Stony Brook University

Scalable Wireless LAN Traffic Monitoring and Analysis

A THESIS PRESENTED

BY

PRIYA THANGARAJ

TO

THE GRADUATE SCHOOL

IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

IN

COMPUTER SCIENCE

STONY BROOK UNIVERSITY

DECEMBER 2009

Stony Brook University

The Graduate School

Priya Thangaraj

We, the thesis committee for the above candidate for the
Master of Science degree,
Here by recommend acceptance of this thesis.

Professor Tzi-cker Chiueh, Thesis Advisor
Computer Science Department

Professor Samir Das, Thesis Committee
Computer Science Department

Professor Jie Gao, Thesis Committee
Computer Science Department

This thesis is accepted by the Graduate School

Lawrence Martin

Dean of the Graduate School

Abstract of the Thesis

Scalable Wireless LAN Traffic Monitoring and Analysis

By

Priya Thangaraj

Master of Science

In

Computer Science

Stony Brook University

2009

Scalable Wireless LAN Traffic Monitoring System provides a comprehensive report on traffic load, radio channel usage and security alerts in real time for all WLAN links. The Monitoring feature does not disrupt the operation or affect the performance of the monitored WLAN's. The scalable wireless LAN traffic monitoring does weighted channel hopping to determine the channels used in the network. Weighted channel hopping gives a brief summary of the channels in IEEE 802.11 framework. During channel hopping the packets are sniffed in promiscuous mode using a third party tool called as KISMET. The sniffer provides the feasibility of determining the load from each channel. Every packet received through a sniffer is carefully dissected and the statistics pertaining to each channel is sent to a central server through SNMP protocol. SNMP (Simple Network Management Protocol) is used as a transport layer to transfer WLAN traffic related statistics from the monitoring client to a main server. The key feature is an accurate radio channel busy time estimation algorithm that correctly takes into account both the back-off delay and corrupted packets in WLAN traffic load computation without requiring any modification to monitor WLAN devices.

Another main aim for scalable wireless LAN traffic monitoring is to develop a unique application called NMIF (Network Management Interface Framework), a framework to be used for Wired and Wireless Monitoring to provide a very clear view of the wireless usage in the network.

To

Everyone

Contents

List of Tables

List of Figures

Acknowledgements

First of all, I sincerely wish to thank Dr. Tzi-cker Chiueh for his guidance and support all through the project. I would like to thank my thesis committee Dr. Samir Das and Dr. Jie Gao for going through my thesis thoroughly and providing valuable suggestions. I would like to thank all my ECSL-mates for their technical expertise and also for providing a great working environment. And finally, I would like to thank my family, who supported me unwaveringly throughout my life.

Chapter 1

Introduction

The sweeping popularity of IEEE 802.11-based wireless LAN (WLAN) technology in both consumer and small/medium business markets has turned WLAN into an increasingly important building block of the networking infrastructure of commercial enterprises. However, on the way towards universal deployment, WLAN still has several technical barriers to cross. Chief among them are management and security issues, or how to effectively administer a large-scale wireless network and how to stop all malicious attacks at the physical, medium access control (MAC), Internet protocol (IP) layer and above. The first and most important issue in managing any networks, including wireless LANs, is to provide real-time visibility for the composition of traffic loads and detailed network resource usage statistics. Because of mobility and changing radio conditions, both input workload and raw network link capacity of a wireless LAN may fluctuate rapidly over time, making real-time traffic/usage reporting particularly valuable to network administrators. The other major element of a network management system, network control, is also essential to production-mode WLAN operation.

Although several emerging standards such as IEEE 802.1x, IEEE 802.11i and WPA, are poised to address most of the security problems found in first-generation IEEE 802.11 WLANs, these solutions require complicated set-up and wide-ranging modifications to the existing IT infrastructure. Moreover, they are not necessarily backward-compatible with legacy IEEE 802.11 devices.

Aiming to solve the management and security problems associated with the current WLAN technology, we set out to develop a scalable WLAN traffic monitoring, which can provide comprehensive reports on traffic load and wireless link usage in real time for all WLAN's in an enterprise, and perform MAC-layer intrusion detection by identifying anomalous network events such as MAC address spoofing, unusual probing and AP switching frequencies.

The scalable wireless traffic monitoring involves weighted channel hopping to have a gist of the channels being loaded, the channels being chosen for further investigation are the one's which has relevant details of interest, especially has Access Points and association of stations. The goal of providing a real time view of the wireless network is achieved by the choice made in weighted channel hopping. This channel hopping has a sniffer to capture the packets from the air, through the PCAP module and then invokes the API's of Kismet, a third party utility to dissect the packet. The channel hopping also invokes some of Kismet's API's to figure out the security alerts in the air. Now every packet provides some valid details pertaining to Access Points and the associated stations

and also security issues, the valid information are then passed through the IPC mechanism to SNMP Agent. The SNMP Agent further transmits the MIB details to SNMP Manager.

Simple Network Management Protocol (SNMP) is used as the transport layer for the exchange of statistics about the network. SNMP is a well defined application layer protocol that enables the exchange of management information between network devices and hence allows the network administrators to manage the network effectively. A SNMP managed network mainly consists of three key components:

1. Managed device, a network node that contains an SNMP agent and that resides on a managed network. It collects and stores management information and makes the information available to network management systems (NMS) using SNMP.
2. SNMP Agent, a network management software module that resides in a managed device. It has local knowledge of management information and translates that information into a form compatible with SNMP.
3. Network Management System (NMS), it executes application that monitors and controls managed devices. One or more NMS must exist on any managed network.

A SNMP Management Information Base (MIB) is a collection of information organized hierarchically. They are comprised of managed objects and are identified by Object Identifiers (OID).

A managed object (sometimes called a MIB object, an object, or a MIB) is one of any number of specific characteristics of a managed device. Managed objects are comprised of one or more object instances, which are essentially variables. Two types of managed objects exist: scalar and tabular. Scalar objects define a single object instance. Tabular objects define multiple related object instances that are grouped in MIB tables.

OID uniquely identifies a managed object in the MIB hierarchy. The MIB hierarchy can be depicted as a tree with a nameless root, the levels of which are assigned by different organizations.

The main operation of SNMP consists of setting information in the network devices (using SNMP SET command) and querying information from them (using SNMP GET command). So we believe that SNMP can also be leveraged to query the real time information i.e. query results from sensor nodes.

Another advantage of using SNMP in our system is it that can be interfaced with already existing SNMP agents to collect data from them and also be interfaced with network management systems for plotting collected data.

The term load has been traditionally defined in different ways. Very often, load is interpreted as the number of stations associated with the access point. Another interpretation of load is the number of frames that are successfully handled by APs per

unit time. Load can also be interpreted as the percentage of time that the access point senses the channel to be busy. However, these definitions of load are incorrect, in the context of the applications that depend on the load estimate. This work aims to develop a meaningful metric to measure load, and to design a simple mechanism to estimate it. The key feature of this thesis is an accurate radio channel traffic load estimation algorithm that correctly takes into account both back-off delay and corrupted packets in WLAN traffic load computation without requiring any modification to monitored WLAN devices or the frames. The management data from weighted channel hopping helps in finding the busy time and idle time period for every packet and helps in estimating the radio channel busy time.

An important other module of the thesis is NMIF, Network Management Interface Framework, the main idea is to have a GUI for the real time view of any network, be it be wireless or wired. NMIF provides a graphic interface to display the deployment of Access Points, the stations associated with them as well as any alerts or alarms that are in the network. NMIF provides a tree view of all the Channels, the Access Points and the Stations. It also has a query support to monitor certain specific alert or alarm of interest. NMIF uses JUNG [15] for the graphical view of Access Points and Stations, SwiXML [16], to define the framework for NMIF, SQLite [17] database server which has the statistical details of the wireless network and SQLiteJDBC [18] for the connectivity to the SQLite server.

The rest of the thesis is organized as follows; chapter 2 surveys the related work in WLAN field. Chapter 3 explains the design and implementation of the system. Chapter 4 discusses about Traffic Load Estimation Algorithm. Chapter 5 has details on various WLAN securities. In chapter 6, performance evaluation of the Traffic load estimation algorithm as well the second aim of giving a clear view of the exchanges that happens in the air. Chapter 7 explains the future work that can be possibly carried on with scalable wireless LAN traffic monitoring and conclusion in chapter 8.

Chapter 2

Related Work

A "*Scalable WLAN Traffic monitoring*" System is critical to successful operation and management of wireless networks. Existing network management tools [1], [2], [3] collected network event data in real time using network management standards or accessing log information kept in wireless devices. For example, VISUM [1] relies on a set of agents to retrieve data from the access points using SNMP and store them on the data repositories for further processing. One of the limitations of these systems is that not all access points support SNMP, thus limiting their deployment in the real world. Another limitation is that SNMP MIB's or the logs generally do not provide detailed link-layer or physical-layer information, such as signal strength, physical transmission rate of each frame, and the number of retransmissions. Thus it is difficult for these systems to calculate the parameters like monitored WLANs signal quality, estimate the channel usage, or detect MAC address spoofing attack.

Several WLAN workload studies [7, 8, 9] performed offline analysis on the trace of a building-wide or a campus-wide wireless network, including syslogs, SNMP, and tcpdump. The monitoring systems used to make empirical observations have increasingly expanded both in complexity and scope over time. Early systems used existing infrastructure, such as the wired distribution network and the APs, to record wireless traffic and network characteristics [26, 27]. Later systems deployed small numbers of dedicated monitoring nodes, sometimes concentrated near the APs, other times distributed throughout the network, thereby pushing the frontier of observation into the link-layer domain [28, 29 and 30]. Recent efforts have substantially scaled monitoring platforms to observe large, densely deployed networks in their entirety [31, 32], providing the ability to observe every link-layer network transmission across location, frequency, and time [32]. The major goal of these efforts is to study the user behavior, traffic characteristics, usage patterns and network performance in a long term, and then to apply the analysis results into the research of new wireless application development, future WLAN deployment and management. In contrast, *WLAN Traffic Monitoring* is designed to report data of interest like traffic loads, network resource usage, anomalies in networks in real time for enterprise WLANs.

