Visualization for the Analysis of Congestion in IEEE 802.11b Wireless Networks

Raji Kannah
Undergraduate Student Researcher

Amit P. Jardosh
PhD Student Mentor

Elizabeth M. Belding
Faculty Mentor
Department of Computer Science,
University of California, Santa Barbara
rajikannah@gmail.com, {amitj, ebelding}@cs.ucsb.edu

Abstract

In network communication, wireless networks are getting more popular and are being deployed everywhere. The growing number of users and wide-spread adoption of this network increases congestion in the wireless networks. We believe it is essential to understand link-layer behavior in the congested or heavily utilized wireless networks. A good visualization tool increases understanding of the problem and allows us to approach in another dimension to find a solution. Our general approach is to create a wireless network analyzer application, based on the research work done by fellow researchers. The Graphical User Interface is developed using Sun Java NetBeans and GUI user can visualize sniffer and access points in action. Our goal is to face the vicinity sniffing challenges with better tool and also would aid in better analyzing the uncaptured frames. The objective of this GUI is to bring in visualization of the vicinity sniffer's location with multiple access points (AP).

Key Word: Visualization of wireless Network Analyzer

1. Introduction

Congested wireless network can be defined as the state in which the transmission channel is close to being completely utilized. As wireless networks become heavily utilized, the importance of studying behavior of MAC layer increased as well. Congested wireless network is characterized by high medium occupancy, large number of retransmission of data and high throughput [2] and data rate variations. After the transmission of a data frame, when the sender does not receive an acknowledgement at the specific time period generally IEEE 802.11b MAC layer retransmit the data frames. The congestion level increases because of these larger numbers of retransmission of data frames and also increases possibility of losing data frames during retransmission of data frames. (1)What are the effects of congestion? [2].some of the most common effects of congestion, (1) Average MAC frame sizes are received more compare to larger frame sizes. (2) Smaller data frames are sent at lower data rates level successively received by a device in the network. But larger frames at high data rates are dropped.

The method adopted to collect information from the wireless network is referred as wireless network monitoring [4] or vicinity sniffing. Vicinity sniffing is the best currently available method to collect link layer information from an operational wireless network. The vicinity sniffing challenges are sniffer location and uncaptured frames.

In this project work, we take our first steps toward the placement of sniffers in the congested network. Sniffers are unable to record all frames in the large scale usage. To increase the data collections in the congested wireless network finding sniffer location is helpful. We hope our potential solution to this vicinity

challenges to present the design and implementation of the GUI work. It would help to find better place for the sniffer location in the large scale networks and analyze its access points.

This paper is organized as follows. Section 2 explains tethereal and its functions. Section 3 provides information about the related work was done by the previous students. Section 4 describes the graphical user interface in Java for the Wireless Network Analyzer. Section 5 brings the conclusions from our work.

2. Background

Tethereal is a network protocol analyzer which allows the user to capture the data from a live network or read packets from already saved file. We can decode the packets and print it in a standard output or write the packets to a file. This project was focused on the research paper [2].Related work of this project work was done by a previous student. The code is used to read the tcpdump file from the live capture. The end result of running the code produces a "stats.out" file that lists all BSSes (Basic Service Set) that actually had transmitted a data packet. The lists of BSSes sorted by the number of data ack and by the number of packets transmitted on the BSS. A wireless sniffer is supposed to capture all the packets from the access point. But it couldn't trace all the packets due to loss of signals (similar to noisy rise in the congested network in the radio waves or telephone signal). The research work [5] shows how we can trace complete wireless packets by FSM (finite state machine)

3. Visualization

3.1 NetBeans:

NetBeans has quite a collection of visual widgets that enable users to develop a rich GUI application. NetBeans is an excellent Integrated Development Environment (IDE) for editing, compiling and running Java programs and also works on cross platform such as Windows, Linux and Solaris. NetBeans are widely used by more users. It provides the extension of swing components and titles.

NetBeans is a good choice to try and modify different design for the project work. It has lots of applications especially window tools Palette manager and Inspector. These are used to add Swing Components for the frame design. Borders can be visually seen by the user and given choices to select from various collections. Palette manager is used to make customized panel and frames for the GUI application. Inspector is used to provide graphical representation of the application. Initially, the only problem was faced using NetBeans was fixing the place for button, labels, panels and Text Area. Because all the swing components are designed to drag and place it on the frame or panel wherever the designer wants. This handling design problem will be solved by playing with the components.

3.2 Design and Development approach

We used Fedora version 5 to run NetBeans 5.5.1 version for the project development. We designed our GUI design according to our requirements. It was more like very iterative work. The two source packages of Java main application named as MyComponents and displayap. AP class, Sniffer class and APDisplayPanel classes are stored under MyComponents source package. Mainframe and StatParser classes are stored under the displayap source package. The idea of creating separate source packages is to manage many implementation of Java platform easily.

