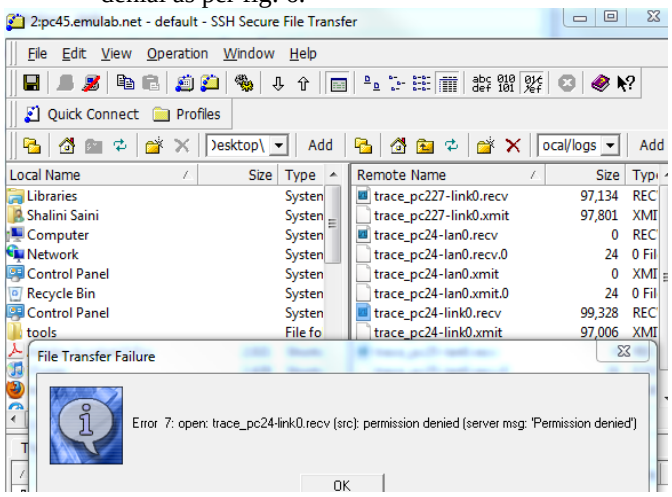

Fig. 6: Problem in downloading files from delay node PC 45

Then we tries to open file at pc45 itself and we could see the following details:

```
> cd /local
> cd logs
> ls
trace_pc227-link0.recv  trace_pc24-lan0.xmit    trace_pc25-lan0.recv
trace_pc227-link0.xmit  trace_pc24-lan0.xmit.0  trace_pc25-lan0.recv.0
trace_pc24-lan0.recv    trace_pc24-link0.recv   trace_pc25-lan0.xmit
trace_pc24-lan0.recv.0  trace_pc24-link0.xmit   trace_pc25-lan0.xmit.0
> sudo vim trace_pc24-link0.recv

2
<NO IP> 'ether src 00:d0:b7:10:08:46'
<NO IP> 'ether src 00:d0:b7:10:08:46'
1284514546.379340 (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (icmp:0,0 tcp:0,0 udp:0,0
 other:0,0) (dropped:0)
1284514547.379345 (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (icmp:0,0 tcp:0,0 udp:0,0
```

Fig. 7(a): trace files at delay node pc45

```
1284514548.379325 (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (dropped:0)
1284514549.379322 (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (dropped:0)
1284514550.379321 (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (dropped:0)
1284514551.379322 (icmp:0,0 tcp:0,0 udp:328,1 other:0,0) (icmp:0,0 tcp:0,0 udp:328,1 other:0,0) (dropped:0)
1284514552.379321 (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (dropped:0)
1284514553.379319 (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (dropped:0)
1284514554.379317 (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (dropped:0)
1284514555.379317 (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (icmp:0,0 tcp:0,0 udp:0,0 other:0,0) (dropped:0)
```

Fig. 7(b) trace_pc24-link0.recv file details from pc45

Pc 24 pings pc25: it showed some network activities in wireshark captured traffic which also includes some other Utah node as 155.98.32.70

We tried to analyze different captured traffic as per tools provided by wireshark like 'expert info' which summarize the details of errors and types of events occurred in captured traffic.

We can see from summary and expert info that there are large number of bad checksum errors. Wireshark captured traffic also showed different size of encrypted messages while pc24 and pc 25 were communicating as clinet server. More in-depth analysis is required to clarify the initial details through these traffic capturing events for these traffic capturing events for this Emulab experiment to observe traffic between wireless nodes.

