Wi-Fi Netmon : Performance Observation, Anomaly Detection and Diagnosis in Long Distance Wi-Fi Networks

A Thesis Submitted

in Partial Fulfillment of the Requirements

for the Degree of

Master of Technology

by

Dheeraj Golchha

to the

Department of Computer Science & Engineering

Indian Institute of Technology, Kanpur

July 2007

Certificate

This is to certify that the work contained in the thesis entitled "*Wi-Fi Netmon : Performance Observation, Anomaly Detection and Diagnosis in Long Distance Wi-Fi Networks*" by *Dheeraj Golchha* has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

(Dr. Bhaskaran Raman)
Department of Computer Science & Engineering,
Indian Institute of Technology,
Kanpur-208016.

Abstract

The last few years have seen lot of work for providing network connectivity in remote rural areas. An alternative of wired network has been to use off the shelf 802.11 equipment with high gain antennas to provide network connectivity to rural areas. The attractiveness of this setting comes from cost effectiveness and low power consumption. Althougth there are many such network deployments, a very little has been done for monitoring and management of these networks. An initiative has been taken at IIT Kanpur for monitoring and management of long distance Wi-Fi networks called Wi-Fi Netmon.

In our work, we focus on performance observation, anomaly detection and diagnosis of long distance Wi-Fi networks from central location. We present a centralized, client - server based architecture for monitoring of long distance Wi-Fi networks. We provide a set of experiments such as *packet error rate*, *UDP throughput*, *TCP throughput*, *get configuration* and *set configuration* to observe performance and to retrieve or configure wireless parameter values of links on these networks from central location. We use these experiments to observe performance of links in long distance Wi-Fi networks and detect and diagnose problems in case of poor performance of links.

We focus on detecting and diagnosing problem known to be commonly occuring on these networks. The problems that we consider are *power reset*, *link misalignment*, *insufficient transmit power*, *presence of hardware quirk*, *interference detection*, *packet dropping at receiver* and *MAC level ACK timeout*. We design and implement a debugging algorithm for observing performance and detecting and diagnosing the problems stated here.

We also evaluate our work by creating a wireless link in the lab. We introduce the problems in the link and check whether we are able to detect and diagnose these problems through the debugging algorithm. We successfully detect and diagnose the problems power reset, link misalignment, insufficient transmit power and interference detection.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The last few years have seen lot of work for providing network connectivity in remote rural areas. The challenges of providing connectivity in rural areas come from various reasons. The population density is low, per person income is very low and unavailability or interrupted availability of power. Hence the network should be such that the setup cost, service cost and power consumption should be low. Laying out cables for wired networks to those area is very expensive due to low population density in those areas.

An alternative of wired network has been to use in which off the shelf 802.11 equipment with high gain antennas. Due to commodity usage of these equipment, they are available at very low prices. Their setup involves less cost and less time compared to laying out cables. These equipment are used with small single board computers especially designed for communication purpose having low processing power and low memory[15][17]. These platforms consume low power and can also be powered through batteries in the areas where power is unavailable. There have been some works to further reduce this power consumption[8][9].

There are several such network deployments. Some examples include Digital Gangetic Plains [10] in UP (India), Ashwini Project [3] in West Godavari district AP (India) and some deployments by UCB based TIER group [16] in India, Ghana, Guinnea Bissau and Philippines. One of the deployments, the Ashwini project is shown in Figure 1. This network deployment of Ashwini project is currently being used for providing consultation of experts(agriculture, health, education etc.) to villagers through video-conferencing.

The length of links in these networks vary from O(1km) to O(10km). In this network few links are point to point (P2P) which are achieved using directional antennas on both sides while some are point to multipoint (P2MP) links, which are achieved using sector

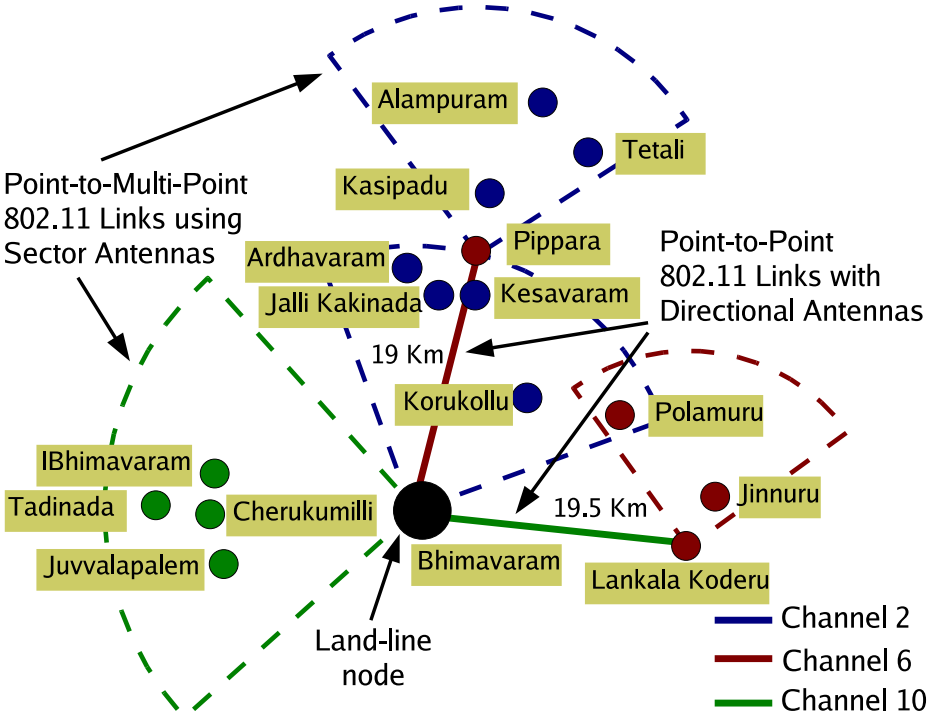


Figure 1.1: The Ashwini network topology

antenna on one node and directional antennas on all the other nodes.

1.1 Motivation & Problem Statement

In [12], authors conducted some experiments on the deployments of Ashwini network for the use of IEEE 802.11g technology. They observed very poor results for the experiments. Some of the links were giving UDP throughput and TCP throughput of as low as 1Mbps. Clearly something is going wrong on those links and the problems need to be detected and rectified. In one of the experimental studies on long distance Wi-Fi networks in [5] also, authors have talked about their experiences and problems faced during experiments. To correct those problems every time some personnel were needed to travel to those far places. The nature of some of those problems suggested that they can be detected and rectified from central location without the need of traveling to those far places, if some central monitoring system is present which could monitor the network continuously. And if the links are not performing properly then it can detect, diagnose and correct the problems. This gives us the motivation for Wi-Fi NetMon.

Along with the different faults occurring on the links, conducting experiments itself has been very tedious and a time consuming job. It not only requires presence of personnel at both the places, but it also requires alternative way of communication between personnel at different nodes to setup experiment parameters and synchronize among them. This gives us a motivation to design and implement a system which can help in running experiments without the need of going to those far places.

Hence our problem statement is to design and implement a monitoring and management system for long distance Wi-Fi network which can:

- help in conducting experiments from a central location without the need of going to locations where links are installed.
- can provide complete the view of network at a central location.
- can detect and diagnose the problem(s) in the network
- can rectify the problem(s), if possible, otherwise notify the network administrator about the problem.

1.2 Solution Approach

The approach we have chosen to detect and diagnose faults is a centralized approach in which it is the duty of a central node to monitor the network and detect, diagnose and correct the problems. In our approach, the central node conducts experiments on the links which are explained in detail in Chapters 3 and 4, analyzes those results to observe whether link is performing well or not and to detect or diagnose the problems. The reasons behind choosing centralized approach is that we want to provide an overall picture of the network at a central location. One major reason behind choosing centralized approach is the nature of one of the problems "interference" that we are detecting. Although we are only detecting this problem and not solving it, but solving this problem will require global view of the network. Solving this problem requires assigning proper channel and transmit power values not only to link concerned but to other links also which can be done optimally only if we have global view of network.

1.3 Assumptions

We have made some assumptions during our work and we state them below.

- During problem detection we conduct some experiments. We assume that although the performance of link is degraded, the link is not broken completely i.e. we are able to run the experiment.
- We have also assumed that RSSI of link in network is stable over a long period and varies in the band of 3-4 dbm only which is also reported in[5].
- We also assume that the MAC protocol being used in the network is TDMA based MAC. This assumption is important while running experiment over point to multi-point link. Since in TDMA based MAC time slot is fixed for all the nodes and they do not contend with each other, we can safely assume that performance achieved at any particular node should be (performance achieved over point-to-point link)/(number of nodes). Hence for problem detection on P2MP link, experiments can be conducted as they are conducted on P2P links.
- We have assumed the values for the different thresholds from the measurement study in[5]. We state these thresholds wherever appropriate.

1.4 Main Results of Evaluation

We conducted evaluation of our work to observe whether we are able to detect the problems that we have considered or not. We created a wireless link in the lab and introduced these problems on the link. We conducted experiments to detect the problems naming power reset, insufficient transmit power, link misalignment, packet dropping at receiver, hardware quirk and interference. We could not conduct the experiments for MAC level ACK timeout because it was not possible for us to introduce this problem in the link.

For all the problems stated above, we successfully detected the problem of power reset and link misalignment. We also detected the problem of insufficient transmit power successfully and observed that packet error rate drops sharply after increasing transmit power. We also detected the interference from Wi-Fi sources in the vicinity of link successfully and observed that most of the Wi-Fi sources in the vicinity were operating between channel 6

and 11 and hence no interference was observed in channel 1. While conducting experiments to detect packet dropping at receiver, we found a bug in our program. Due to this bug, number of packets received at hardware are shown less than number of packets received at higher layer. We are not yet successful in fixing this bug. Due to this bug, we could detect packet drops at receiver only in some cases.