Load estimation analysis is done by piggybacking the CW (contention window) value in every frame to estimate the load in an AP [33]. A certain implementation [34], expects a frame to have additional details about the number of the already associated stations to the AP and the mean RSSI value. Some older papers [34, 35 and 36] equate load to the number of stations in the Basic Service Set (BSS). These papers assume that each station has the same traffic pattern, and hence the same bandwidth requirement. Another weakness of this definition of load is caused by the multi-rate capabilities of the 802.11 a/b/g/n [37]. Since stations may talk to the access point at different data rates, the time taken for a frame transmission depends on the data rate. Consequently, a load of say, x

frames per second might indicate congestion if the data rate is 1Mbps, but the same load might imply low channel utilization if the data rate is 54Mbps. From the above points, either the implementation in finding load is either complicated by changing the frame which might reduce the performance as well missing some of the important factors as back-off time, thus “Scalable WLAN Traffic Monitoring System’s”, traffic load estimation has heuristics about the back-off time and therefore comes with a good analysis of channel usage time.

Kismet [6] is an 802.11 wireless LAN sniffer, and provides several intrusion detection functionalities. It is able to work with any wireless NICs that support promiscuous mode. It is mainly designed for monitoring a signal region, and cannot scale to an enterprise network directly. In addition, none of them reports the channel utilization and performs spoof detection function. Adya et al. [24] presents a flexible client-based framework for detection and diagnosis of wireless faults. Instead of spreading the sensors all around the building, it takes advantage of the stations and AP's to discover disconnected clients, detect Rogue AP's and estimate wireless network delay. However it requires modification of existing AP's and clients. Also, the set of functionalities it provides are more limited when compared with the system in discussion. OpenNMS [38] is the network management platform provides the GUI for wired and wireless network management, but it fails to acknowledge some of the critical alerts such as MAC address spoofing, and fails to notify any traffic load at AP or Channel level.

IEEE 802.11k [25] is a proposed standard for radio resource management with the goal of optimizing the network performance. To have a better traffic distribution within a network, it requires the clients and access points to share the channel usage information and redirects some clients to underutilized access points. However, IEEE 802.11k itself does not define how radio resource usage should be measured. Therefore the traffic load estimation algorithm described could be a useful building block for the IEEE 802.11k standard.

Chapter 3

System Architecture

3.1 Overview

"Scalable Wireless Traffic Monitoring" system is basically developed on a normal PC with wireless NIC card, which can support the Promiscuous Mode for passive sniffing of packets. The development is on a Linux machine which is highly scalable and can be ported to any other Linux varieties.

3.2 WLAN Traffic Monitoring

The first goal of "*Scalable WLAN Traffic Monitoring*" is to provide real-time visibility of what is happening in the radio space. In Wireless Network, the number of devices deployed and being used are always uncertain because of the new advancement in technologies wherein, a user's laptop can be setup as an Access Point to establish connections. Therefore, a clear visibility of all the devices that were deployed and others who abuse the network was always a requirement for wireless network. In counterpart, there was always a clear view of the wired network, because of the use of unlicensed spectrum, the network administrators are at risk who cannot distinguish the problem encountered. From a single GUI-based management interface, a network administrator can monitor the radio resource usage of all the WLAN links across an enterprise, including

- The radio channel usage in each WLAN,
- Each AP's configuration including channel assigned, public name, measured transmission power, employed security feature, a list of currently associated stations
- Each Station's traffic load, raw transmission rate, measured transmission power, and position trajectory in terms of the sequence of Access Points with which it is associated over time,
- Clients and Access Points who have invalid configuration, for instance, clients sending excessive probe messages without any successful association
- Traffic Load estimation at each Access Point would help in figuring out the devices that abuse the network, the load can be shared across other access points and finally, one can distinguish between a wireless and wired mishaps and
- Bad fish that could bring down the throughput of a WLAN because it transmits at a lower physical transmission rate than others, and thus consumes substantially more channel time than others.

From these reports, a network administrator can manually reconfigure the access points to improve the WLAN coverage, detect/troubleshoot misconfigurations of specific AP's or clients, and identify congested spots, all in real time.

3.3 Architecture

Architecturally, the system consists of Sensor Nodes and a Manager Node. The Sensor Nodes deployed in various regions of a building is dedicated to collect interesting facts about the wireless network. The sensor node by means of passive sniffing without disturbing the performance collects all hidden facts in the network. There is enough parsing done within every sensor node to categorize the valuable data based on statistical data or security related information rather than passing every collected data to the Manager. The Manager node frequently updates its own local database with the information from every sensor Node. A GUI implementation in Manager Node uses the local database to give a graphical display of the data. System cost is considerably less because the PC's are deployed to be as Sensor Nodes or Manager Nodes.

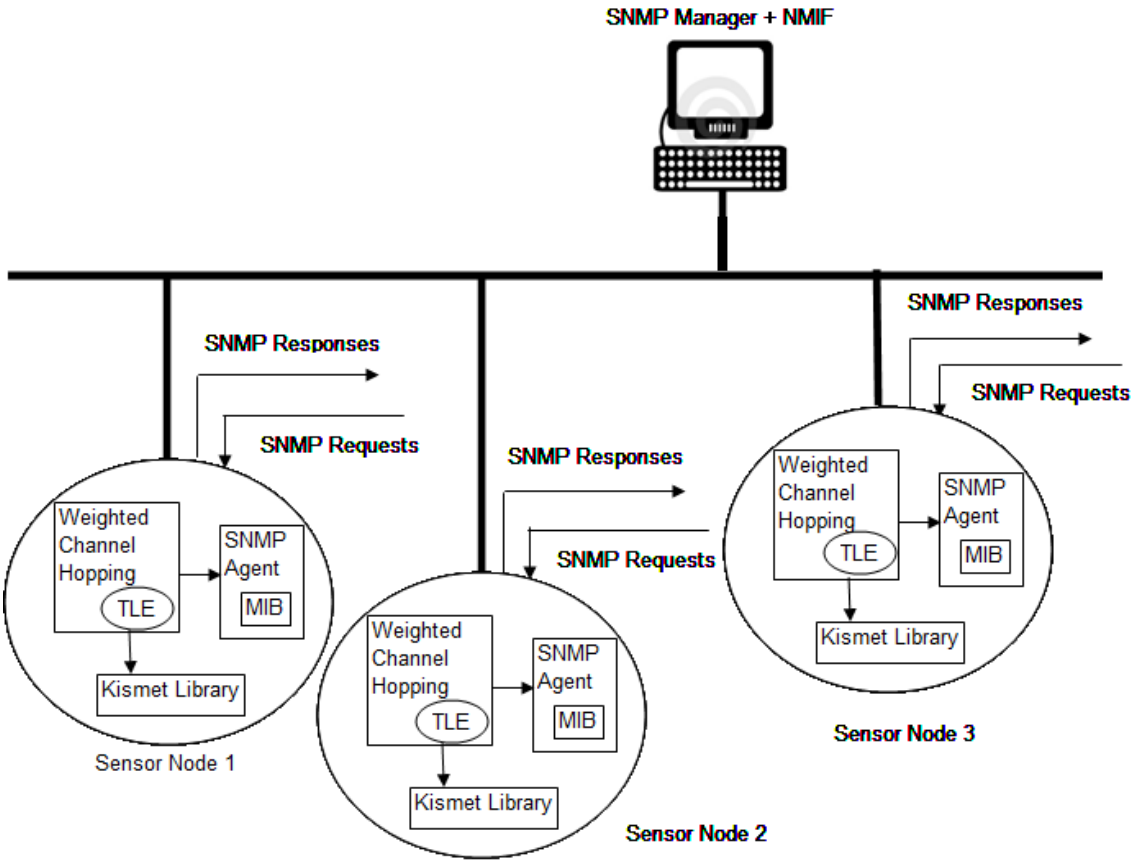


Figure 1: ***High Level Architecture of Scalable Wireless LAN, TLE – Traffic Load Estimation***

Figure 1 gives the overview of the Scalable WLAN Monitoring. The system consists of Nodes which transmit data to the manager node. Each and every Node sniff, monitor the traffic in the air and send the statistics to the Manager node. The Manager node exposes the real time view of the wireless network using NMIF, Network Management Interface Framework. The statistics of every packet helps in calculating the channel busy time.

The system starts with weighted channel hopping, wherein there is frequency hopping to every channel in the radio to collect some initial statistics regarding every channel. Choice of channels to inspect is dependent on the weighted channel hopping, because these channels could be of interest providing valid information of the radio spectrum. The information collected from these channels serve for two main purposes. First, calculate the channel busy time, to estimate the accuracy of channel usage because of the unlicensed spectrum use in WLAN network. Second, to provide a clear view of the statistics collected from every node as GUI.

3.4 Design and Implementation

The WLAN monitoring is based on a distributed system where in the task of data collection is distributed for further analysis. The collected statistics helps in calculating the accurate channel busy time which is an uncertain feature of the radio channel. With such data that is distributed, we need a system that coordinates the collection of information. There exists a manager system which co-ordinates the collection of statistical data. Scalable Monitoring makes use of Simple Network Management Protocol (SNMP) as the basic transport mechanism of collected data. One another advantage of using SNMP is that it doesn't affect the performance of the wireless network Secondly, by using SNMP our system can be interfaced with already running SNMP agents to collect data from them and perform in-network processing.