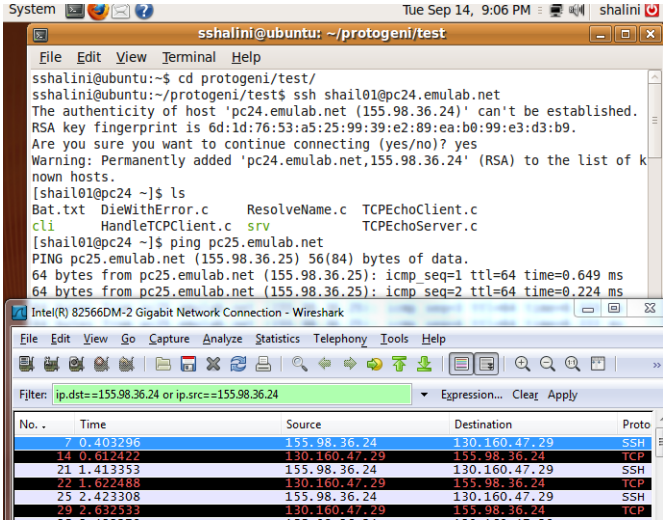

Fig. 7: PC 24 pings PC25- while no tracing/monitoring through Emulab tool

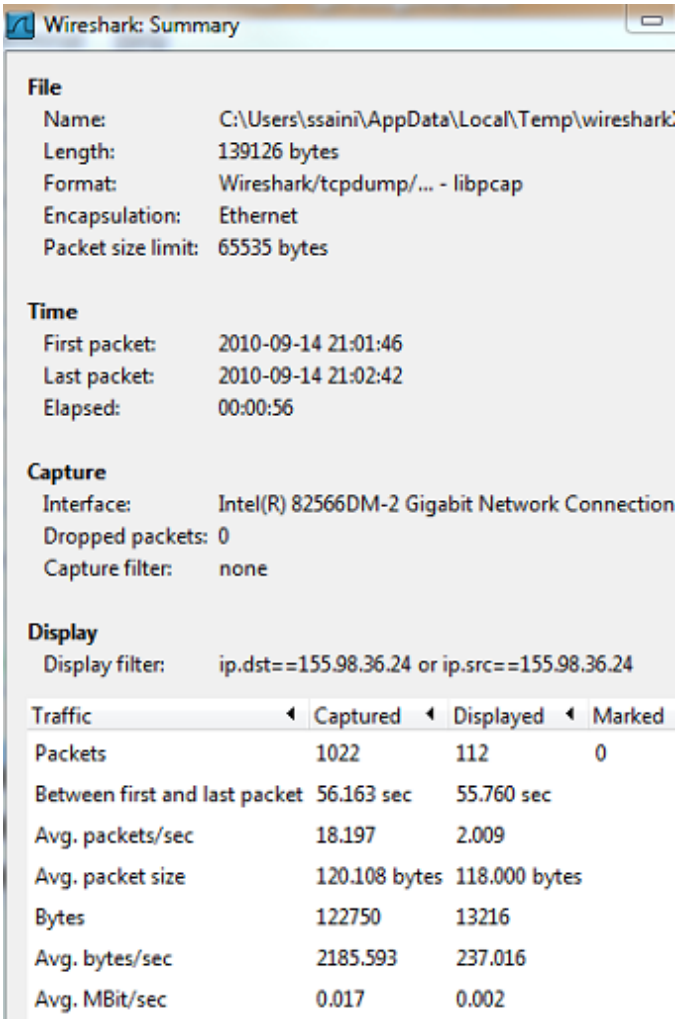

Fig. 8: Wireshark summary for pc24, when pc24 pings pc25

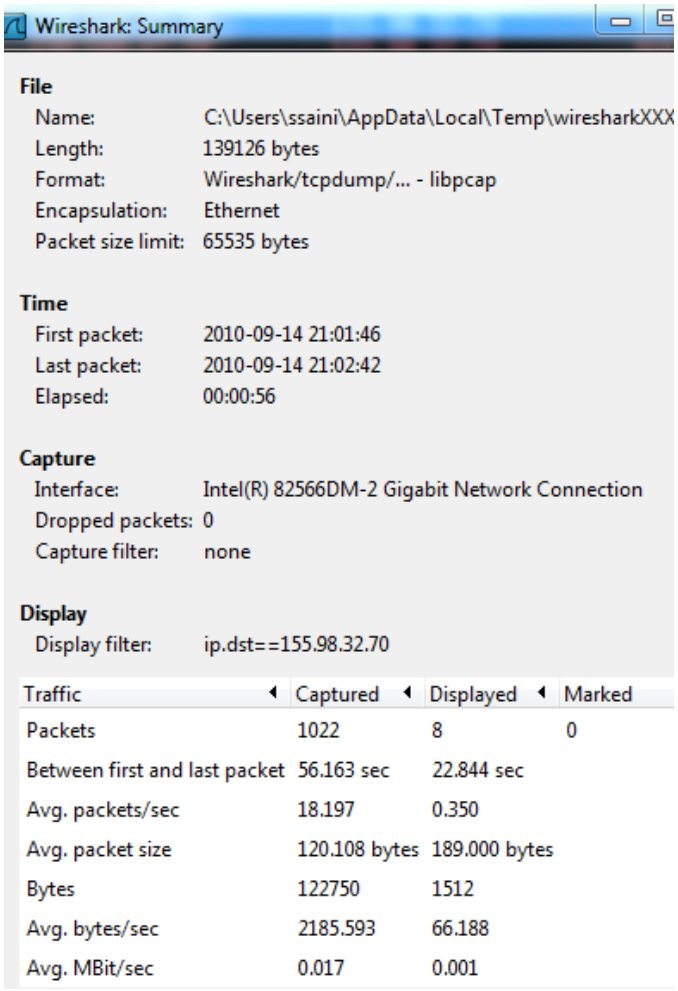

Fig. 8: Wireshark summary for 155.98.32.70- some other Utah node in traffic

```
sshalini@ubuntu:~/protogeni/test$ ssh shail01@pc24.emulab.net
[shail01@pc24 ~]$ ls
Bat.txt  DieWithError.c     ResolveName.c  TCPEchoClient.c
cli      HandleTCPClient.c  srv            TCPEchoServer.c