1.5 Thesis Outline

In the rest of the report, we explain some of the similar works and background to our work in Chapter 2. In Chapter 3, we describe possible design choices for network monitoring and our approach of monitoring long distance Wi-Fi network. We describe possible causes of performance degradation in long distance Wi-Fi network and our approach to detect and diagnose those problems in Chapter 4. We, then explain the methodology for evaluation of our work and the results of evaluation in Chapter 5. And, finally we conclude our work in Chapter 6.

Chapter 2

Related Work and Background

This chapter presents some of the similar works that have been done earlier. The works that we present can be broadly categorized into two types of works. While [13], [7], [1] and [11] focus on fault detection, network monitoring and management in wireless networks, [5], [2] and [6] talk about measurements on wireless network. In this chapter well also talk about the background to our work that has been done earlier at IIT Kanpur [14][5].

2.1 Related Work

In this section, we first describe prior work related to network monitoring, fault detection and diagnosis and then we describe prior work related to performance studies.

2.1.1 Network Monitoring, Fault Detection and Diagnosis

In [13], the authors have worked on a monitoring system for infrastructure Wi-Fi networks. They present the mechanisms to determine the root causes of performance degradation at the physical layer and what is the effect of those problems on upper layers of network stack. They talk about the remedy step taken in the 802.11 standard and suggest remedy for each of the faults. Their solution uses sniffers at various places in the network. They collect data from those sniffers and analyze them to infer possible causes of problems. Our work is different from their work in the sense that the nature of networks considered in both cases are different. The network considered in [13] is unplanned 802.11 WLAN's while we have considered well planned long distance Wi-Fi mesh networks. While they use sniffers, we collect data from the nodes in the network itself.

In [7], the authors have designed a monitoring system to monitor the performance of the network. In this work, they focus more on architecture of monitoring system, data collection etc. and very less on wireless properties of the network. While in our work, we focus on monitoring different wireless properties such as packet error rate, RSSI, noise etc. and anomalies on the network, [7] just gives a high level framework about the monitoring system describing data collection etc. and does not describe what to monitor and how to monitor.

In [1], the authors described a technique called "client conduit" for detecting and diagnosing certain faults in 802.11 infrastructure networks. The problems they have approached are detecting RF holes, diagnosing performance problems, detecting rogue AP's and helping a client to recover from authentication problem. Our work is different from their work in context of both the networks as well as faults considered to detect and diagnose. The faults they have considered are particular to the standard of 802.11 protocol in infrastructure mode while the problems we have considered can occur in any wireless network, although the detection and diagnosis approach is designed particularly for the long distance Wi-Fi mesh networks [3] [10].

In [11], authors have used network simulators to detect certain faults in multihop wireless networks. From the difference between the performances between actual network and simulator, they detect the problems in network. They have used this approach to detect packet dropping, link congestion, MAC misbehavior and external noise sources. Although some of the problems that authors have approached in [11] are same as the problems that we are approaching, solution approach is different in both the works. While they have used simulators to generate the performance of network, we do not use simulators and directly perform all the performance related experiments on nodes of the network itself.

Our work is different from all these works in the sense that none of them consider solving problems on long distance Wi-Fi networks. Although [11] considers some of the problems that we are handling, they use simulators to detect the problems. Considering the fact that the behavior of wireless networks depend on many factors simulator may not always be able to produce actual results. We use links in network itself to run experiments and detect problem by analyzing those results. We have shown comparison of our work with all these works in Table 2.1.

WORK	**N/W**	**METHO DOLOGY**	**FAULTS CONSIDERED/ METRICS MONITORED**	**ARCH./ SIMULATION / IMPLEMENTATION**
MOJO [13]	WLAN	Sniffers	• Hidden Terminal • Capture effect • Signal strength variation • Noise	Architecture
VISUM [7]	WLAN	AP	N/A	Architecture / Implementation
Arch. & tech. for diagnosing faults [1]	WLAN	Client	• Locating disconnected clients • Rogue AP detection	Architecture / Implementation
Troubleshooting WMN [11]	WMN	Client	• Packet losses • Link congestion • External noise • MAC misbehavior	Architecture / Simulation
Wi-Fi Netmon	Long Distance WMN	client-server	• Power reset • Packet dropping at receiver • Interference detection • Insufficient transmit power • Link misalignment • MAC level ACK timeout • Hardware quirk	Architecture / Implementation

Table 2.1: Related works: network monitoring & management

2.1.2 Measurement studies in wireless networks

[5] talks about the performance and their experiences of long distance links and the causes of performance degradation on those links. This work serves as background for our work. Our focus was on detecting and resolving those problems faced in[5]. Since our work is based on this work, we describe about it in detail in Chapter 4.

[2] talks about the performance of network and causes of packet loss in wireless mesh networks. They observe that loss rates are stable over time, SNR and distance between the nodes of link have little effect on the loss rates and high loss rates due to multipath. This work is relevant to our work because it gives insight into possible causes of performance degradation in wireless mesh networks.

Some of the results in [2] and [5] are quite contradictory although both the networks have small difference in nature. While network in[2] is unplanned and has lot of buildings between the nodes to cause multi path and attenuation, the network in[5] is well planned and nodes in each link have complete line of sight. [2] finds that interference does not cause significant error rates. On the other hand, [5] says that interference degrades the performance substantially.

In [6], authors try to observe the effect of multiway interference(i.e. cumulative interference from two or more different nodes while they do not cause interference alone) on the particular node of network but they end up observing that very small fraction of nodes in network suffer performance degradation due to multiway interference. We chose this work because it gives us insight into another problem that can occur in the wireless networks, but since results in [6] show that there is only small effect of multiway interference which could be even more less in long distance networks due to length of the links, we did not consider this problem to detect and solve in our work. We have shown summary of all these works in Table 2.1.2.

2.2 Background

One of the students in Computer Science & Engineering department, IIT Kanpur Mr. Rahul Shrivastava worked on conducting experiments on a long distance Wi-Fi link from a remote computer [14] for his B.Tech. project. We would like to explain this work briefly here.

WORK	NETWORK	RESULT(S)
Long Distance links: Performance, measurement & experiences [5]	Long distance Wi-Fi network	• Link stable over time • Strong correlation between pkt error rate and RSSI • Small effect of pkt size on error rate • Link detrimental to interference • h/w quirk found
Link level measurements from 802.11 WMN [2]	Wireless Mesh Network	• Multipath fading is culprit • Small effect of interference • Small loss rate variation over time
Characterizing Multiway Interference in WMN [6]	WMN	Significant throughput degradation to small fraction of links due to multiway interference

Table 2.2: Related works: Experimental/ Characterizing behavior

In this work he designed and implemented a daemon which runs continuously on each of the nodes of link. One of the nodes in the link is connected to remote computer through wired network. These daemons on the nodes are listening for the experiment parameters. Remote computer sends experiment parameters to the node (say node A) with which it is connected through a php script. These experiments parameters are in the form of an XML document. After receiving experiment parameters, node A conveys these parameters another node on the link(say node B). Both the nodes conduct experiment between them. After conducting experiment node B sends results with it to node A. Node A sends results of both the nodes to remote computer through wired network. This can be better understood through Figure 2.1.

There is another earlier work which we have used in our work and would like to explain here briefly. In[5], authors have used some driver modifications in the hostap driver to get low level information which are not passed upto higher level. They make use of */proc* file system to pass on low level details. */proc* file system is virtual file system residing in memory used in Linux and is used for communication between user space and the kernel. Whenever there is a call to read a file in */proc* file system, a system call is made to read the information from kernel buffer and the corresponding information is returned to user. This functionality is implemented in hostap driver such that whenever a packet is received,

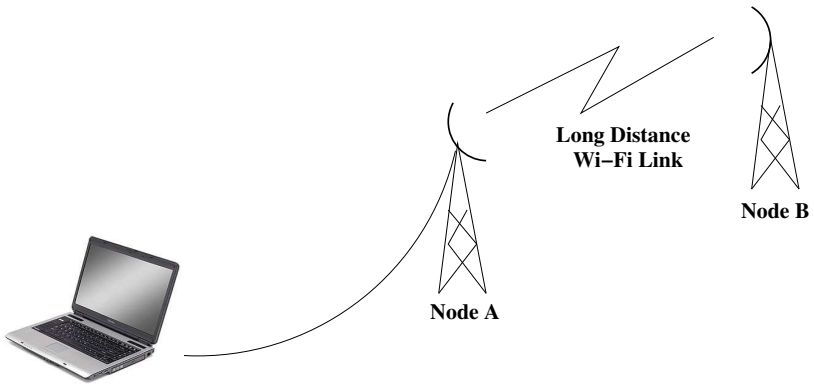


Figure 2.1: Conducting experiments on a long distance link through remote computer

details about that packet is stored in driver buffer before passing on that packet to higher layers. This information can be accessed later through proc file system. Figure 2.2 explains this functionality appropriately.