3.4.1 Design of Wireless LAN Traffic Data Sensor

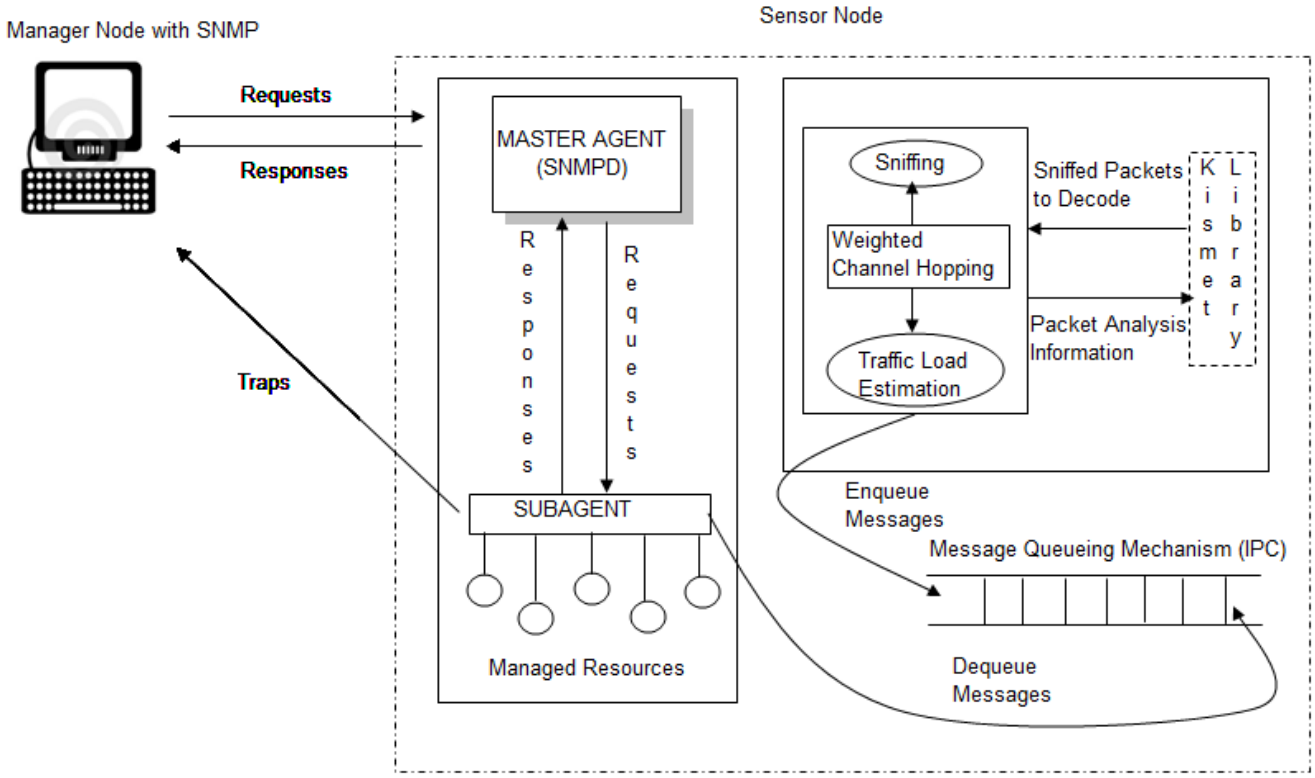


Figure 2: ***Low Level Architecture of the system connecting to every component***

Figure 2, gives a very brief description about the components of a single node and there interaction with the SNMP Manager. Weighted Channel Hopping collects interesting information from the channels uses Kismet Library to detect the anomalies present in the network. This application generates a database of all the statistical data, which would be further used by the sub-agents through the IPC-mechanism. Sub-agent is an AgentX thread which binds with the Master Agent on start-up fetches the management data through message queues, IPC-mechanism and then redirects it to the SNMP Manager through the SNMP Agent. SNMP Manager is now very intelligent to display the current position or status of all the channels and their usage.

3.4.2 Weighted Channel Hopping

For channel hopping, various designs were considered to choose an optimal one based on the requirement. Three designs were considered they are as,

In Design 1, the following steps are considered with the assumption that the number of AP's using a specific channel is likely to introduce huge load in the channel.

1. Hop through the channels to identify the AP's in the network
2. Create a Table to consist of the channels ordered with the maximum channel usage.

If 3 AP's in a channel is considered to be the threshold value then,

- 1st List holds the values greater than or equal to three
- 2nd List has values less than three

3. Hop through the channels with maximum usage alone, assuming the fact that AP's alone contribute to the channel load.
4. After an interval, update the table by proceeding from step 1

The flaw identified in this design is that, the stations associated with AP are not considered as channel load, so if there are eight AP's in a channel but without any association of stations, then frequently hopping this respective channel is a poor design. Therefore an AP with association of many clients can increase the traffic load drastically, which may be unnoticed.

Design 2, is a slight variation of design 1, but stations in the channel are also considered to add to the load of the channel.

1. Hop through the channels to identify the AP's and their associated Stations.
2. Create a Table to consist of the channels ordered with the max channel usage.

If 3 AP's in a channel is considered to be the threshold value then,

- 1st List holds the values greater than or equal to 3
- 2nd List has values less than 3 and greater than 1.

3. Hop the channels with max usage, modified if max count is less than 3.
4. After an interval, update the table by proceeding from step 1

The Channel without an AP is considered as an idle channel, because no other stations can associate with it. Therefore, channel hopping can be avoided for this channel for a sample period. Completely avoiding scanning a channel because of the absence of APs can lead to some anomaly or load being unnoticed and the claim of traffic load can be proved wrong.

The third design is, weighted channel hopping followed from weighted round robin, wherein frequency hopping is done on every channel for a fixed sample time to create the ACTIVE CHANNEL List. The active channel list distinguishes itself from the remaining by having a good history of Access Points and Stations using that channel. The channels

in Active List have weights greater than the other channels therefore the sampling time varies based on the weight. Now passive scanning is done to retrieve much more valid information from every channel. Kismet is a third party utility tool which is used by the weighted channel hopping. Every packet that is sniffed from the channel of active list is parsed using Kismet's API and then categorized to Access Points and its associated stations. In the complete process of sniffing every packet is parsed to notify security alerts.

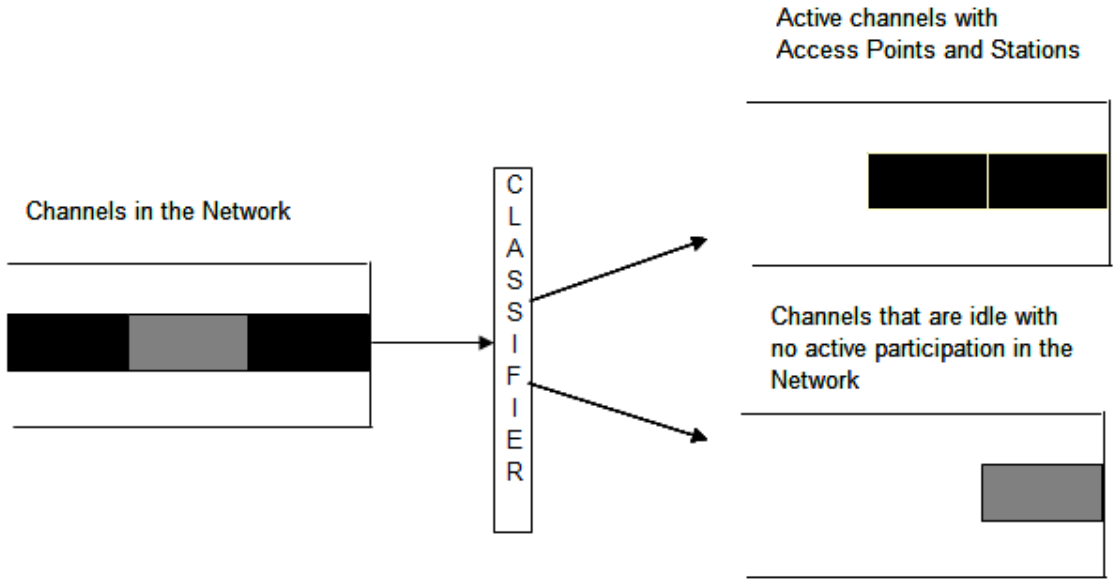


Figure 3: ***Weighted Channel Hopping***

Kismet [6] is used as a library in Scalable WLAN traffic monitoring. It's an 802.11 wireless LAN sniffer, and provides several intrusion detection functionalities. It is able to work with any wireless NICs that support promiscuous mode. The constraint of too many processes and scalability made the choice of Kismet being a library. Therefore, Kismet is mainly used to detect the anomalies in the network. The packets sniffed carried lots of information regarding the anomalies whose details are obtained by again parsing through the Kismet API's.

The sniffed raw packets from weighted channel hopping application are given as input to a Kismet API, GetPacketInfo, which reduces the packets to its equivalent IEEE header details and returns a packet structure. This packet structure is again an input to the ProcessPacket API of kismet to identify the AP or Station or any other security alerts. Through the Kismet API's a vector of Wireless Network/AP and Wireless Clients/Stations are formed in the Channel Hopping Application, which judges the active channel list based on this vectors.

3.4.3 SNMP Agent, AgentX and Manager

SNMP [19], Simple Network Management Protocol is used as a transport mechanism to exchange the collected data. SNMP has three versions as v1, v2c and v3, but the scalable WLAN traffic monitoring uses SNMP v3, to utilize the secured SET operation, which distinguishes itself from previous versions. Since SNMP is a UDP-based network protocol, the performance of the system is never degraded. SNMP is a choice because authorization and authentication is required to exchange the data between the sensor node and the managed node. SNMP Agent is the client machine to store the management data, the manager or master can query the Agent for management data.

The SNMP protocol supports scalar or tabular values and has the following operations as GET, GETNEXT, GETBULK, SET, TRAP or INFORM to be effectively used in the exchange of management data between the Sensor Node and the Managed Node (Master). SNMP protocol supports polling or trap. By Polling, the manager can query the SNMP Agent and in trap the agent can inform the manager. GET, GETNEXT and GETBULK are the operations used by manager node to query for statistical data, SET is again used by Manager to modify the content at the Agent and TRAP/INFORM is used by the agent to notify the Manager immediately on some adverse conditions. SNMP uses MIB's [20] (Management Information Base), that defines the structure for the stored management related data which basically is a virtual database. The objects in MIB are defined using subset of ASN.1 [21] (Abstract Syntax Notation) called as Structure of Management Information.

SNMP Agent

SNMP Agent is present in the sensor node whose ideal job is to fetch the management data from weighted channel hopping and to store in the form of MIB. Since the data from the weighted channel hopping application is a database of access points and stations in the network, the MIB tabular storage feature is chosen to make the retrieval much simple. SNMP MIB structures for polling request from SNMP Manager are as follows,

Attribute Name	Data Type
apBSSID	String
apName	String
apChannel	Interger32
apChannelPercent	Integer32
apMode	String
apPkts	Integer32
apDataSize	Integer32

Table 1: ***Structure of apTable MIB***

Attribute Name	Data Type
stationMAC	String
stationCryptPkts	Integer32
stationDataPkts	Interger32
stationDataSize	Integer32
stationDataRate	Integer32
stationAP	String

Table 2: ***Structure of stationTable MIB***

Attribute Name	Data Type
channelNumber	Integer32
channelAPs	Integer32
channelStations	Interger32
channelPkts	Integer32

Table 3: ***Structure of channelTable MIB***

Attribute Name	Data Type
apInfoStatBSSID	String
apCarrier	String
apCrypt	String
apSignal	Integer32
apMaxrate	Integer32
apMaxSeenRate	Integer32
apFirstSeen	Timestamp
apLatestSeen	Timestamp
apSignalPower	Integer32
apSignalNoise	Integer32

Table 4: ***Structure of apInfoStatTable MIB***

Attribute Name	Data Type
pktsFromAP	Integer32
pktsData	Integer32
pktsCrypt	Interger32
pktsWeak	Integer32
pktsLLC	Integer32

Table 5: ***Structure of pktsTable MIB***

The above MIB structures basically serve the purpose of polling from a Manager. There are MIB structures being defined for TRAP from SNMP Agent. A TRAP is like an interrupt from the Agent to notify the Manager on adverse conditions. Traps are mainly designed to notify the anomalies found in the network. The SET operation of SNMP is used to monitor on a specific anomaly.