[shail01@pc24 ~]$ ./srv 12345
Handling client 155.98.36.25
```

```
sshalini@ubuntu: ~/protogeni/test
File  Edit  View  Terminal  Help
Received: Hello pc24
Received: Hello pc24
Received: Hello pc24
Received: Hello pc24
```

Fig.9: PC24 and PC 25 in ubuntu virtual machine terminal as server and client

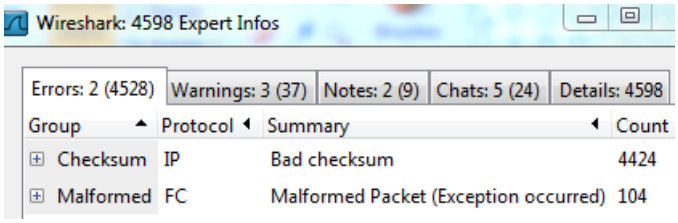

Fig.10: Wireshark expert info for PC 25 as client

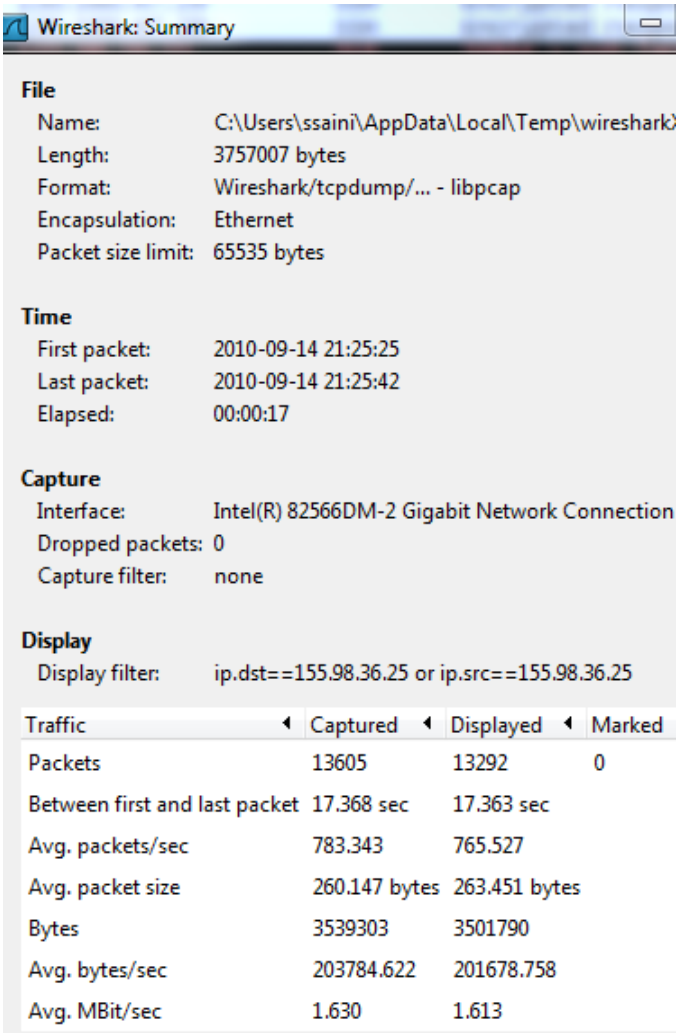

Fig. 11: PC24 as server and PC 25 as client in an infinite loop of sending and receiving message