StatParser class has StringTokenzier method which parses one string as a token from the output file of the wireless network. Each access points are identified by its BSSID and store the details of BSSID separately. User can analyze the details of data packets, ACK, RTS, CTS and other management frames. Mainframe class controls the layout of the GUI. Generally it contains the buttons, labels, panels, text area. Each button has mouse clicked event handler, which execute each command.

AP class uses array to store access point's details. AP is identified by its BSSID. To get a nice text format for displaying BSSID rectangles are used.

Sniffer class is for each sniffer draw and generates the random color for the sniffer display. The main objective of this class is anywhere if user clicks on the display panel; the sniffer location has to appear. The idea behind this requirement is to get better place for the sniffer. We used clickinside method to get sniffers location on the panel. This method draws a rectangle for the sniffer and also it has event handler method handles the details of each sniffer. If user double clicks on the sniffer rectangle it will give the details of all of its access points and AP's signal strength. AP Display Panel class handles all the mouseListener events such as mouse dragged, mouse released and mouse clicked. Having these entire mouse listeners helps user to move and place sniffer and AP anywhere on the panel. Basically this is where sniffer, access points are drawn and if the rectangles overlapped at each other it will be replaced by adjusting its x, y coordination. Also placing accesspoints around the sniffer is determined by x, y in all direction.

At first, when user clicks on execute live capture button on the left side of the GUI. It will open the separate execute dialog box for the user to enter and modify the live capture and stop command. Second the filetextfield which is under execute live capture button used to read the output file from the running of wifi_parser.cpp file. Then user clicks on the read button, it gives the total numbers of the access points. To understand better which access point (AP) is located near the sniffer is seen by its BSSID number in the rectangle. Customized AP display panel is used to place sniffer wherever user clicks on the panel. For better understanding, access point is seen by its BSSID number in the rectangle. Display button displays all the access points around the selected location of the sniffers. The GUI has two separate text areas sniffer text area and AP details text area. When the user double clicks on the sniffer, sniffer text area displays all the access points of the selected sniffer and also each access point's signal strength. The AP details text area displays each access point details separately. These details would help the user to study the data patterns, how many times the data packets are being retransmitted and missed ack, matched ack,RTS(request to send),CTS.(clear to send).The relationship between sniffer and AP was shown by different color lines and also access point's signal strength.