The disadvantage of changing SNMP Module to adopt new module support is overcome by using a protocol called as Agent Extensibility (AgentX), which supports dynamically loadable modules. Therefore, AgentX [22] is the subagent, which runs as separate process binds with the SNMP Agent (master agent), and then the exchange of information happens between the weighted channel hopping application, master agent and AgentX.

Net-SNMP was used to generate the template code for SNMP. Mib2c is a tool support in Net-SNMP, with the definition of every MIB in a configuration file, can be given as input to mib2c along with the type of accessible data either mib2c.scalar.conf or mib2c.tabular.conf. The output is a template code with the usage of all Net-SNMP API's. The Mib2c tool reduces the effort of learning the Net-SNMP in-and-out. The sub-agent is a different main file to be written with a specific call to initiate the module configuration MIB file.

Message Queue Mechanism has been used for Inter-Process Communication between Weighted Channel Hopping Application and SNMP Agent. The Channel Hopping application which has the details of access point structures, station structures writes into a message queue. SNMP Agent, another process which frequently checks for the update of a message, retrieves the very recent data from the message queue.

SNMP Manager is the Network Management System used by the network administrator to invigilate the collected data from every SNMP Agent. SNMP Manager is responsible to fetch the management data from SNMP Agents periodically through the method called as polling to always give a very clear view of the WLAN usage. SNMP Manager can also receive TRAPS from SNMP Agents when an anomaly or security issue happens in the network.

Traffic Load Estimation algorithm and security alerts are also generated from sniffed raw packet frames and are discussed in following chapters.

Chapter 4

Traffic Load Estimation Algorithm

WLAN's can be managed effectively if the load for each Access Point is known; instead estimating the channel's usage on each WLAN is worthwhile. There are three reasons why channel usage estimation can be beneficial than the load at AP. Firstly, the channel usage statistics can clarify a network administrator between a wired and wireless congestion. Secondly, it would be useful to balance the load across different channels and thirdly, any anomaly that abuses the channel can be easily identified.

Traffic load estimation algorithm is an alternative algorithm based on the percentage of channel time the AP consumed for frame transmissions. The Weighted Channel Hopping and the statistics would be helpful to estimate a radio channel's usage and the traffic load on the associated AP. Channel busy time is the amount of radio channel time used for frame delivery. The channel busy time includes the channel time due to successful or corrupted frame transmissions, inter frame spacing, as well as the channel idle periods in which the stations are in the back-off mode. The reason to include back-off time in the channel busy time is because; back-off time is a non-trivial amount.

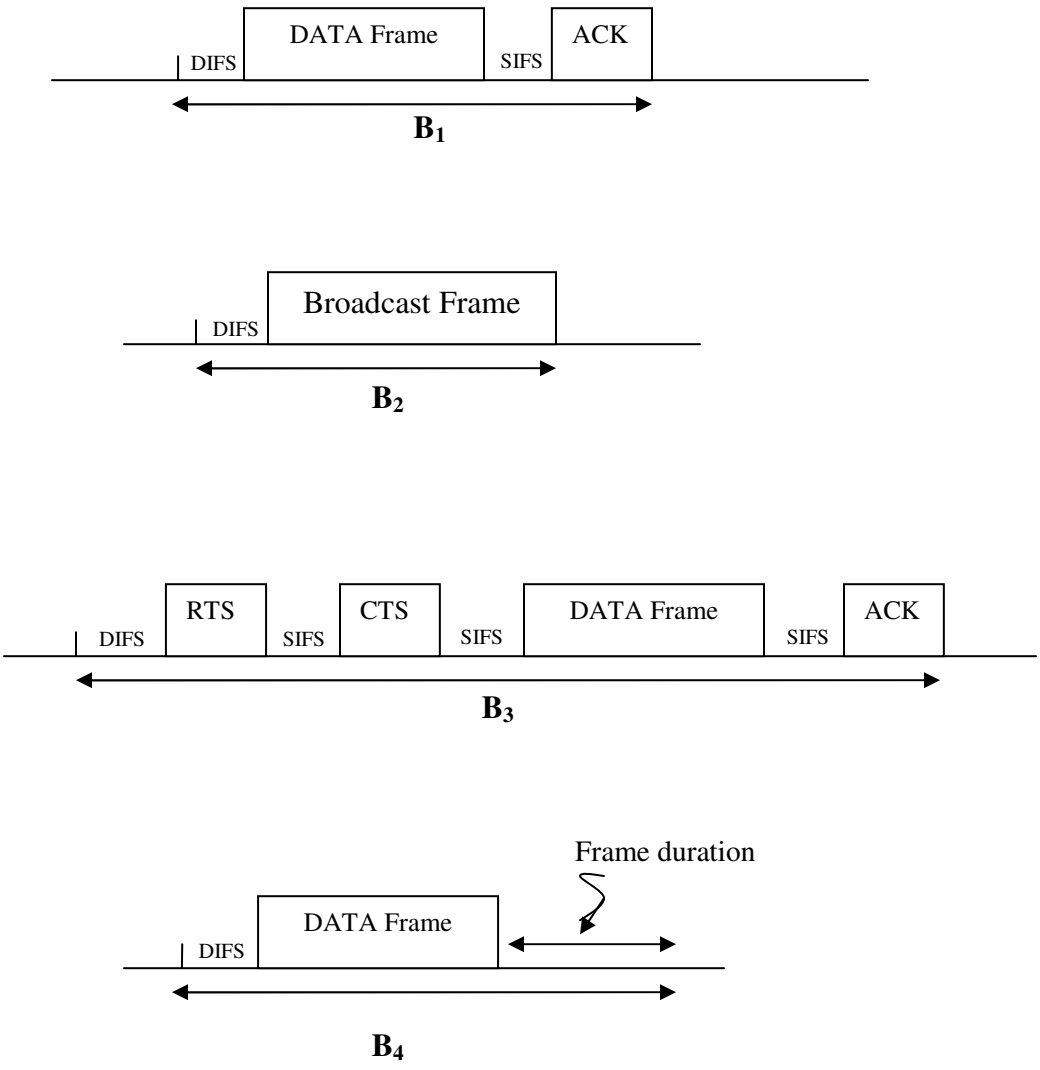
The algorithm includes busy periods of a packet, back-off time and true channel idle time to estimate the traffic load.

4.1 Busy periods of a packet

A sequence of link-layer frames used to deliver a single management or data frame represents a frame transmission transaction. Figure 5 shows the busy period composition for different IEEE 802.11 frames. Each busy period corresponds to one frame transmission. A data frame, broadcast frame along with their inter-frame spaces as specified in IEEE 802.11 constitute in the busy period of a channel. The most important three factors considered in this algorithm are the BUSY Period, IDLE Period and the BACK-OFF Period. At any point in time, only one frame transmission transaction can occur. I_j represents the idle period immediately preceding B_j, the busy period of the jth frame transmission transaction. Each I_j is further divided into true channel idle period, and back-off period. The summation on back-off period represents the medium access control overhead due to channel contention. True channel idle is the spare capacity of a radio channel. Therefore both the busy periods and back-off periods are part of the channel busy time.

Busy Period represents the frame transmission associated with their inter-frame spaces. Every packet frame has DIFS, distributed inter-frame space and SIFS, short inter-frame space followed by a data or an acknowledgement frame. These inter-frame spaces introduced to avoid collision contribute to the channel busy time. EIFS, Extended Inter-

frame space is included when there is a collision of the packet. Every station that would like to use the channel has to wait for a frame space and then sense the carrier to proceed with transmission. Figure 5 describes the inter-frame spaces associated with every frame.



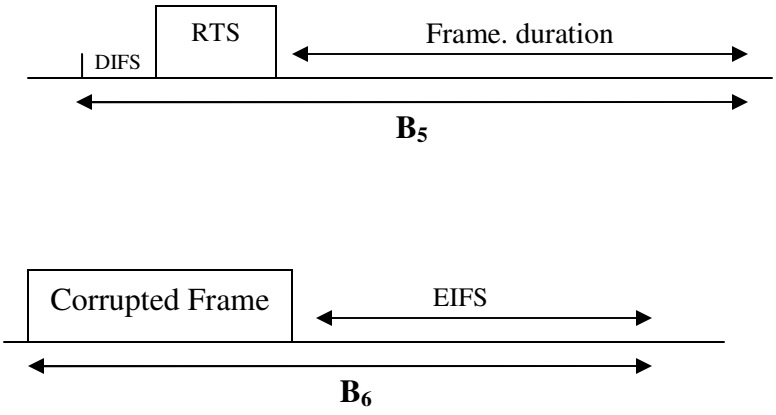


Figure 4: ***The busy periods of different types of IEEE 802.11 frames transmission transactions have different compositions. B1, B2, B3, B4, B5, B6 represent the busy periods of a data frame delivery, a broadcast frame delivery, a data frame with RTS/CTS delivery, a data frame without acknowledgement, an RTS without CTS reply, and a corrupted frame.***

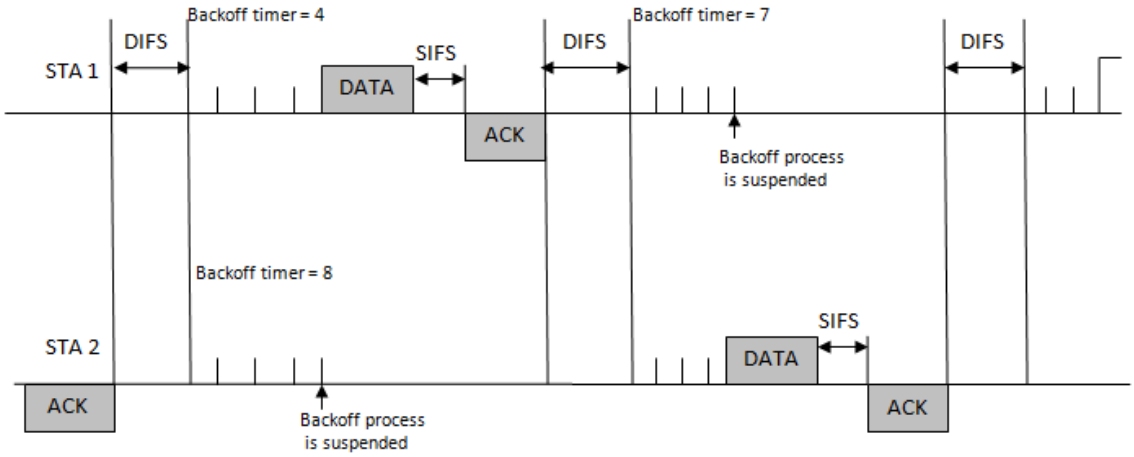


Figure 5: Two IEEE 802.11 stations access the radio channel simultaneously. Station 1's back off period is overlapped with stations 2's back-off period.