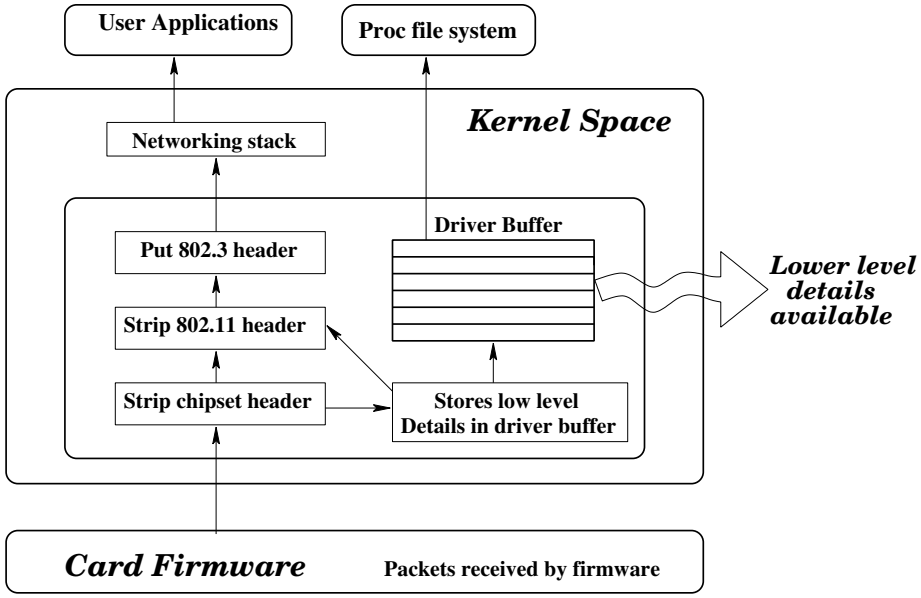


Figure 2.2: Modifications in hostap driver to access low level information at user level
Image Source: [4]

From this modification, following informations are available to us at user level

1. t = time at which packet is received
2. ts = timestamp
3. fty = frame type
4. fsty = frame subtype
5. ln = length
6. mty = message type field

7. sq = sequence no.
8. er = CRC error or not
9. sl = silence value
10. sg = RSSI value
11. rt = transmit rate at which packet received
12. rxf = rx flow
13. addr1 = Destination address
14. addr2 = source address
15. addr3 = Receiver address(BSSID)
16. addr4 = Transmitter address

While running experiments, nodes also send some of these information along with the results to the server which is required to detect the problem. For instance, to detect poor SNR, server need to know the RSSI and noise values for the received packets. Hence these values of RSSI and noise floor are used from meas log to calculate SNR.

Chapter 3

Network Monitoring: Overall Approach

Network monitoring, as the name implies, is a system to monitor a deployed network. The network may be monitored for many different reasons. For instance, it can be monitored to provide the overall picture of the network, in terms of performance or topology. It can be monitored to see if some node or link is not working properly or is down. It can also be monitored to see if some part of network is over utilized (congested) while some other part is under utilized.

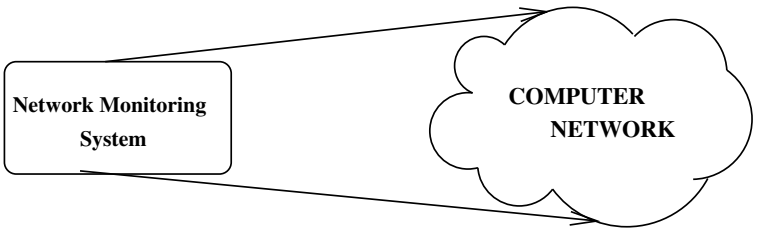


Figure 3.1: Network Monitoring

We have designed a network monitoring system for long distance Wi-Fi network. Our objective of designing a network monitoring system was to get the overall picture of network at the central location in terms of performance for each link. It also provides us the ease of conducting experiments on these networks from a remote location, such remote experimentation has been a very tiresome manual job thus far. There are many problems in long distance Wi-Fi networks which can be taken care from central location without the need of going to the actual locations of installation. We seek to detect and diagnose such problems.

Hence our problem statement is to design and implement a monitoring and management

system for long distance Wi-Fi network which can:

- help in conducting experiments from central location without the need of going to locations where links are installed.
- can provide a complete view of network at a central location.
- can detect and diagnose any problem(s) in the network
- can rectify the problem(s), if possible, otherwise notify the network administrator about the problem.

3.1 Design Choices

There are primarily two ways to monitor a network :

- **Active Monitoring:** In active monitoring, we inject the traffic in the network to get the performance related information. In this case we specify in which ways to inject the traffic in the network in terms of duration, packet size, packet interval etc. This is performed preferably when the network is idle with no regular traffic in the network.
- **Passive Monitoring:** In passive monitoring, we do not inject traffic by ourselves. We monitor the regular traffic at different times and intervals and monitor the performance.

Our work comprises of Active Monitoring while Passive Monitoring has been approached in another work [4].

Another dimension of design choice in network monitoring is where the control resides. Again there are two design choice:

- **Centralized:** In this approach, the network administrator, sitting at a central location monitors the network. If he observes that some link is not performing according to the expectation then he tries to figure out that what is the problem which is causing the link to perform poorly. After figuring out the problem, he can take certain measures to make sure that link performs well.

- **Distributed:** In this approach, every node in the network, itself observes the performance on its own. If there is some problem, then it tries to resolve the problem by taking some measures on its own.

Both the centralized and distributed approaches have their own advantages and disadvantages. While in centralized approach, the network administrator can have the picture of whole network at a single point, it also adds overhead in terms of network traffic. Similarly in distributed way, although the nodes can take care of the problems locally and there is no overhead, the nodes have only the local picture of network and they can not take care of problems caused by the other links of network.

We have chosen centralized way of network monitoring due to the following reasons:

- Since conducting experiments on the long distance Wi-Fi network has always been a very tiresome job, we wanted to provide a way to conduct experiment on these networks from remote locations without the need of going there.
- We wanted to provide overall picture of network at the central location to the network administrator.
- Some problems in the network that we are approaching such that interference detection and measurement need the global view of network instead of local view to solve the problem.

3.2 Architecture of Wi-Fi Netmon

Our system, Wi-Fi Netmon, works on a client server model and centralized approach of monitoring where the central node works as a server and all the other nodes work as clients. All the analysis and decision making functions are performed by the central server. The server can conduct performance related experiments on the clients and observe the results of those experiments. If it observes that the link is not performing according to the expectation (here expected performance can be defined in terms of minimum UDP or TCP throughput required for the application - being used on the network, to run properly), then it tries to diagnose the problem. If it can correct the problem, it corrects it so the link can perform well, otherwise it informs the network administrator at central location about the problem.

Figure 3.2 explains the architecture of Wi-Fi Netmon briefly while Figure 3.3 explains the working of client - server model in Wi-Fi Netmon.

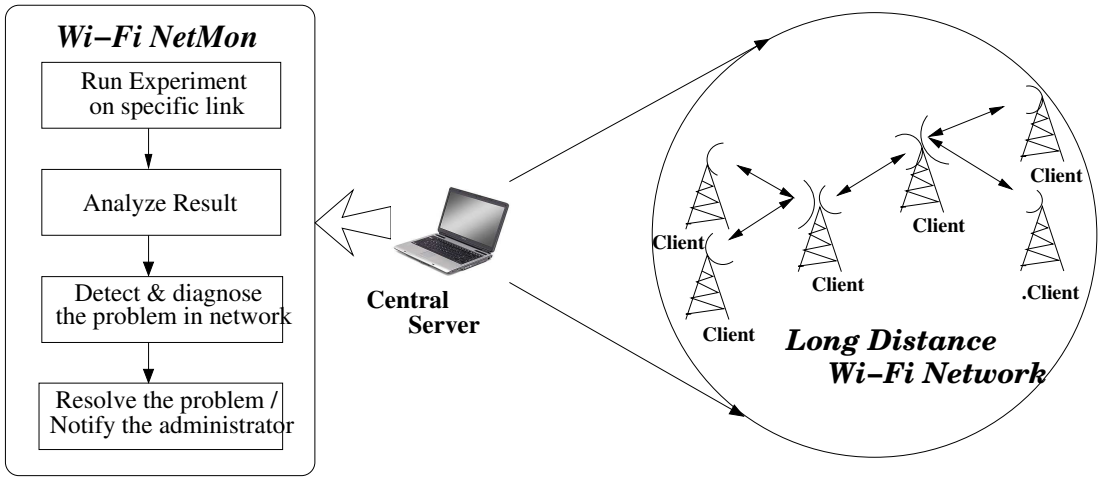


Figure 3.2: Architecture of Wi-Fi Netmon

In the following subsections we explain the working of the client and the server. We shall explain the working of server side briefly in this chapter. We describe the problems that can occur in long distance Wi-Fi networks and our specific approach for detecting and diagnosing each of those problems in detail in the next chapter.

3.2.1 Server

All the clients are listening for the experiment parameters from the server. The central server eventually sends the experiment parameters to the client in form of an xml document[18]. We have chosen xml format to send experiment parameters due to following reasons:

- It provides easy human readable structured format.
- We can define our own tags according to our requirements.
- And most programming languages provide the library functions to process xml documents which makes implementation of the server independent of the implementation of the client.

Since experiments are being conducted on a link, the XML document specifies parameters for the wireless radios at both ends of the link. Experiment parameters for both

the radios are categorized as experiment parameters and wireless parameters. Experiment parameters give the details about parameters specific to the experiment while wireless parameters specify the parameters for wireless interface settings. For better understanding, we have included sample xml file for experiments in Figure 3.4.

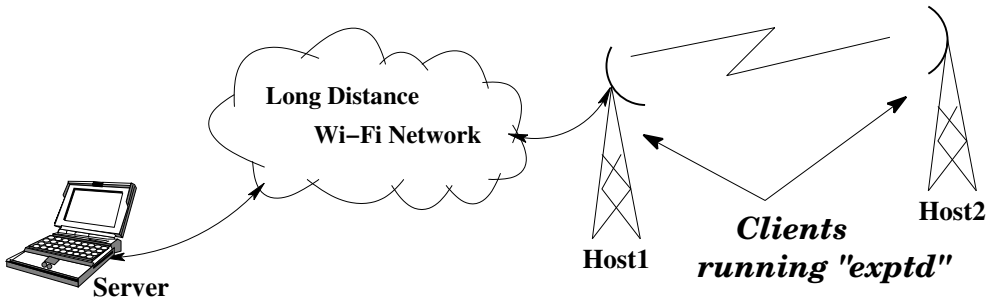


Figure 3.3: Function of Clients and Server

As seen in Figure 3.4, different tags reveal different informations. For instance, tag *expt_info* tells which experiment has to be conducted. Tags *host1* and *host2* tell the experiment parameters for host1 and host2. Similarly each of the tags *host1* and *host2* contain the tags for experiment parameters (*expt_params*) and wireless parameters (*expt_wless_params*).

The client, on receiving the experiment parameters, conducts the experiment on the specified link and sends back results to the central server. While conducting the experiments, it uses the wireless settings as specified. Server after receiving the results analyzes those results to observe the performance of link and takes the certain measures to detect and diagnose the problem if the link is not performing well. Figure 3.3 explains the functions of server and clients briefly, while Figure 3.2 explains the steps performed by server.