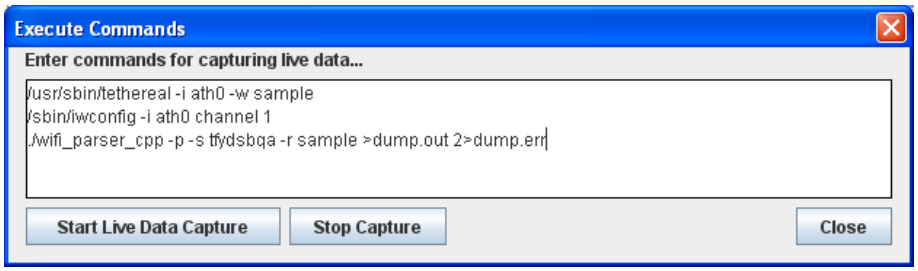

Figure (2) execute dialog

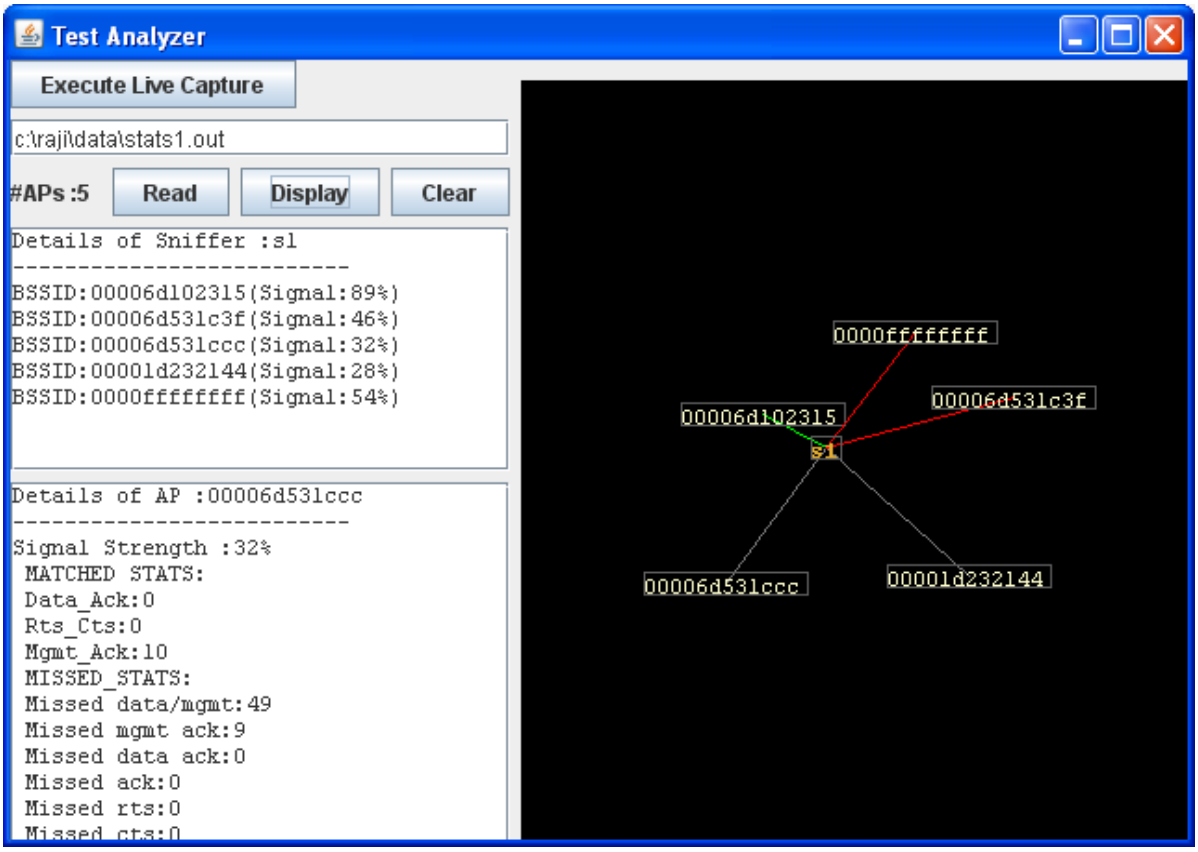

Figure (1) A sniffer and its access points

Fig (1), visualize the sniffer location and its access points from the saved output file. Here a sniffer s1 appeared on the panel and displayed all its access points. The relationships between sniffer and access points were shown by different color lines. Theoretically, access point signal strength somewhere the range between 0 to 100. The signal strength percentage in five different ranges.

- 0- 20% - light gray color
- 20-40% - Gray color
- 40-60% - Red color
- 60-80% - Yellow color
- 80-100%- Green color

The left side of this GUI has details of both sniffer and access point's details. We supposed to place AP from sniffer by AP's actual signal strength. But for some reason we could not read the actual signal strength from beacon frames. Here according to randomly (5-95) generated signal strength, AP was placed by calculating its x, y co-ordinates.

A separate dialog figure (2) would be useful for the user to enter and modify the live capture and stop command. This enables them to dynamically change the commands as it is required in the actual field. In the design, this flexibility was thought late in the development process but seems to be of great use.