From a sniffed frame trace, start and end of each frame transmission transaction have to be identified. This requires enumerating all possible frame transmission transactions in IEEE 802.11 standard. There are three different possibilities; Equation 1 corresponds to a frame transmission transaction that has been completely captured, including the link layer frames and their associated inter-frame spaces. Equation 2 corresponds to a frame transmission transaction I which the receiver does not respond to the sender because it does not receive the frame correctly or is out of the communication range, because IEEE 802.11 employs a virtual carrier sensing mechanism to reserve the channel for the following frame within the same frame transmission transaction, all other stations

consider the medium to be busy for the time define in the *duration* field. Equation 3 corresponds to a frame transmission in which a corrupted frame is transmitted.

$$B_j = \sum_{k=1}^{N} Frame_k^j + (N-1) \times SIFS + DIFS \qquad (1)$$

$$B_j = Frame_1^j + Frame_1^j.duration + DIFS \qquad (2)$$

$$B_j = Frame_1^j + EIFS \qquad (3)$$

4.2 Back-off Estimation

Estimating the back-off time accurately is the key to deriving the total channel busy time. When an IEEE 802.11 WLAN node sends a frame, if it senses the channel is busy then it waits for back-off interval, whose length is randomly chosen from the range [0, CW], where CW (Collision Window) is initially set to 31 channel time slots and doubled for every retransmission. The average value of the back-off intervals is CW/2. A naïve way to derive back-off interval is to simply assume CW/2 time slot as the back-off interval for transmitting each frame. This will cause the over-estimation of total back-off time, because multiple stations can be in the back-off mode at the same time. The following algorithm describes a heuristic to determine the value of the back-off period from frame traces.

4.3 True Channel Time Estimation

The following heuristic is to determine the value of back-off period. If the idle time I_j is smaller than CW/2, then I_j is assigned to back-off period. If I_j is between CW and CW/2 then either I_j or CW/2 is assigned as back-off interval based on a control parameter called the back-off interval adjuster (BIA). BIA is initially set to 0 and updated each time as per the back-off interval.

$$BIA_j = BIA_{j-1} + \frac{CW/2 - I_j^B}{CW/2}$$

In addition, BIA is bounded by upper and lower bound, BIA_{max} and BIA_{min}. In most cases, I_j is assigned to I_j^B when I_j is between CW and CW/2. However, this over estimates the average back-off interval in the long run, because the long term average of I_j^B should

I_j^B no more than CW/2. To correct this, BIA is introduced to "remember" the number of times I_j^B may bI_j^Bver-estimated. When the number of over-estimates reaches a threshold, i.e, when BIA is equal to BIA_{min}, then CW/2 is assigned to I_j^B. This continues until BIA is larger than BIA_{min} again. Empirically, 3 and -3 are good choices for BIA_{max} and BIA_{min}, respectively.

True Channel Idle Estimation Algorithm

$BIA_0 = 0$
For $j = 0$ *to* N ***do***

Read $< I_j , B_j >$

$I_j^B = I_j$

If $I_j > CW$ ***then***

$I_j^B = (CW/2)$

Else if $I_j \leq CW$ *and* $I_j > (CW/2)$ ***then***

If $BIA_{j-1} = BIA_{min}$ ***then***

$I_j^B = (CW/2)$

End if

End if

If $\dfrac{CW/2 - I_j^B}{CW/2} \geq 0$ ***then***

$$BIA_j = min\left(BIA_{j-1} + \frac{CW/2 - I_j^B}{CW/2} , BIA_{max}\right)$$

Else

$$BIA_j = max\left(BIA_{j-1} + \frac{CW/2 - I_j^B}{CW/2} , BIA_{min}\right)$$

End if

$I_j^T = I_j - I_j^B$

End for

$$Backoff\ Time = \sum_{j=1}^{N} I_j^B$$

$$True\ Channel\ Idle\ Time = \sum_{j=1}^{N} I_j^T$$

Chapter 5

WLAN Security

Security is the main concern which comes second after the performance. The *scalable WLAN monitoring* system identifies various security issues involved in the wireless network. The sniffed packets are carefully decoded to identify serious problems within the network. The exponential growth in the deployment of IEEE 802.11-based wireless LAN in enterprise makes WLAN an attractive target for attackers.

5.1 MAC-Address spoof based attacks

Exploiting link-layer protocol vulnerabilities require a different set of intrusion detection mechanism. Most link-layer attacks in WLAN's are denial of service attacks and work by spoofing either access points (APs) or wireless stations. By MAC address spoofing, various denial-of-service attacks are possible.

A station must authenticate and associate with an AP before it can communicate with the AP. The IEEE 802.11 standard provides de-authentication and disassociation frame for the STA or AP to terminate the connectivity between them. Unfortunately, the de-authentication and disassociation frames themselves do not come with sender authentication. Consequently an attacker can send a spoofed de-authentication and/or disassociation frame on behalf of the AP to STA or vice versa and eventually stop the data communication between the STA and AP. The result is a Denial-of-Service (DoS) [40] attack. By spoofing to de-authenticate or dissociate packets, arbitrary or all clients can be disconnected from the network. When the AP receives a spoofed de-authentication frame, it will remove all the state associated with the victim STA. Our test shows that if the victim STA does not send any data to the AP, the AP will silently drop any frames destined to the STA. This means that the victim STA is disconnected from the AP unknowingly. Only when the victim STA starts sending frames will the AP send a de-authentication frame to the STA, which then repeats the authentication process. Many attacks use a broadcast disassociate or de-authenticate to disconnect all users on a network, either to redirect them to a new fake network or to cause a denial of service or disclose a cloaked SSID. Broadcast disassociations are rarely, if ever, legitimate.

AP Spoofing is yet another MAC address spoofing wherein an attacker can spoof the MAC Address and SSID of a legitimate AP. This AP spoofing is called the Rogue AP in security field, since this AP shows up with stronger signal strength, there is high likely for stations to detect this AP and associate. Rogue AP is one of the serious security concerns because it can impersonate to be an Access Point within the network and can later on be a denial-of-service attack or Man-in-the-Middle attack to gain user credentials. The Intrusion Detection System of Kismet has been widely deployed to identify every anomaly in the network. WEPWedgie [41] is yet another tool which helps

in scanning the network once station spoofing acquires AP's MAC address-based access control list to gain access to a WLAN.

5.2 Radio NIC Spoofing

Another eloquent method for denying service includes fooling valid radio NICs with fake 802.11 frames. For example, someone could setup their radio NIC (or 802.11 frame generator) to send a continuous stream of CTS (clear-to-send) frames, which mimics an access point informing a particular radio NIC to transmit and all others to wait. (CTS is part of 802.11's RTS/CTS function.) The radio NIC being given permission to transmit could be a fictitious user. As a result, the legitimate radio NICs in end user devices will continually delay access to the medium.

A Man-in-the-middle attack technique can also happen by impersonating a legitimate NIC, called rogue radio NIC [42]. After gleaning information about a particular wireless LAN by monitoring frame transmissions, eg. Wireshark, a hacker can program a rogue radio NIC to mimic a valid one. This enables the hacker to deceive the access point by disassociating the valid radio NIC and re-associating again as a rogue radio NIC with the same parameters as the valid radio NIC. As a result, the hacker can use the rogue radio NIC to steal the session and carryon with a particular network-based service, one that the valid user had logged into.

5.3 Jamming based Denial-of-Service attacks

Jamming or causing interference to an 802.11b network is simple. A jammer or micro-wave can cause the maximum amount of interference. There are several commercially available devices that that will bring a wireless network to its knees. For example, a Bluetooth-enabled device can disturb 802.11b networks, when located approximately ten meters of 802.11b devices, the Bluetooth device will cause a jamming type of denial-of-service attack. The same is true of several 2.4GHz cordless phones that are currently available. This is because the 2.4GHz band is becoming widely used and is considered shared, thus allowing all kinds of devices to use it.

The signals generated by these devices can appear to be an 802.11 transmission to other stations on the wireless network. The other possibility is that the devices will just cause an increase in RF noise, which could cause the 802.11b devices to switch to a slower data rate. Denial-of-Service attack on *virtual jamming* [39], physical jamming may prohibit victim nodes from accessing the shared channel by the use of interfering signals. Similarly, due to the use of the virtual carrier-sense function in the IEEE 802.11 MAC layer, well-behaved nodes may be misled by misbehaving nodes to update their NAVs (Network Allocation Vector) in such a way that they cannot access the shared channel such an attack called as virtual jamming. Since virtual jamming consumes much smaller amount of energy compared to physical jamming, it is more viable or efficient for misbehaving nodes or malicious nodes to launch such an attack.

5.4 Other Common Attacks

The BSS timestamp sent with beacons and some probe frames cannot be spoofed with standard firmware or drivers even while forging raw frames. A BSS mismatch is likely an indication of an attempt to spoof the SSID and BSSID of an access point. This alert contains flap-detection to minimize false positives caused by random bogons and AP recycling. Man-in-the-Middle attacks attempt to re-direct users to a fake AP on another channel is termed as AP Channel Change Alert. The IEEE 802.11 specification allows a maximum of 32 bytes for the SSID, however the IE tag structure allows for 256. Oversized SSIDs are indicative of an attack attempting to exploit SSID handling. Many firmware versions from different manufacturers have a fatal error when they receive a probe response with a 0-length SSID tagged parameter. In the research paper by Bellardo J. and Savage S., a host which legitimately disassociates or deauthenticate from a network should not be exchanging data immediately thereafter. Any client which DOES exchange data within 10 seconds of disassociating from the network should be considered a likely victim of a disassociate attack. Active' or 'Firmware' network scanning tools work by letting the card probe for any network and recording those that respond. These tools include NetStumbler, PocketStumbler and many others. An alert is raised when a client is seen to be probing for networks but never joins any of the networks which respond. False positives are possible in noisy/lossy situations, disabling this alert may be desirable in some installations. The performance degradation due to low-rate station is called as *Bad Fish* [43].