3.2.2 Client

All the clients run an experiment daemon, which we call *exptd*. This daemon *exptd* is continuously listening for the experiment parameters to be sent from the server. When exptd receives experiment parameters, it conducts experiment on the link. After conducting the experiment it sends back results to server again in form of xml document. The working of *exptd* daemon can be understood through the state diagram in Figure 3.5.

As shown in Figure 3.3, there are two nodes on the link called them host1 and host2. Lets say host1 receives experiment parameters from server, and host2 receives experiment parameters from host1. Initially, the client daemon is in LISTENING state, listening for

```
<?xml version="1.0" encoding="UTF-8"?>
<comm_msg>
        <host_id>PHP_HOST</host_id>
        <msg_type>RECV:XML_EXPT_DATA</msg_type>
        <msg>
                <expts>
                        <expt_info type="UDP_THPUT" id="UDP_THPUT_4RT56Y">
                                <host1 ip="192.168.1.1">
                                        <expt_wless_params>
                                                <wless_mode>master</wless_mode>
                                                <wless_channel>1</wless_channel>
                                                <wless_txrate>4</wless_txrate>
                                                <wless_txpower>128</wless_txpower>
                                                <wless_essid>voyage</wless_essid>
                                                <wless_ip>10.0.0.1</wless_ip>
                                        </expt_wless_params>
                                        <expt_params>
                                                <mode>-s</mode>
                                                <port>60000</port>
                                                <ngbr_ip>10.0.0.2</ngbr_ip>
                                                <hbint>10</hbint>
                                                <bdcst>yes</bdcst>
                                                <duration>20</duration>
                                                <packet_size>1400</packet_size>
                                        </expt_params>
                                </host1>
                                <host2 ip="192.168.1.2">
                                        <expt_wless_params>
                                                <wless_mode>managed</wless_mode>
                                                <wless_channel>1</wless_channel>
                                                <wless_txrate>4</wless_txrate>
                                                <wless_txpower>128</wless_txpower>
                                                <wless_essid>voyage</wless_essid>
                                                <wless_ip>10.0.0.2</wless_ip>
                                        </expt_wless_params>
                                        <expt_params>
                                                <mode>-r</mode>
                                                <port>60000</port>
                                                <ngbr_ip>10.0.0.1</ngbr_ip>
                                                <hbint>10</hbint>
                                                <bdcst>yes</bdcst>
                                                <duration>20</duration>
                                                <packet_size>1400</packet_size>
                                        </expt_params>
                                </host2>
                        </expt_info>
                </expts>
        </msg>
</comm_msg>
```

Figure 3.4: Example XML Experiment Parameters File

experiment parameters on both the nodes. When host1 receives connection request from server , it goes to CMD_CONN state and starts receiving experiment parameters. When experiment parameters are received successfully, it goes to CMD_RECVD state. After receiving the experiment parameters, host1 establishes a TCP connection with host2 to send experiment parameters to host2 and goes to CMD2PEER state. When experiment parameters are conveyed successfully to host2, host1 changes its state to RUN_EXPT.

When host2 gets the connection request from host1, it goes to CMD_CONN state and starts receiving experiment parameters. After receiving experiment parameters, host2 changes its state to CMD_RECVD state. When host2 is in CMD_RECVD state, it directly transitions to RUN_EXPT state.

Once both the nodes are in RUN_EXPT state, they conduct experiment between themselves. When experiment is completed, both the nodes go to EXPT_DONE state. Even if the experiment was unsuccessful, both the nodes transition to state EXPT_DONE. After this, host1 tries to establish connection with host2 to get the results and goes to RES_FROM_PEER state. When the results are received from host2 or there is timeout, host1 goes to RES_DONE state. After receiving the results, host1 sends them to the server. When the results are sent to server successfully or there is timeout, host1 goes back to LISTENING state.

After completing the experiment, host2 goes to LIS4RES_CONN state and waits for the connection request from host1. When it receives connection request from the host1, it goes to RES2PEER state and sends results to host1. After sending the results to host1, host2 goes to LISTENING state. If host2 does not get the connection request from the host1 in certain time frame then it directly transitions to LISTENING state.

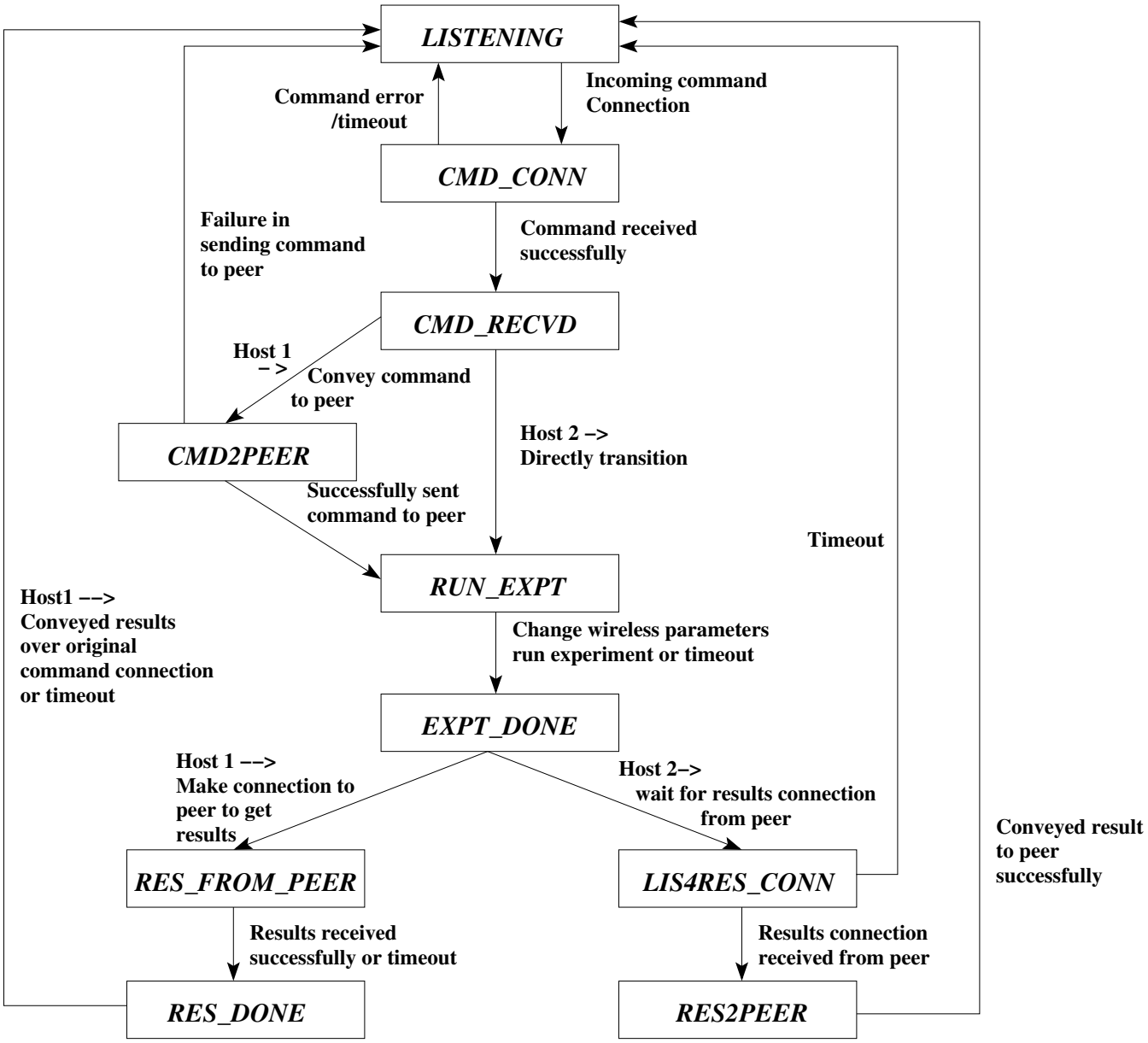


Figure 3.5: State diagram of exptd daemon running at clients

Chapter 4

Performance Observation, Problem Detection & Diagnosis

This chapter presents details on how we are going to observe the performance of link in the network, we also talk about the problems faced or experienced by authors of [5] while doing experiments on the long distance wifi mesh network and finally we talk about how we are going to address those problems.

4.1 Experiments for Performance Observation

To observe the performance of the link on the network, we have following experiments: Packet error rate, UDP throughput and TCP throughput. These experiments were used in earlier work [14] which we have restructured and reimplemented to integrate with our tool and added some more functionalities in them which we shall talk about later in the chapter. We have also designed new experiments get configuration and set configuration. We a detail about all these experiments.

- **Packet Error Rate:**

  This experiment creates UDP sockets on both the nodes. One node acts as a sender while the other one as a receiver. Experiment parameters specify the number of packets to be sent, size of each packet and time interval between sending 2 consecutive packets. The sender sends the specified number of packets while receiver tries to receive those packets. After the experiment, receiver reports the packet error rate. It also reports certain informations about packets received both at UDP layer as well as

hardware level information such as number of packets, average signal, average noise etc.

After conducting the packet error rate experiment, both the nodes are kept in monitor mode for certain period. In the monitor mode, they try to listen for the packets from other Wi-Fi equipments in the vicinity. After coming out of monitor mode, both the nodes report the mac addresses, average signal and average noise of packets received from Wi-Fi sources in their vicinity.

- **UDP Throughput:**

  This experiment also creates UDP sockets on the the nodes. In this experiment, duration of experiment is specified for which sender sends the packets, instead of number of packets. After the experiment, receiver reports the UDP throughput achieved.

- **TCP Throughput:**

  In this experiment, a TCP connection is created between both the nodes. The sender sends the packets to receiver for specified amount of time. After the experiment the receiver reports the TCP throughput achieved.