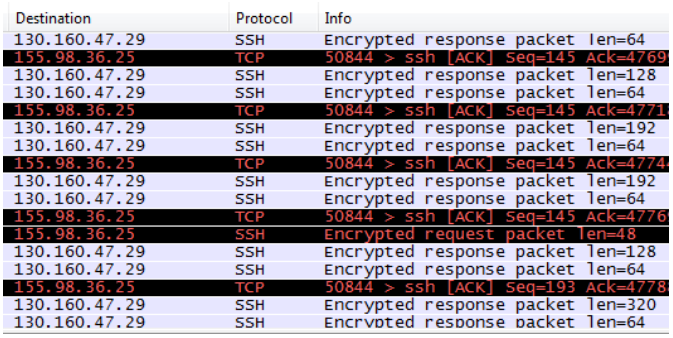

Fig. 12: Variations in Encrypted response packet when source was PC 25

4.5 *ProtoGENI Experiment for Wireless Traffic Capturing Capabilities*

A rspec file was created to request two wireless nodes. From Emulab account node status details, we identified another set of available wireless nodes located nearby. A new slice wl3 was created with wireless nodes. Pc21 and pc22 were acquired to capture and observe the wireless traffic in ProtoGENI.

One of the observation before capturing the traffic was to login to the remote ProtoGENI nodes without storing the passphrase on local machine. As per our earlier work on test scripts, we applied rememberpassphrase.py and forgetpassphrase.py to enhance the host security a bit. We verified that time that slice or sliver operations will ask passphrase at every step if password file does not exist on local machine in a particular place but we did not try to login on remote nodes with already existed and active slices/slivers. From this observation , we can say that ProtoGENI user can create the slice and slivers with passphrase files and then can remove the passphrase file until further use of password file in slice/sliver related operations like renewing or updating slivers.

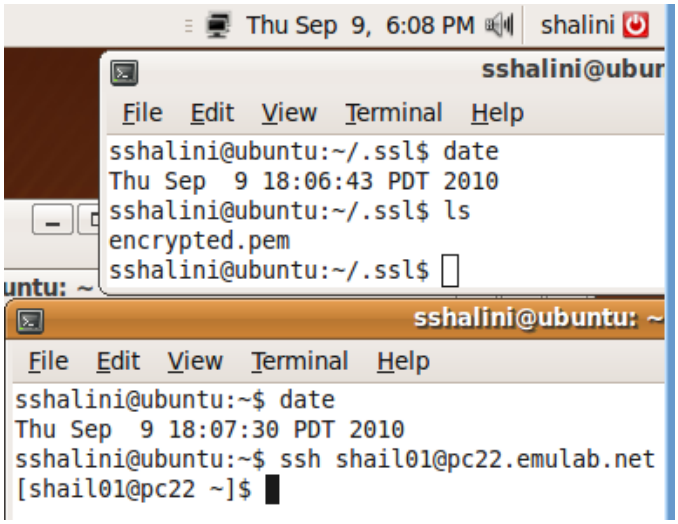

Fig.13: Login to remote ProtoGENI nodes without passphrase file on local machine

To capture and observe the traffic between wirelees nodes in ProtoGENI, we applied similar approach to Emulab experiment. First we captured the traffic between wireless nodes pc21 and pc22 when pc22 pings pc21, and then client-server programs were uploaded to both wireless nodes to analyze traffic between both while client keep sending and receiving the messages from server in an indefinite loop.