5.5 Implementation

The above sections described about the various security problems associated with a WLAN network. The implementation section gives a brief overview of the security alerts developed in "*Scalable Wireless LAN Monitoring and Analysis*". The WLAN monitoring system has two categories *critical system alerts* and *low level alerts*. The MAC address spoofing based alerts such as Rogue AP, de-authentication/disassociation alerts, Station's Excessive Roaming, Bad Fish, broadcast disconnect, AP Channel Change are considered to the critical alerts with the WLAN monitoring system. The other alerts such as BSS Timestamp change, long SSID, Probe No Join, AP Configuration change are low level alerts, which are unlikely to bring down the performance or disrupt the connectivity. The other alerts discussed in other sections are for future work.

Chapter 6

Network Management Interface Framework

6.1 Overview

NMIF, Network Management Interface Framework, is a GUI tool specifically designed for the goal of the *Scalable WLAN traffic monitoring* and also to enable GUI for any network related applications. SNMP Manager can retrieve data from all agents on polling, but there is no graphical interface to display the management data in human comprehensible form. Therefore, NMIF provides a clear picture of the Access Points, its associated stations in a Channel, available channels, channel usage and various other statistical details. NMIF also alerts on a security alert from SNMP agent.

There are other network related graphical interfaces available, like openNMS, AirMagnet, but these interfaces require quite a lot amount of changes to fit into our requirement. The graphical interfaces available either support the wired network or the wireless network. So we came up with a design of supporting both wired and wireless as well any network related applications. For example, ECSL has another project called as MINT - a test bed for wireless network was integrated into this graphical interface to monitor the movement of every wireless device and also to control the devices in the network.

6.2 Architecture of NMIF

Network Management Interface Framework has two most important modules, Framework and Interface Modules. Framework Module is more of a generic framework which can suffice the requirement of any network-related application, basically to be used as a library. The framework has various sub-modules such as, a graphical module (JUNG) to display the movement of any network device especially in a wireless environment. Query module (SwiXML) to specify any query of interest, such as to highlight every Access Point in a Channel, can be a query like “Access Points in Channel”, and the input could be the channel number eg: “6”. The next module is a display of all devices in the network in a Tree view. In wireless network, each channel has many access points, and every access point has stations associated with it and this dependency can be viewed as a tree. The framework design also has the implementation of alerting on any serious security alarms. Thus the framework module has the API’s to support network management related function calls.

Interface Module is the most important module which is the Manager node of “Scalable Wireless LAN Traffic monitoring”. The SNMP Manager is the Network Management Station which is solely responsible to manage all the management data from

every SNMP Agent and then re-direct the collected data to a local database. Be it be statistical data or alerts are stored in the local database (SQLite Server) by the SNMP Manager. Triggers in database notify the alerts to the above wireless manager layer. The interface module would be invoked by the framework module on start-up, the tree view, graphical view are framed by establishing a connection to the local database and displaying in the framework.

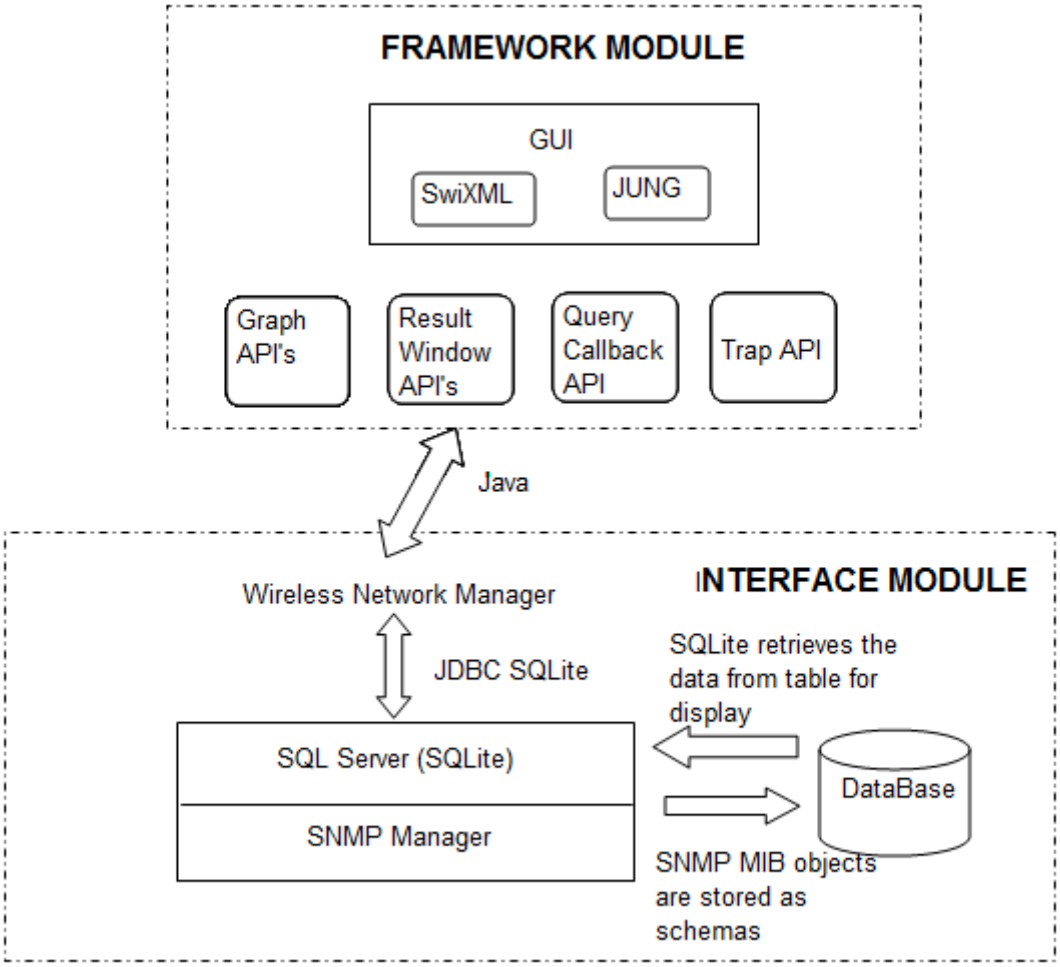


Figure 7: ***NMIF architecture***

6.3 Implementation

6.3.1 Initiation by Interface Module

The framework of the GUI is developed from an xml configuration file called as Layout.conf, which is a built-in design of framework module. The Layout.conf file was

developed using SwiXML, as layouts from swing means to be tedious, so SwiXML helps in producing Swing GUIs from XML language. The overall GUI layout is created by SwiXML. The following GUI layout is achieved by the below given XML file. The tree view is to display per channel access points or per cryption type or per mode or per ssid and so on, which can be chosen by the menu. The menu also has the support to enable or disable the alerts in the network. The query window is to display any query given by the user in the result window, which is below the topology window. The topology window, displays the graphical view of the access points.

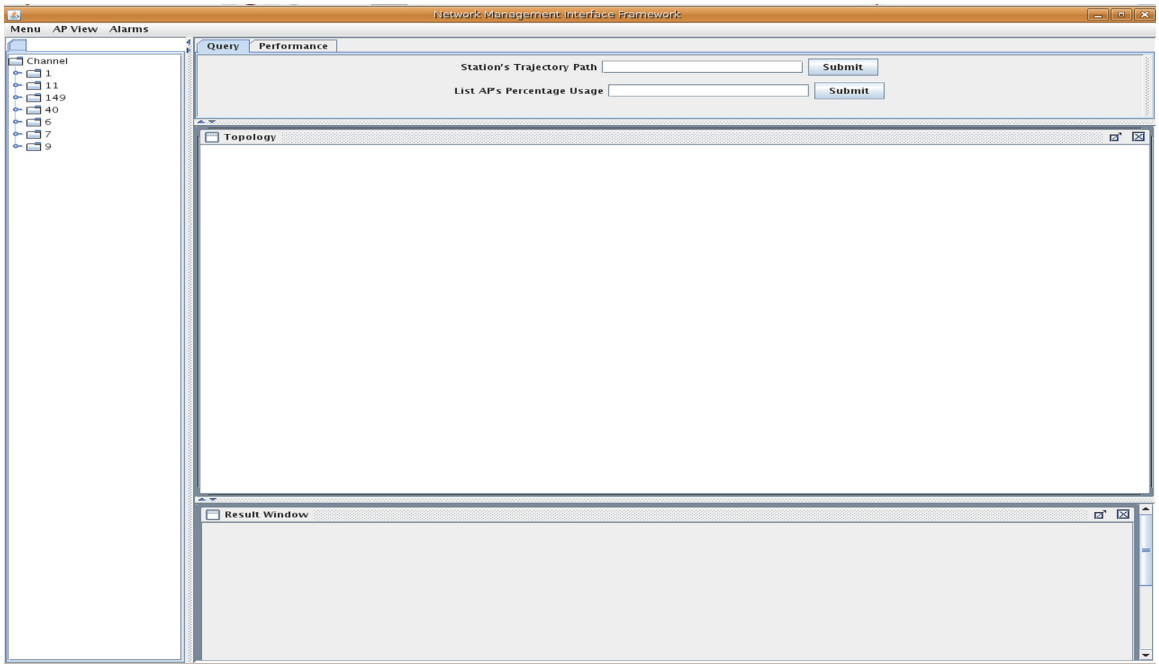


Figure 7: ***NMIF GUI Layout***

```
<?xml version="1.0" encoding="UTF-8"?>
<frame id="fr" size="810,840" title="Network Management Interface Framework"
defaultCloseOperation="JFrame.EXIT_ON_CLOSE">
   <splitpane oneTouchExpandable="true" dividerLocation="200" Background="white">
        <tabbedpane id="tbp" PreferredSize="620,480" Resizable="true">
           <scrollpane>
              <tree id="tr">
           </scrollpane>
        </tabbedpane>
        <panel layout="borderlayout">
            <splitpane oneTouchExpandable="true" dividerLocation="120" orientation="HORIZONTAL">
                <tabbedpane id="tb" PreferredSize="520,100" titles="Query" Resizable="true">
                </tabbedpane>
               <panel layout="borderlayout">
                  <splitpane oneTouchExpandable="true" dividerLocation="550" orientation="HORIZONTAL" >
                     <internalframe title="Topology" id="topo" PreferredSize="585,650" name="preview" Visible="true"
Resizable="true" closable="true" maximizable="true" background="white">
                     </internalframe>
                     <scrollpane title="Result Window" id="result" Visible="true" Enabled="false" Resizable="true"
constraints="BorderLayout.South" Background="white">
                        <internalframe title="Result Window" id="res" Visible="true" closable="true" maximizable="true">
                        </internalframe>
                     </scrollpane>
                  </splitpane>
               </panel>
            </splitpane>
```

```
        </panel>
    </splitpane>
</frame>
```

Figure 8: ***Layout.conf XML for Initiation of GUI Layout***

Wireless.conf file has components very specific to wireless domain also uses SwiXML, this file is provided by the Interface module, because the framework module was designed to be generic to accept any configuration given to the system, example either wired or wireless or any other network application. The framework module accepts this file as input and then merges with the layout.conf file to produce the expected layout as shown in Figure 8. In addition to creating components, SwiXML has support to specify actions. Every component's action field can be specified in the XML file with the Action field, by which things are much simpler to implement.