- **Get Configuration & Set Configuration:**

  These experiments are used to retrieve values of different wireless parameters set at the nodes and to set the parameters values at the nodes. These parameters include ip address, essid, operating mode (master-managed, ad-hoc), operating channel, transmit power, transmit rate, pseudo ibss mode (special mode of hostap driver in which management frames are not sent), alc (to turn on or off automatic level control), beacon interval etc. A parameter file is maintained at all the nodes. We assume that all the parameter values are set through Wi-Fi Netmon which makes sure that values of wireless parameters set at the nodes are reflected in the parameter file.

4.2 Problem Detection & Diagnosis

In [5], several performance studies of long distance links are presented. The authors list various problems that can normally occur on long distance Wi-Fi links which can degrade the performance of the link. Since these links are in remote rural areas and there is no

availability of experts in these areas, currently the only possibility to resolve these problems is to travel to those areas and diagnose the problem. The characteristics of the problems suggested that these problems can be taken care of without the need of going to remote places where the links are installed, if there is a presence of some sort of central coordinating system. Such central coordination is the functionality of our Wi-Fi Netmon system.

In our work we have detected, diagnosed and resolved those problems from the central location without the need of going to remote villages. If it is not possible to resolve the problem through central location than we notify the network administrator about the problem. In the following subsections, well talk about these problems and our approach to detect and resolve them.

- **Power Reset:**

  Many times the authors found that the transmit power on those links automatically got reset to a default value and thus decreasing RSSI at the receiving end. As mentioned earlier, we maintain a file for wireless parameter values at every node. We assume that the parameter values are changed through Wi-Fi Netmon tool which makes sure that after changing the parameter values, the parameters value file is updated with new value, i.e. the values of parameters are always reflected by the parameter file.

  Now if the transmit power at node gets reset automatically, the value in parameter file will not be updated. We have designed a experiment get_ config which retrieves all the parameter values written in file (which we call default_ config) and parameter values which are currently set at both the nodes on the link (which we call current_config). Although we are only using transmit power value, this experiment can be useful to retrieve all relevant information about the link. If the transmit power value in default_ config is different from the value in current_ config, we conclude that power has got reset.

  To deal with this problem, we have designed a command which works in similar way as the experiments (we call it set_ config) and sets all the parameters with the values from parameter file including the transmit power and hence removing the problem of power reset. This experiment can also be used for setting new parameter values on the nodes of the link which we shall describe later.

- **Insufficient Transmit Power:**

There is another case when RSSI at receiver is low due to which the packet error rates are high. If the transmitter is not transmitting at maximum power than RSSI can be improved by increasing transmit power and thus packet error rate can be decreased.

To deal with this problem, we first run packet error rate experiment with current transmit power to see whether the packet error rate is low or not. If the packet error rate is high, then we need to diagnose the problem. we first check whether the transmitter is transmitting at the maximum power. If the transmitter is not transmitting at maximum power, we repeat the packet error rate experiment with maximum transmit power. If the packet error rate with maximum transmit power is low than we can conclude that link was performing poorly due to insufficient transmit power. To correct this problem we use a command set_config which we described earlier, to set the transmit power to the maximum value.

- **Link Misalignment:**

  If there is link misalignment then RSSI will be lower than the RSSI received earlier. We maintain the history of RSSI values received at each node. We compare current RSSI measured in packet error rate experiment conducted with current transmit power with RSSI history. If current RSSI is significantly lower than RSSI history then we conclude that link might have been misaligned.

  It is also possible that RSSI value might have gone down due to power reset. To get away with this possibility we first check for the power reset and then check for link misalignment.

  Since correcting this problem requires aligning the antennas manually, it is not possible to correct this problem from the central location. Hence we notify the network administrator about the possibility of link misalignment.

- **Hardware Quirk:**

  Many times it was observed that even at higher SNR, packet error rate was not 0%. On close examination [5] found that these losses were due to packets received with CRC errors. They found that the RSSI for these CRC error packets was significantly lower than other packets (about 15 dB). This was due to a hardware quirk in the senao pcmcia cards.

To detect this problem, we have again used kernel level information passed to the user level. If the packet is received with CRC error than the value of field er is marked 1. We check for those er=1 packets. If the RSSI values for these packets is significantly lower than the other packets, we conclude that there is presence of hardware quirk.

Again, since this problem can not be rectified, we inform the network administrator about the problem.

- **Packet Dropping at Receiver:**

  One of the problems that was observed in [5] was that when the packets were being transmitted at higher transmit rate, the soekris board was not able to handle those packets due to the low processing power of the soekris board. This resulted in packets being dropped at higher layer at the receiver even though they were received at the radio hardware.

  This problem can be detected, if we can get the information about the number of packets received at hardware and number of packets received at upper layer. To get the information about the number of packets received at hardware, we count the number of packets received at meas log. We already have the count of packets at higher layer. If we see a significant difference between the number of packets received at upper layer and number of packets received at hardware, we conclude that hardware is not able to handle packets at higher transmit rate.

  To rectify this problem, we again check the packets drops at lower transmit rates. If the hardware is able to handle packets, we change the transmit rate of link to lower transmit rate. We use experiment set_config to change the transmit rate of the link to lower value.

- **MAC Level ACK Timeout:**

  MAC level ACK timeouts occur on long distance links when length of link is too long. In [5] MAC level ACK timeout was observed on link of length 37 km. In presence of MAC level ACK timeouts, packets will be retransmitted again and again, resulting in lot of duplicate packets in meas log. To detect this problem, we observe the meas log trace. If we observe the higher percentage of duplicates in the MAC sequence numbers of packets received, we can conclude that MAC level ACK timeouts are occurring.

Upon detection of MAC level ACK timeout, we notify the network administrator about it.

- **Interference Detection:**

  It was observed in [5] that in the long distance links, the presence of interference deteriorates the performance gradually. Hence there is definitely a need to handle the interference. Interference can be either from the Wi-Fi equipment operating in same frequency as the link or from the non Wi-Fi equipment generating radio signals of same frequency range as Wi-Fi equipments. Some of the equipments known to be generating frequencies of same range are microwave ovens, elevators etc. Since long distance links operate at height of about 40 meters, we hope that there will not be interference from non Wi-Fi equipment at that height but still we are handling that case in our algorithm.

  - **Interference from Wi-Fi equipment:**

    To detect the interference from Wi-Fi equipment, we are using monitor mode working of hostap driver. In monitor mode, hostap driver passes all the packets received to higher layers irrespective of whether the packet is destined for it or not. This monitor mode working is being used in packet error rate experiment in which both the nodes of the link are put in monitor mode for some time at the end of experiment. We record all the received packets at meas log, if packets are being received from other MAC addresses along with the packets from the links nodes mac addresses then there is an interference in the link.

  - **Interference from non Wi-Fi equipment:**

    Since interference from non Wi-Fi equipment is not detected as any MAC packet, we can not use monitor mode working of hostap driver to detect interference from the same. The interference from the non Wi-Fi equipment is in the form of energy being generated at the same frequency band. We can use the noise value for the packets received in packet error rate experiment to detect interference from these equipments. Noise values are signal values recorded just before the packet is received. If the noise floor is high than we can conclude that there is an interference from the non Wi-Fi equipment.

Correcting interference will not be a trivial job, it may require change in channel and

transmit power values not only in the same link but in the other links of network also. In our work, we have not focused on correcting interference and we are just notifying the network administrator about the presence of interference. Mr. Akhilesh Bhadauria has worked on interference management on the long distance links [4].

Detecting and diagnosing all the above problems can be explained briefly through the flow charts in Figures 4.1, 4.2 and 4.3. Detection of problems on long distance Wi-Fi links is performed in three steps. In first step, we detect problems naming power reset, hardware quirk, insufficient transmit power, link misalignment and interference. In second step we detect packet dropping at receiver and in third step we detect MAC level ACK timeout.

In first step we first try to detect power reset. We get the configuration of link at server. If default configuration of transmit power is different from current configuration then there is a power reset. We use command set_config to set transmit power back to default value. After checking power reset, we conduct packet error rate experiment with current transmit power and transmit rate and packet size 1400 bytes and inter packet interval of 20 ms. After receiving results of packet error rate experiment, we check for presence of hardware quirk. If average RSSI of CRC error packets is below threshold then we notify the network administrator about the presence of hardware quirk. Now we check for link misalignment, if average RSSI for packets received in packet error rate experiment is lower than RSSI history then there is possibility of link misalignment. We notify network administrator about the link misalignment.

Now we check for insufficient transmit power case. If average SNR for packets received in packet error rate experiment is lower than SNR threshold for the transmit rate and transmit power currently set at the link is lower than maximum transmit power then we repeat packet error rate experiment with maximum transmit power. If packet error rate with maximum transmit power is less than threshold than the transmit power set at the link is insufficient, we set the transmit power of link to maximum transmit power.

At the end of step 1, we check for presence of interference. We first check for interference from Wi-Fi sources in vicinity. If other MAC addresses are present in the results of packet error rate experiment from the monitor mode experiment then there is presence of interference from Wi-Fi sources in the vicinity. If other Wi-Fi sources are present in the vicinity of link, we notify the network administrator about their presence and also notify

their MAC addresses and average RSSI received from them.

In the step 2, we check for packet dropping at receiver. We conduct the UDP throughput experiment with 11 Mbps transmit rate, 100 bytes packet size, small inter packet interval and large number of packets. If number of packets received at meas log are more than number of packets received at UDP layer then hardware is not able to handle packets at such a high rate and is dropping packets between hardware and driver. We notify the administrator about the presence of packet dropping at receiver.

In step 3, we conduct TCP throughput experiment. If there are large number of duplicate packets received at meas log then we notify the administrator about the presence of MAC level ACK timeout and that the length of link is too large.

4.3 Thresholds used in Algorithm

In this section we state threshold values used for different metrics in debugging algorithm. Some of these threshold values have been taken from [5]. We have chosen threshold for RSSI to be -75 dBm for the average RSSI of packets received with CRC error to detect hardware quirk. Value of RSSI history will be different for different links depending on the RSSI values received in past for those particular links. In our experiments, we found that packet error rate does not exceed 3 - 4 %, if link is performing well. Hence we have chosen threshold for packet error rate as 4%. Thresholds values for SNR are taken from [5] which are 10, 9, 8 and 7 dB for transmit rates 11, 5.5, 2 and 1 Mbps respectively.