1. Pc 22 pings pc2

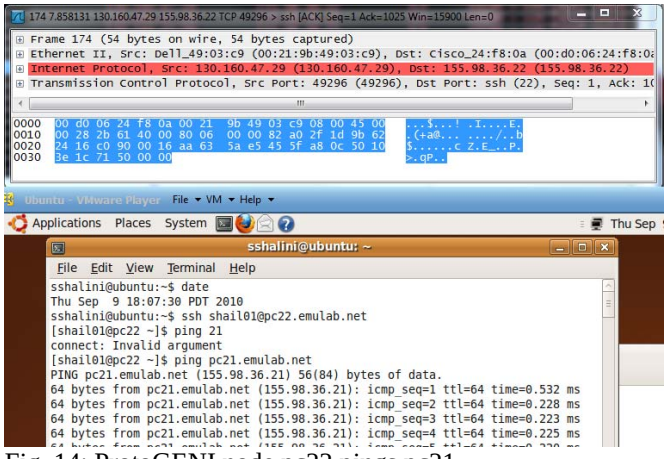

Fig. 14: ProtoGENI node pc22 pings pc21

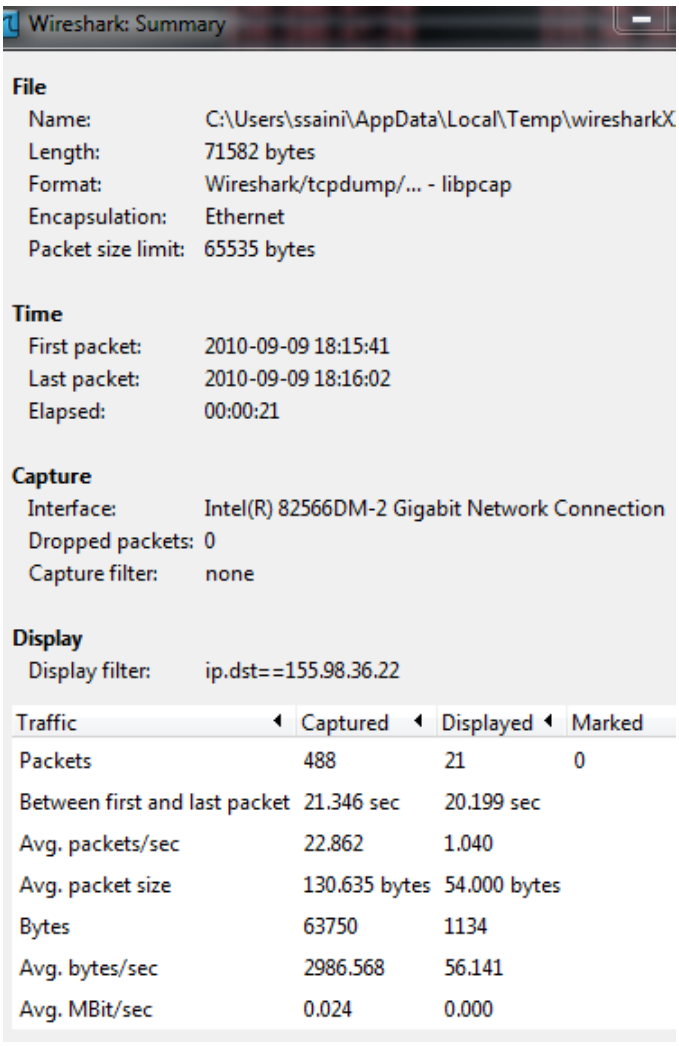

Fig. 15: Wireshark summary when pc22 pings pc21

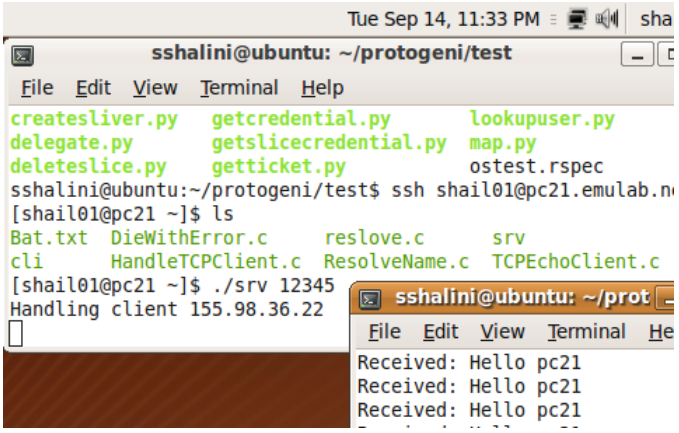

Fig. 16: ProtoGENI wireless nodes Pc21 and pc22 as server and client

Traffic captured during the client server communication between pc21 and pc22 was analyzed as whole and also filtered for pc22 to see the details closely. Different sequence of packets sent and various lengths were shown in wireshark captured traffic. We can see details from the following pictures:

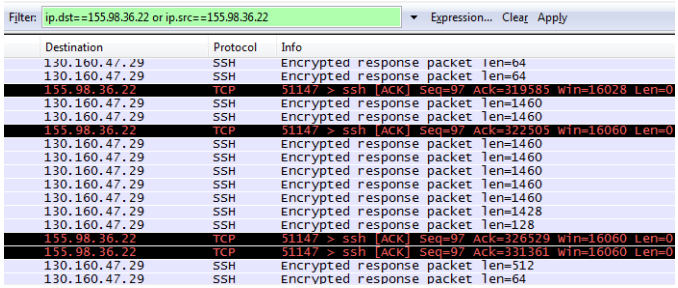

Fig. 17: Wireshark details filtered for pc22

File

Name:	Z:\Fall 2010\thesis\emulab wireless traffic monit
Length:	4425902 bytes
Format:	Wireshark/tcpdump/... - libpcap
Encapsulation:	Ethernet
Packet size limit:	65535 bytes