The interface module starts the whole graphical part, providing the configuration information as well as the data for the GUI. In SwiXML, since actions can be specified in the xml itself, so wireless.conf carries the configuration information and the data pertaining to Wireless domain.

6.3.2 Sub-Modules of Framework

The Framework Module has the following sub-modules other than the GUI part, graphical, query, result window and Traps.

In graphical part of framework module, JUNG is used. JUNG, JAVA Universal Network/Graph Framework, helps in creating the network or graph from the provided data. The graphical part of exposing the Access Points and its associated stations are represented as a graph using JUNG. This JUNG has the capability to represent a directed or undirected graph, while a directed graph can be quite useful in the wired network. Apart from the network graph, more details of a node in the network at many levels. Three levels are designed in this module, first level is to view the most important relevant details of that node, example for a Access Point, its carrier type, cryption type etc. are show. In the second level, every other detail of Access Point are shown, and in third level a graph of its station are shown.

In query sub-module, the interface module can specify the queries to be supported within the system. Query.conf file is given as input by the interface module which is used as callback functions to invoke a Java method based on the selection made by the user. Queries are considered to be a specific module to support some of the interesting features within a domain, example trajectory path of a station would expect a station's MAC Address as an input, and then displays the trajectory path in the graphical portion.

In result window sub-module, some of the query based results are shown here as inputs, specifically if the distribution of a channel is being queried, then the result cannot overlap with the graphical view of the network rather shown as a histogram or pie chart in the result window.

In Traps module, any alerts which are notified by SNMP Agent to the Manager are to be immediately shown up in the graphical view.

6.3.3 Sub-Modules of Interface

Wireless Network Manager is the main component of the Interface Module. The whole system starts up from this module because framework module is used as a library. This module initiates by providing the wireless configuration file and the data to generate the graph and other details with respect to the framework. Wireless Network Manager extracts data from the SQLite server through SQLiteJDBC. SQLite server is populated by the data from SNMP manager, which has MIB objects being transferred from SNMP Agent of the Sensor nodes.

6.4 Snapshots

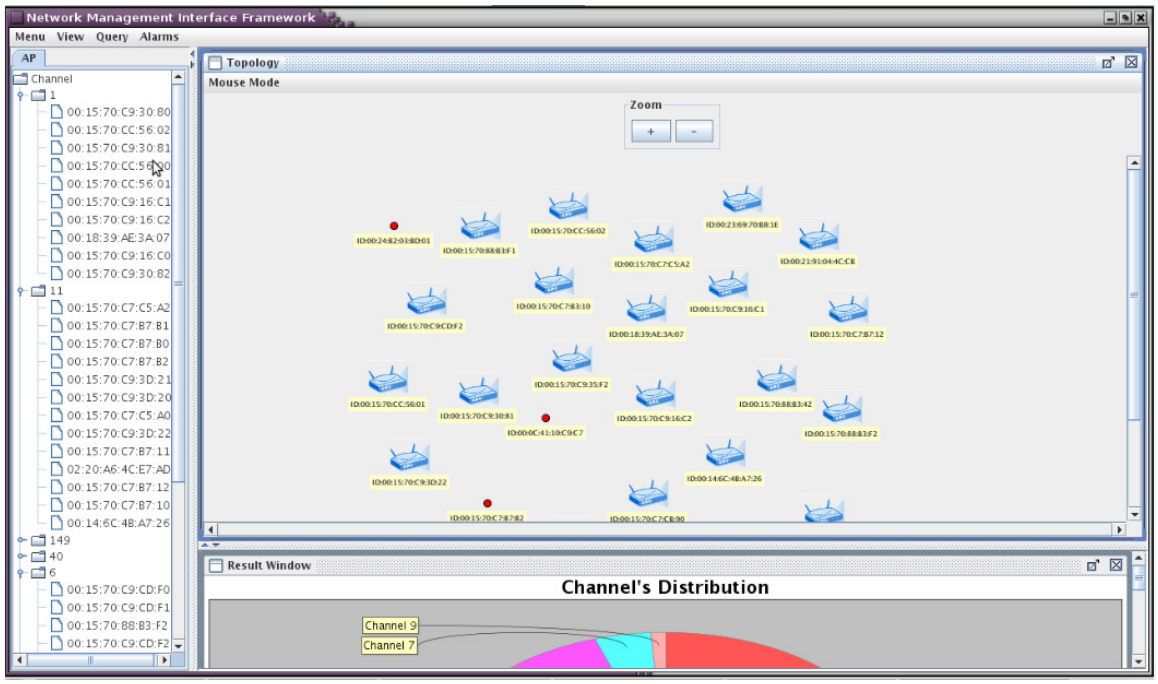


Figure 9: ***NMIF, with alerts being displayed as red spots in place of the Access Point, specifies the fact that those Access Point have some serious alerts.***