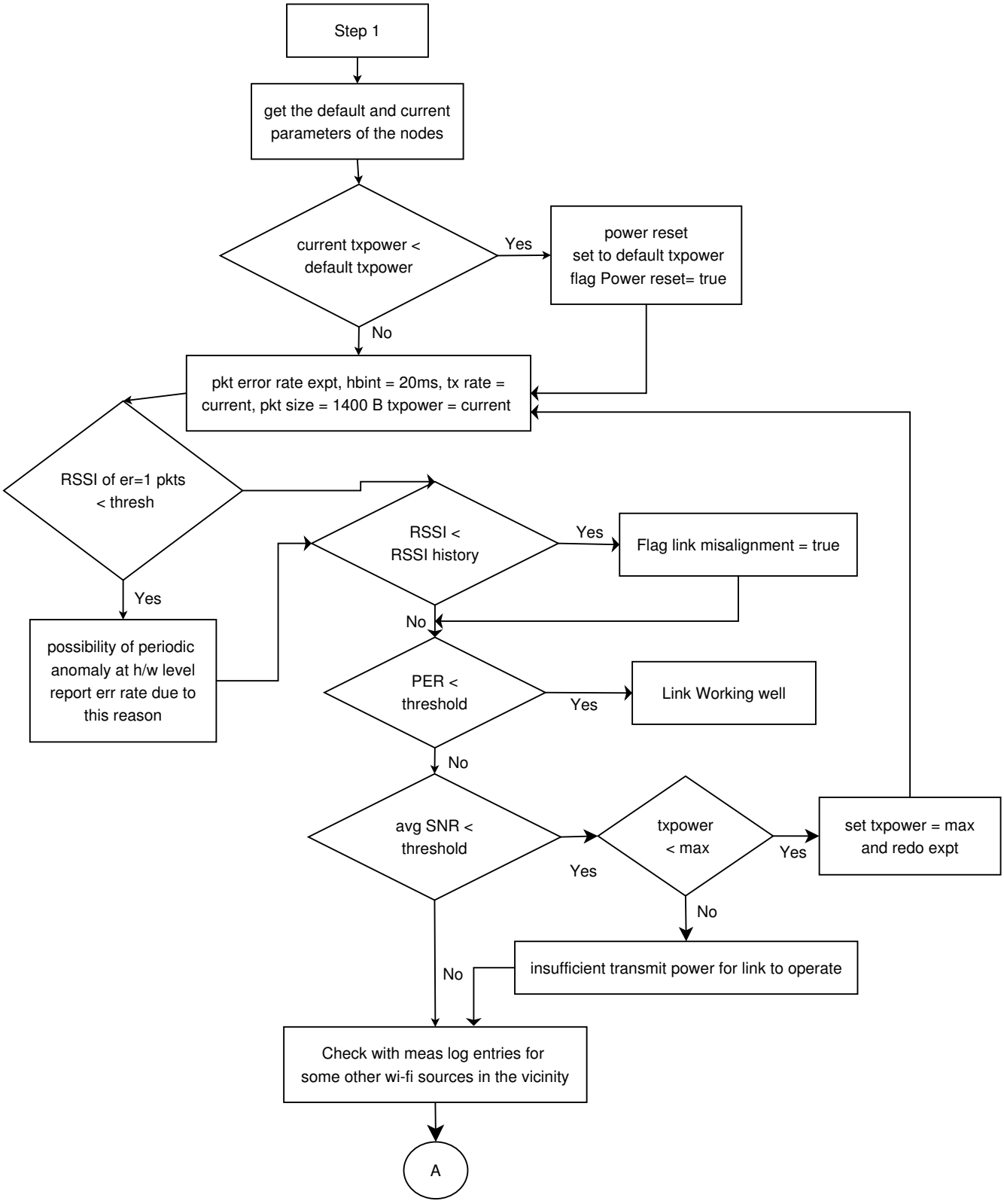


Figure 4.1: Debugging Algorithm : Step 1

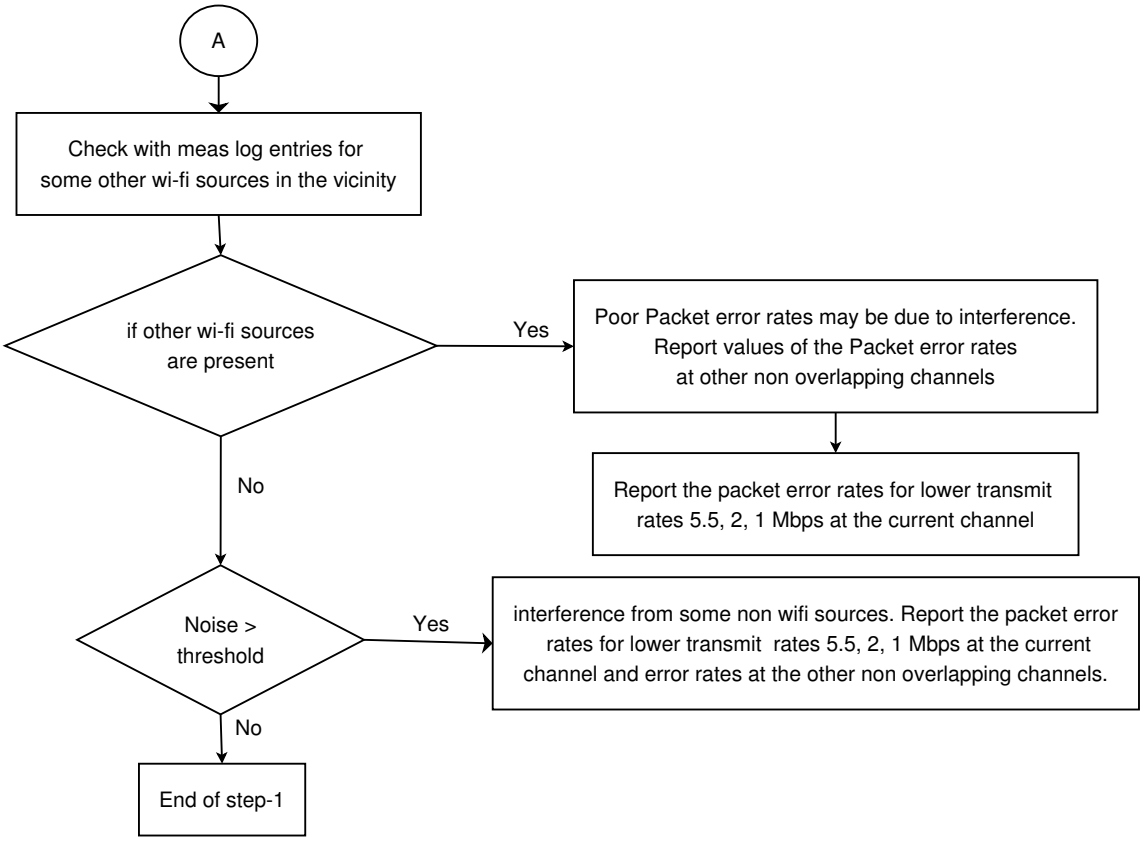
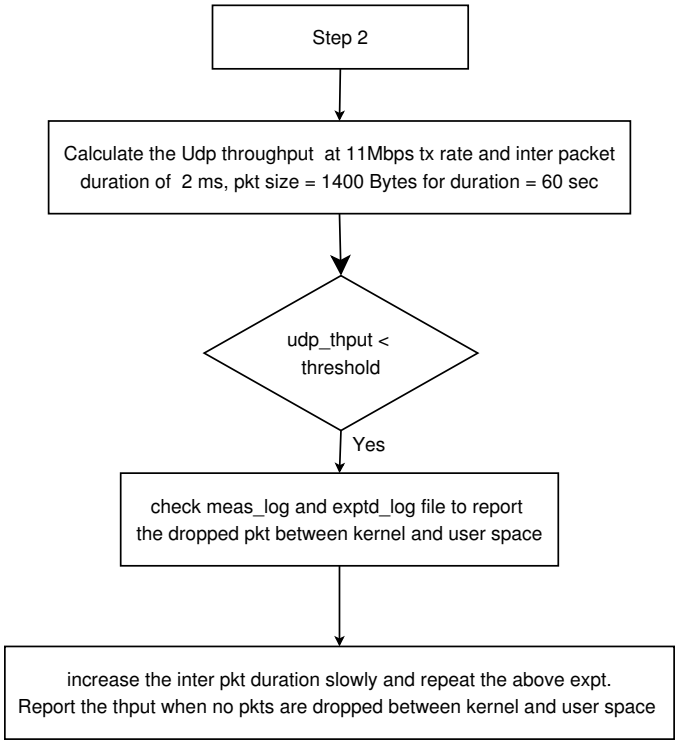
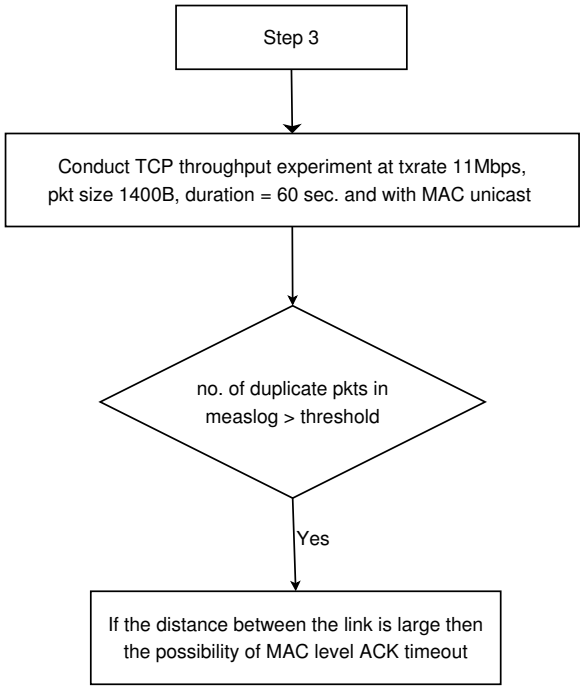


Figure 4.2: Debugging Algorithm : Step 1(contd..)