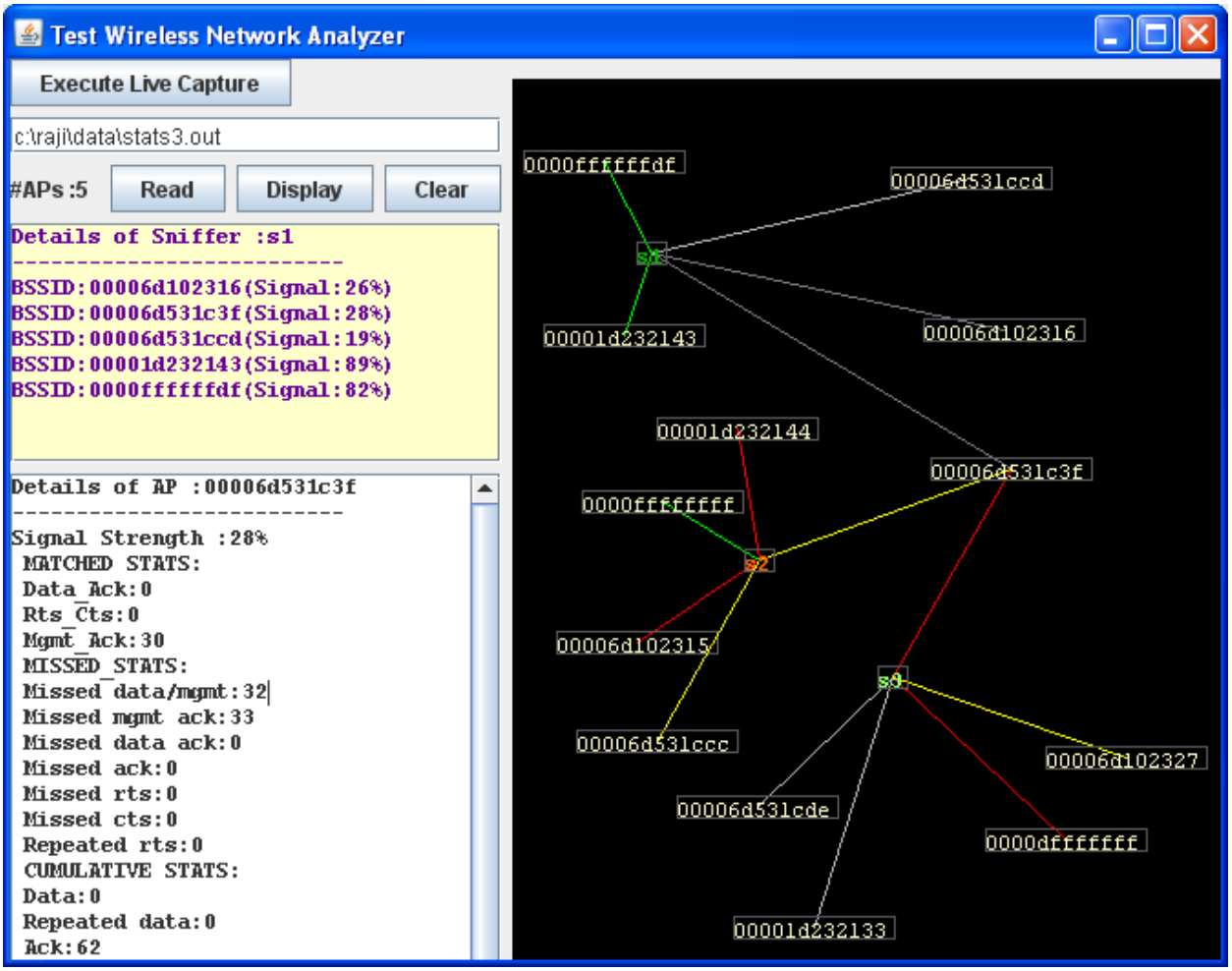

Figure (3) An AP accessed by multiple sniffers.

The most important feature of this visualization is if an AP was accessed by multiple sniffers that AP has to be connected with all sniffers in order to study data flow patterns. And also its details were different according to each sniffer's location and we have to display them in the separate text area. By getting this, user can determine the required position for the sniffer. Here, in this example, s1, s2, s3 sniffers are accessing the same AP BSSID: 00006d531c3f. Figure (3) shows s1is getting AP at only 28 %(gray) whereas s2 is getting stronger signal (yellow) of AP compare to s3 and details of all the sniffers and ap details will be shown in the assigned text areas. The location of one or more sniffers can be captured from the wireless network.

Future we want to focus more on getting actual AP signal strength of each sniffer from the live capture and adjust its position according to its signal strength of each sniffer to analyze congestion in wireless networks.

5. CONCLUSIONS

From this project, we have created a new tool to analyze congested wireless networks. We tested our GUI with live capture data and with saved output file. We have addressed the problems of getting access point signal strength and placing them around the sniffer. In the longer term, we are planning to use our GUI to better place the access point (AP) around the sniffer using actual AP's signal strength and also want to improve analyzing uncaptured frames.

6. ACKNOWLEDGEMENTS

This research was supported by the CRA-W Distributed Mentor Project award for summer 2007 year at University of California, Santa Barbara, CA-93106.

For more information about my research work, visit http://moment.cs.ucsb.edu/rajikannah

7. REFERENCES

[1] Amit P. Jardosh, Krishna N. Ramachandran, Kevin C. Almeroth and Elizabeth M. Belding-Royer. Understanding Link-Layer Behavior in Highly Congested IEEE 802-11b Wireless Networks. In proceedings of SIGCOMM'05 workshops, August 22-26, 2005, Philadelphia, PA, USA.

[2] Amit P.Jardosh, Krishna N. Ramachandran, Kevin C. Almeroth and Elizabeth M.Belding – Royer Understanding Congestion in IEEE 802.11b Wireless Networks. In the Proceedings of the ACM SIGCOMM Workshop on Experimental Approaches to Wireless Network Design and Analysis, Philadelphia, PA, August 2005

[3] J.Yeo, M. Youssef and A.Agarwala. A Framework for wireless LAN Monitoring and its Applications. In proceedings of the ACMWorkshop on Wireless Security, pages 779, Philadelphia, PA, October 2004.

[4] Ratul Mahajan, Maya Rodrig, David Wetherall, John Zahorjan. Analyzing the MAC-level Behavior of Wireless Networks in the Wild. In proceedings of SIGCOMM'06, September 11-15, 2006, Pisa, Italy.