Time

First packet:	2010-09-14 23:29:40
Last packet:	2010-09-14 23:29:59
Elapsed:	00:00:18

Capture

Interface:	Intel(R) 82566DM-2 Gigabit Network Connection
Dropped packets:	unknown
Capture filter:	none

Display

Display filter:	ip.dst==155.98.36.22 or ip.src==155.98.36.22

Traffic	Captured	Displayed	Marked
Packets	14577	14273	0
Between first and last packet	18.365 sec	18.339 sec	
Avg. packets/sec	793.752	778.283	
Avg. packet size	287.621 bytes	291.858 bytes	
Bytes	4192646	4165690	
Avg. bytes/sec	228299.368	227148.129	
Avg. MBit/sec	1.826	1.817	

Fig. 18: Wireshark summary of traffic details involving pc22

Display

Display filter:	none

Traffic	Captured	Displayed	Marked
Packets	14577	14577	0
Between first and last packet	18.365 sec		
Avg. packets/sec	793.752		
Avg. packet size	287.621 bytes		
Bytes	4192646		
Avg. bytes/sec	228299.368		
Avg. MBit/sec	1.826		

Fig. 19: Wireshark summary of total traffic captured during client-server communication beteen pc21 and pc22

From fig. 18 and fig. 19, we can see that the traffic was dominated by client server communication between wireless ProtoGENI nodes pc21 and pc22.

Again, traffic has some other elements from other sources as shown in fig 20.

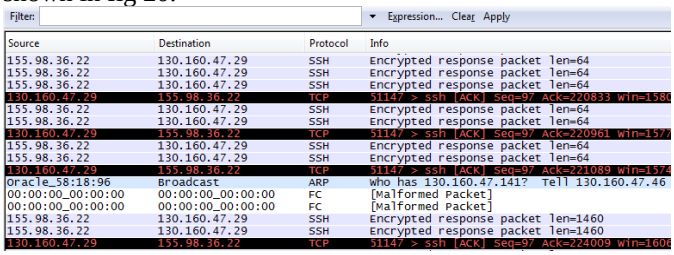

Fig. 20: A sequence of details from the total captured traffic during client-server communication

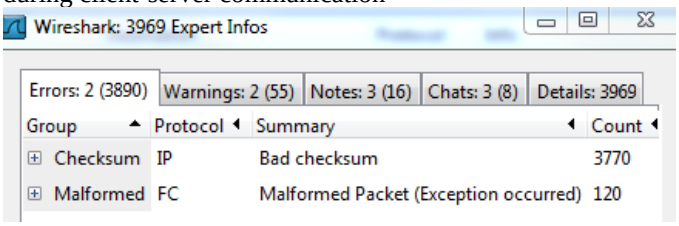

Fig. 21: expert info for total traffic with pc21 and pc22

4.6 Conclusion, Main Observations and Concerns

4.6.1 Emulab configurations:

1. Physical nodes, links, locations are used. Thus, emulated delay does not apply. Packet loss comes from natural wireless losses (no emulated loss is allowed). Bandwidth test gives mixed results.
2. More to test on the Emulab wireless multihop.
3. On-line control experiment: One way to control experiment on-line is from users.emulab.net using link_config script. However, the link_config was not found in users.emulab.net. Another way is to use XMLRPC to control the experiment from local machine.

4.6.2Emulab wireless experiment

1. Experiment was not swapped in showing the probable cause of not availability of particular wireless nodes defined in ns file. Details were modified twice but did not work while nodes were being shown free in Emulab node status details.
2. Added a fixed node to get the delay node to collect traffic tracing and monitoring details provided by the in-built Emulab options. Files exist there but could not download to local machine as saying "… permission denied".

4.6.3ProtoGENI wireless experiment

1. Need to understand more the interpretation of traffic captured by Wireshark to make more detailed statement about traffic status and problems

4.6.4 Summary

After analyzing details for both Emulab and ProtoGENI captured wireless traffic, it seems numbers of errors are less in ProtoGENI traffic. We have initial details for captured traffic.

It shows different patterns and details about traffic regarding the sequence of operations and packet lengths, errors encountered etc. We need to analyze the information more to interpret certain findings as positive, negative or no impact on overall effect on Emulab or ProtoGENI wireless experiment environment. More experiments are required to assess more about wireless traffic capturing capabilities and problems to outline in Emulab and ProtoGENI wireless traffic.