(a) Step 2



(b) Step 3

Figure 4.3: Debugging Algorithm : Steps 2 & 3

Chapter 5

Evaluation

The objective of our work is to detect & diagnose different problems that have been observed in earlier works [5] on long distance Wi-Fi networks. Hence in the evaluation, our objective would be to introduce those faults or get the effect of those faults in the network, run the algorithm and observe that whether we are able to detect and diagnoses those problems responsible for the degradation of performance.

In the following sections, we first explain how we introduce those problems in the network. Then well explain the experiment setup and finally we explain the evaluation results.

5.1 Introducing faults

- **Power Reset**

  In this problem the power gets reset to default factory value automatically. To introduce this fault, we start the daemon with the settings of the node then we log in to the soekris board through another session and change the hardware register corresponding to the transmit power value. We do this using the command get_config. Now the transmit power value reflected in the parameter file is different from the actual transmit power value.

- **Insufficient transmit power**

  To introduce this fault, we wrapped up the soekris boards manually inside a thick obstruction, we need thick clothes for this. This attenuates the signal. We set the transmit power at transmitter to a value such that RSSI at the receiver is lower than the threshold and packet error rate is high.

- **Interference**

  Since we are doing experiments inside our lab, signals from a lot of Wi-Fi equipments are already present in the environment. We move the current working channel of soekris boards to same as that of other Wi-Fi sources in the vicinity.

- **Packet dropping at receiver**

  It has been observed that the hardware of soekris board is not able to handle packets when operating at 11Mbps and packet size is 100 resulting in packets being dropped between the hardware and the driver. Hence to get this fault we change the operating rate to 11Mbps and perform experiment with packet size 100 bytes.

- **MAC level ACK timeout**

  Achieving this effect is very difficult. In the long distance links also MAC level ACK timeout has been observed only on the longest link (37km). Hence we do not evaluate this particular case.

- **Link Misalignment**

  As observed in [5], RSSI on the links are stable over time. Hence we maintain the RSSI history on all the nodes. If RSSI achieved in experiment is lower than RSSI history than we notify the administrator about the possibility of link misalignment. We perform evaluation of this problem detection along with insufficient transmit power because RSSI is low in the insufficient transmit power case.

- **Hardware Anomaly**

  We do not need to do anything to achieve this effect because this fault is already there in the senao hardware. Hence we just record the RSSI values of packets received with CRC error.

5.2 Experiment Setup

Due to unavailability of long distance links, we performed evaluation of problem detection and diagnosis algorithm in lab. Following is the experiment setup for evaluation:

1. We use two net4521 soekris boards with 133MHz processor, 64MB memory and voyage linux (kernel 2.6.15) with DLink DWL650 PCMCIA cards to create a wireless link. Both the soekris boards act as client.

2. We used a computer with 2.4GHz processor, 512MB memory and fedora core 4 linux (kernel 2.6.11) to act as a server. We connected both the soekris boards and computer to an ethernet switch to monitor the activities of experiment daemons on soekris boards from computer itself through the ethernet.

3. The debugging algorithm takes 2 arguments as ip addresses of the nodes of a link, hence the wireless interfaces of soekris boards should be directly reachable from server. Since server and wired interfaces of soekrises are in the different subnet from wireless interfaces of soekrises, the wireless interface of soekrises are not reachable directly from server. We have used following configurations to make it possible for server to directly connect with wireless interface of soekris boards:

   - ip addresses of wired network:
     - Server : 192.168.200.1
     - Soekris 1: 192.168.200.2
     - Soekris 2: 192.168.200.3
   - ip addresses of wireless network:
     - Soekris 1: 192.168.1.1
     - Soekris 2: 192.168.1.2
   - We configured the default gateway for server as 192.168.200.2 and turned on ip forwarding in Soekris 1.

   Figure 5.1 explains the experiment setup appropriately.

4. Among the different parameters based on both categories wireless interface parameters and experiment parameters, we fixed some parameters and varied other parameters relevant for particular experiment. We explain these parameters along with the results.

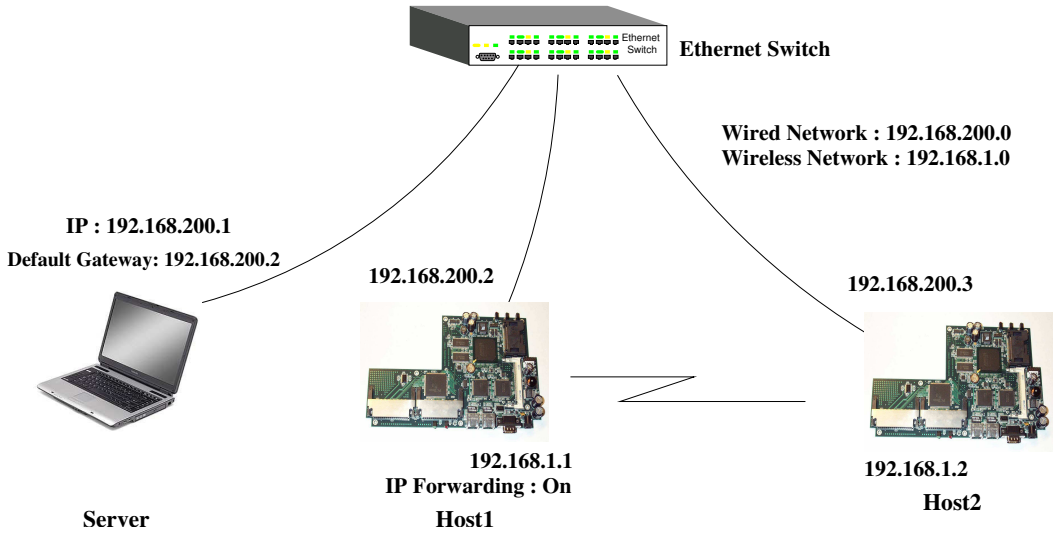


Figure 5.1: Experiment Setup

5.3 Experiment Results

5.3.1 Power Reset

We conducted some experiments for power reset and we observed that algorithm was successfully able to detect power reset at the host. Default transmit power of link was set to maximum as 20 dBm and current transit power was varied to different values and in all cases power reset detected.

5.3.2 Insufficient Transmit Power and Link Misalignment

We conducted another set of experiments to observe whether we are able to detect insufficient transmit power case. We checked for link misalignment case in these same set of experiments because due to insufficient transmit power RSSI is lower than RSSI history, we set the value of RSSI history to -75dbm. For each set of parameters, we repeated the experiments three times. Following are the set of parameters used for the experiments:

- Fixed Parameters:

    1. mode : master-managed
    2. transmit power = -4dbm
    3. inter packet interval = 20ms

Parameters			Results							
Channel	Tx rate (Mbps)	Run	Default tx power				Insufficient txpower	Max txpower		Link Misalignment
			RSSI (dBm)	Noise (dBm)	SNR	PER (%)		RSSI (dBm)	PER (%)	
1	11	1	-86	-96	10	7.78	Yes	-59	1.02	Yes
		2	-87	-96	9	12.45	Yes	-60	0.87	Yes
		3	-87	-95	8	17.23	Yes	-60	1.56	Yes
	5.5	1	-87	-94	7	6.65	Yes	-60	0.27	Yes
		2	-87	-95	8	20.85	Yes	-60	0.86	Yes
		3	-87	-94	7	15.11	Yes	-60	1.12	Yes
	2	1	-87	-95	8	32.67	Yes	-60	0.65	Yes
		2	-87	-95	8	9.20	Yes	-59	0.45	Yes
		3	-88	-95	7	11.49	Yes	-60	0.34	Yes
	1	1	-88	-96	8	29.50	Yes	-59	0.52	Yes
		2	-88	-96	8	37.54	Yes	-59	1.55	Yes
		3	-88	-96	8	40.68	Yes	-59	0.98	Yes
11	11	1	87	-96	9	54.32	Yes	-60	0.71	Yes
		2	-88	-98	8	46.78	Yes	-60	1.46	Yes
		3	-87	-96	9	39.12	Yes	-60	1.06	Yes
	5.5	1	-90	-95	5	28.43	Yes	-60	0.40	Yes
		2	-87	-95	8	56.55	Yes	-60	1.22	Yes
		3	-90	-95	5	14.42	Yes	-60	1.12	Yes
	2	1	-90	-94	4	20.34	Yes	-61	0.52	Yes
		2	-90	-95	5	12.22	Yes	-61	0.90	Yes
		3	-90	-95	5	9.54	Yes	-61	1.18	Yes
	1	1	-90	-95	5	23.50	Yes	-61	1.76	Yes
		2	-90	-95	5	50.11	Yes	-61	0.35	Yes
		3	-90	-96	6	27.90	Yes	-61	0.84	Yes

Figure 5.2: Experiment Results for Insufficient Transmit Power and Link Misalignment

4. packet size = 1400 bytes
5. no. of packets = 1000

- Varying Parameters:

1. channel = 1,11
2. transmit rate = 1, 2, 5.5, 11 Mbps

The results of the experiments for insufficient transmit power and link misalignment cases are shown in Figure 5.2. The threshold values chosen for SNR and packet error rates are same as stated in Section 4.3. In the results we show whether insufficient transmit power and link misalignment are detected or not. We also show RSSI and PER for packet error rate experiments with current transmit power and maximum transmit power. From the results, we observe that both insufficient transmit power and link misalignment are

detected successfully and packet error rate reduces substantially after setting transmit power to maximum value.

5.3.3 Interference Detection

We conducted experiments for interference detection from Wi-Fi sources. In the lab we observed that most of the Wi-Fi sources were working in channel 6 - 11. We performed these experiments with values of some parameters fixed and values of some parameters varying and repeated experiment for each set of parameters three times. The following are the set of parameters used for the experiments:

- Fixed Parameters:
    1. mode : master-managed
    2. transmit power = 20dbm
    3. inter packet interval = 20 ms
    4. packet size = 1400 bytes
    5. no. of packets = 1000
- Varying Parameters:
    1. channel = 1,11
    2. transmit rate = 1, 2, 5.5, 11 Mbps

The results of the experiments for interference detection are shown in Figure 5.3. In the results we show whether interference was observed or not and if it was observed then we also show MAC addresses and avg. RSSI of interferers. We have used notations A,B,C... to show MAC addresses. Please refer Figure 5.4 for these notations. In the results we observed interference in channel 11, while no interference was observed in channel 1.

5.3.4 Packet dropping at receiver

We conducted another set of experiments to check whether packet dropping is detected or not. Again we performed these experiments with values of some parameters fixed and values of some parameters varying and repeated experiment for each set of parameters three times. Following are the set of parameters used for the experiments:

Parameters			Results			
Channel	Tx rate (Mbps)	Run	RSSI (dbm)	PER (%)	Interference	Interferers (RSSI in dBm)
11	11	1	-56	4.6	Yes	A(-82), B(-85), C(-76), D(-86)
		2	-57	3.5	Yes	A(-85), C(-85), G(-81), F(-77), H(-81),I(-88)
		3	-54	2.7	Yes	A(-80), J(-86), C(-77), G(-83)
	5.5	1	-55	3.5	Yes	A(-85), K(-86), C(-76), G(-83), L(-88), M(-88), B(-91)
		2	-55	0.7	Yes	A(-92),B(-90), C(-75), G(-83), L(-86)
		3	-55	0.7	Yes	A(-85), B(-91), C(-75), G(-83)
	2	1	-56	3.4	Yes	A(-85), B(-90), C(-75), G(-83)
		2	-55	3.3	Yes	A(-88), C(-76), G(-83), N(-68)
		3	-55	0.8	Yes	A(-86), O(-83), C(-75), G(-83)
	1	1	-55	0.8	Yes	P(-89), A(-83), B(-91), C(-75), G(-83)
		2	-55	0.3	Yes	A(-83), B(-86), C(-75), Q(-82), G(-83), R(-86)
		3	-55	0.8	Yes	A(-84), C(-76), S(-82), G(-82), T(-82)
1	11	1	-54	3.5	No	0
		2	-54	3.6	No	0
		3	-54	0.6	No	0
	5.5	1	-54	0.6	No	0
		2	-54	0.6	No	0
		3	-54	3.5	No	0
	2	1	-54	3.5	No	0
		2	-54	3.6	No	0
		3	-54	0.6	No	0
	1	1	-54	3.4	No	0
		2	-54	0.6	No	0
		3	-54	0.5	No	0

Figure 5.3: Experiment Results for Interference Detection

- Fixed Parameters:
  1. mode : master-managed
  2. transmit power = 20 dbm
  3. transmit rate = 11 Mbps
  4. packet size = 100 bytes
  5. no. of packets = 10000
- Varying Parameters:
  1. channel = 1, 11
  2. heart beat interval = 0, 2, 4 ms

We found a bug in the code, while doing experiments for packet dropping. In many of the results in Figure 5.5, number of packets received at hardware are less than number of

Notation	MAC Address	Notation	MAC Address
A	00:0f:b5:96:c6:70	K	00:11:95:58:5a:7e
B	00:11:95:d8:e3:46	L	00:77:96:d8:e3:46
C	00:11:95:d8:e3:48	M	36:11:95:d8:e3:48
D	00:11:f3:db:e3:48	N	00:89:98:d8:e3:48
E	00:11:95:d8:b7:f3	O	00:11:0d:d4:45:b4
F	00:11:95:d8:e3:74	P	00:11:0d:d4:45:b4
G	00:11:95:d9:45:b4	Q	00:11:95:d9:45:78
H	98:dc:7a:c3:a5:6b	R	00:11:95:58:5e:e8
I	c0:63:b5:96:c6:70	S	00:11:95:d9:45:48
J	00:11:95:b8:d5:48	T	73:d1:7a:93:77:63

Figure 5.4: Notations used to show MAC addresses

Parameters			Results				
Channel	HB Interval	Run	RSSI	PER (%)	No. of pkts at udp	No. of pkts at meas log	Pkts dropped (yes/no)
1	0	1	-51	86.58	1342	1496	Yes
		2	-51	86.70	1330	1516	Yes
		3	-51	86.64	1336	1508	Yes
	2	1	-52	33.04	6696	6025	No
		2	-52	33.15	6685	6002	No
		3	-51	28.14	7186	6447	No
	4	1	-51	0.01	9999	9458	No
		2	-52	0.03	9997	9470	No
		3	-51	0.00	10000	9464	No
11	0	1	-52	86.62	1338	1497	Yes
		2	-53	86.62	1338	1488	Yes
		3	-52	86.45	1355	1468	Yes
	2	1	-53	27.78	7222	6501	No
		2	-53	33.01	6699	6017	No
		3	-53	33.07	6693	6012	No
	4	1	-53	1.33	9867	9337	No
		2	-53	0.00	10000	9476	No
		3	-52	0.00	10000	9474	No

Figure 5.5: Experiment Results for detecting packet dropping at receiver

packets received at upper layers. We tried to fix the bug but we are not yet successful in fixing the bug. In the results, we are able to detect packet dropping in case of inter packet interval of 0 ms, but unsuccessful for higher heartbeat intervals.

Chapter 6

Conclusion and Future Work

In this work, we designed and implemented a network monitoring tool for long distance Wi-Fi networks based on a centralized client-server approach. We also implemented a set of experiments namely *packet error rate*, *UDP throughput*, *TCP throughput*, *get config* and *set config* to observe performance and retrieve and set wireless parameters on the links of a long distance Wi-Fi network. We also designed and implemented a debugging algorithm to detect and diagnose problems known to be commonly occurring on long distance Wi-Fi networks and listed in [5], using our monitoring tool from a central location.

The problems that we focused on detecting and diagnosing from the central location are power reset, insufficient transmit power, link misalignment, interference detection, hardware quirk and MAC level ACK timeout. In our work, we intregrated the modifications in the hostap driver used in [5] for retrieving per packet kernel level information at the user level such as RSSI, noise, number of packets received etc. with our experiments. In a debugging algorithm, we conducted experiments from the central location and detected and diagnosed problems in case of poor performance.

We evaluated our work by establisihing wireless link using soekris boards in the lab. We introduced faults stated above in the wireless link and used debugging algorithm from server to conduct experiment and detect and diagnose problems. We successfully detected the problems of power reset, link misalignment, interference, hardware quirk and insufficient transmit power. Our results showed that packet error rate dropped sharply by increasing transmit power in case of insufficient transmit power.

There is a lot of scope to further extend this work. We need to test this tool thoroughly on long distance Wi-Fi networks and perform interference measurement once the interfer-

ence is detected. Another future work involves integrating our work with [4]. Our work also requires the implementation of a user friendly GUI for the network administrator at the central server.

Bibliography

[1] A. Adya, P. Bahl, R. Chandra, and L. Qiu. Architecture and techniques for diagnosing faults in ieee 802.11 infrastructure networks. In *MobiCom*, September 2004.

[2] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level measurements from an 802.11b mesh network. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, August 2004.

[3] Project Ashwini: Virtual Delivery Services. `http://www.byrrajufoundation.org/ashwini_home.htm`.

[4] A. Bhadauria. WiFiNetmon: Interference Measurement in Long Distance WiFi Mesh Networks. Master's thesis, Department of Computer Science & Engg., Indian Institute of Technology Kanpur, June 2007.

[5] K. Chebrolu, B. Raman, and S. Sen. Long-Distance 802.11b Links: Performance Measurements and Experience. In *12th Annual International Conference on Mobile Computing and Networking (MOBICOM)*, september 2006.

[6] S. M. Das, D. Koutsonikolas, Y. C. Hu, and D. Peroulis. Characterizing Multi-WayInterference Interference in Wireless Mesh Networks. In *WinTech*, September 2006.

[7] C. C. Ho, K. N. Ramachandran, K. C. Almeroth, and E. M. Belding-Royer. Scalable framework for wireless network monitoring. In *WMASH*, October 2004.

[8] N. Mishra, K. Chebrolu, B. Raman, and A. Pathak. Wake-on-WLAN. In *The 15th Annual Interntional World Wide Web Conference (WWW 2006)*, May 2006.

[9] N. Mishra, D. Golchha, A. Bhadauria, B. Raman, and K. Chebrolu. SWOW: Signature based Wake-on-WLAN. In *The First Annual Workshop on WIreless Systems: Advanced Research and Development (WISARD 2007), A COMSWARE 2007 Workshop*, January 2007.

[10] Project RuralNet (Digital Gangetic Plains: DGP) 802.11 based Low Cost Networking for Rural India. `http://www.cse.iitk.ac.in/users/braman/dgp.html`.

[11] L. Qiu, P. Bahl, A. Rao, and L. Zhou. Troubleshooting multihop wireless networks. In *SIGMETRICS*, June 2005.

[12] B. Raman. RuralNet Project Report, July 2006.

[13] A. Sheth, C. Doerr, D. Grunwald, R. Han, and D. Sicker. MOJO : A Distributed Physical Layer Anomaly Detection System for 802.11 WLANs. In *MobiSys*, June 2006.

[14] R. Shrivastava. Network Montoring Tool for 802.11 Wireless Long Distance Rural Networks. Master's thesis, Department of Computer Science & Engg., Indian Institute of Technology Kanpur, July 2006.

[15] Soekris Engineering. `http://www.soekris.com`.

[16] TIER group, University of California Berkeley, USA. `http://tier.cs.berkeley.edu/wiki/Wireless`.

[17] WRAP boards. `http://www.pcengines.ch/wrap.htm`.

[18] Extensible Markup Language (XML). `http://en.wikipedia.org/wiki/XML`.