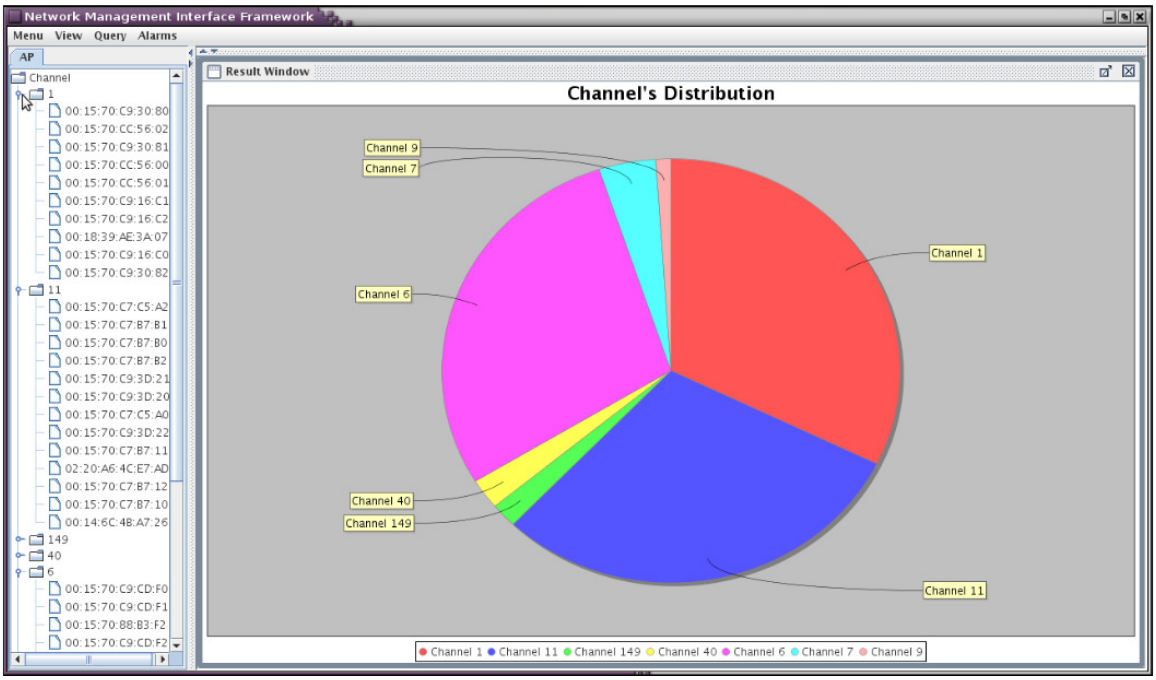


Figure 10: ***NMIF, with Channel's Distribution in the radio spectrum***

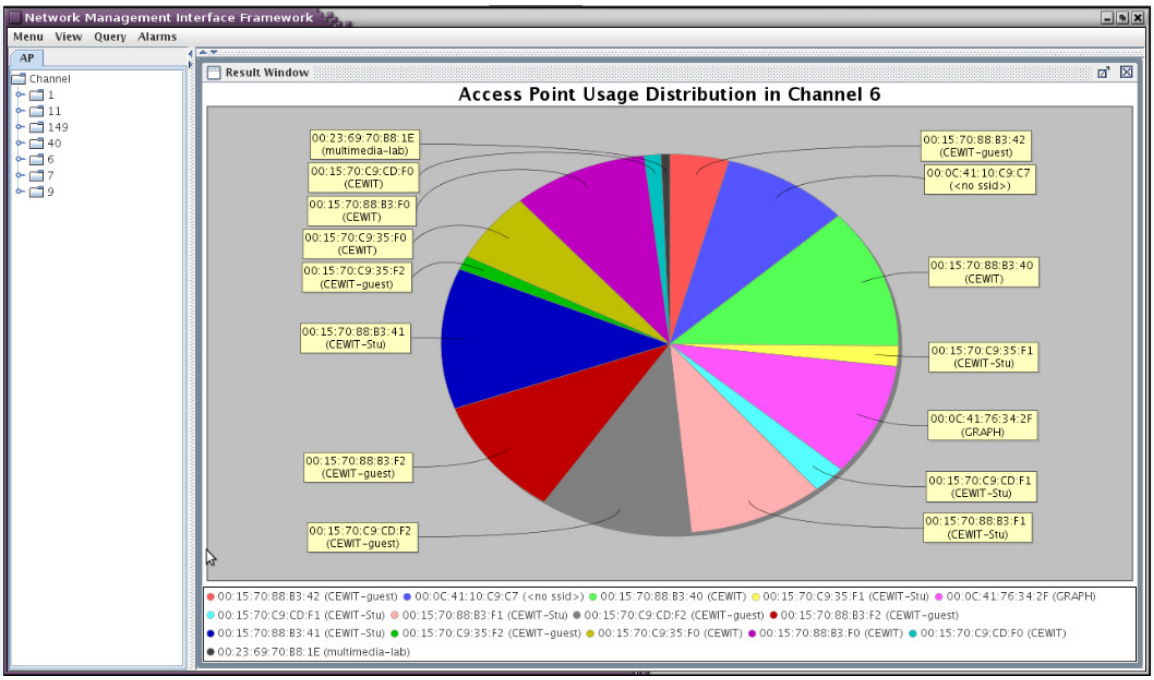


Figure 11: ***NMIF, each Access Point's load in Channel 6***

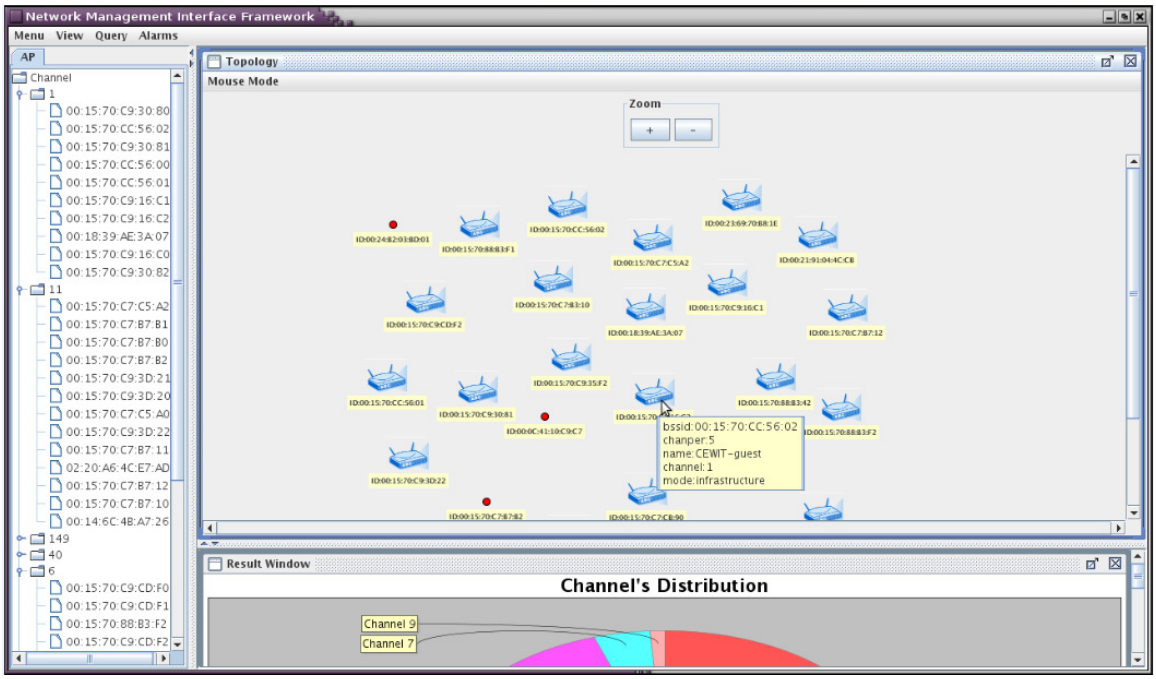


Figure 12: ***NMIF, mouse move-over on a Access Point displays most important information of the AP such as its BSSID, mode, SSID, Channel Used Percentage and Channel***

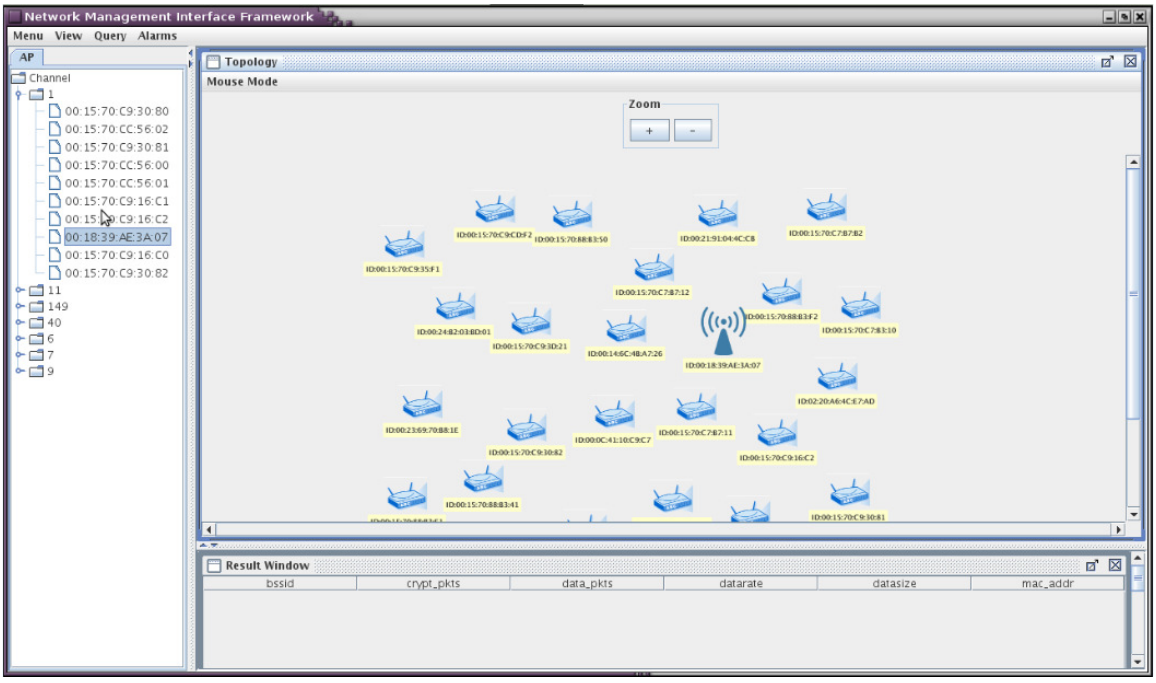


Figure 13: NMIF, a click on the tree vie 1Figure 13: NMIF, a click on the tree view of Access Point chooses the corresponding node of the graph.

Chapter 7

Evaluation

7.1 Traffic Load Estimation evaluation

The traffic load estimation algorithm involved identifying the busy periods, back-off time interval heuristically and true channel idle time. To double-check the accuracy of true channel idle time estimate, it is converted to available bandwidth, and empirically measured if that much bandwidth is indeed available from the channel. The conversion from channel time to bit rate depends on many factors, such as frame size, collision possibility, and PHY transmission rate, etc. Because the goal here is to verify the channel idle time estimate rather than to predict the available bandwidth, the additional load used to measure the channel idle time has the same characteristics as that before the measurement. So the Estimated Available Bandwidth is derived by multiplying the current traffic load by the ratio between the true channel idle time (TIT) and the sum of channel busy time and back-off time as follows,

$$\text{Estimated Available BW} = \frac{\text{Total Bits}}{\text{Total Time} - \text{TIT}} \times TIT$$

Where Total Bits represent the total number of bits successfully transmitted in the measuring period and Total Time is the elapsed time of the measuring period.

Therefore, a synthetic set-up was carried over with 4 stations to verify the estimation of the algorithm. The synthetic set-up consists of two steps to verify the available channel bandwidth. Each station sends UDP packets of length 1460 bytes.

2-steps of Synthetic Set-up

1. Three stations are associated with an AP with their consistency in the data rate and then to calculate the channel traffic load and available bandwidth
2. 4th station injects packet to the AP without disturbing the data-rate of three other stations is to verify the estimated available bandwidth with empirical one.

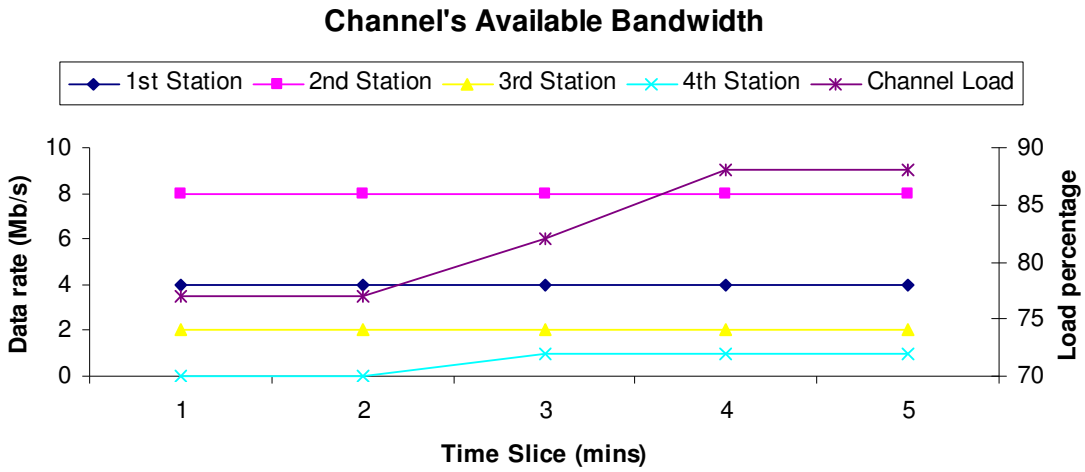


Figure 14: ***Channel's Available Bandwidth Estimation***

7.2 Channel Load and Access Point Load

From the traffic load estimation algorithm, channel's usage in the radio spectrum and APs distribution in a channel are determined. The graph drawn is an output of testing carried over a very long period of three to four weeks. The last graph is the anomalies found in the network for the four weeks period.

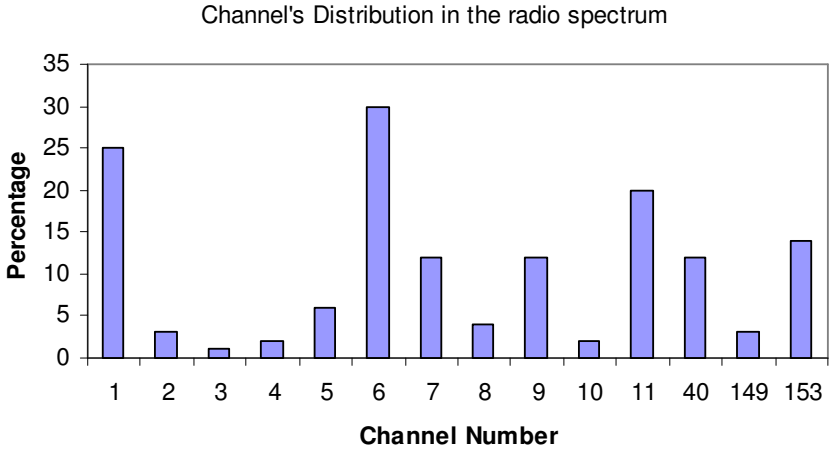


Figure 15: ***Channel's load in the radio space***

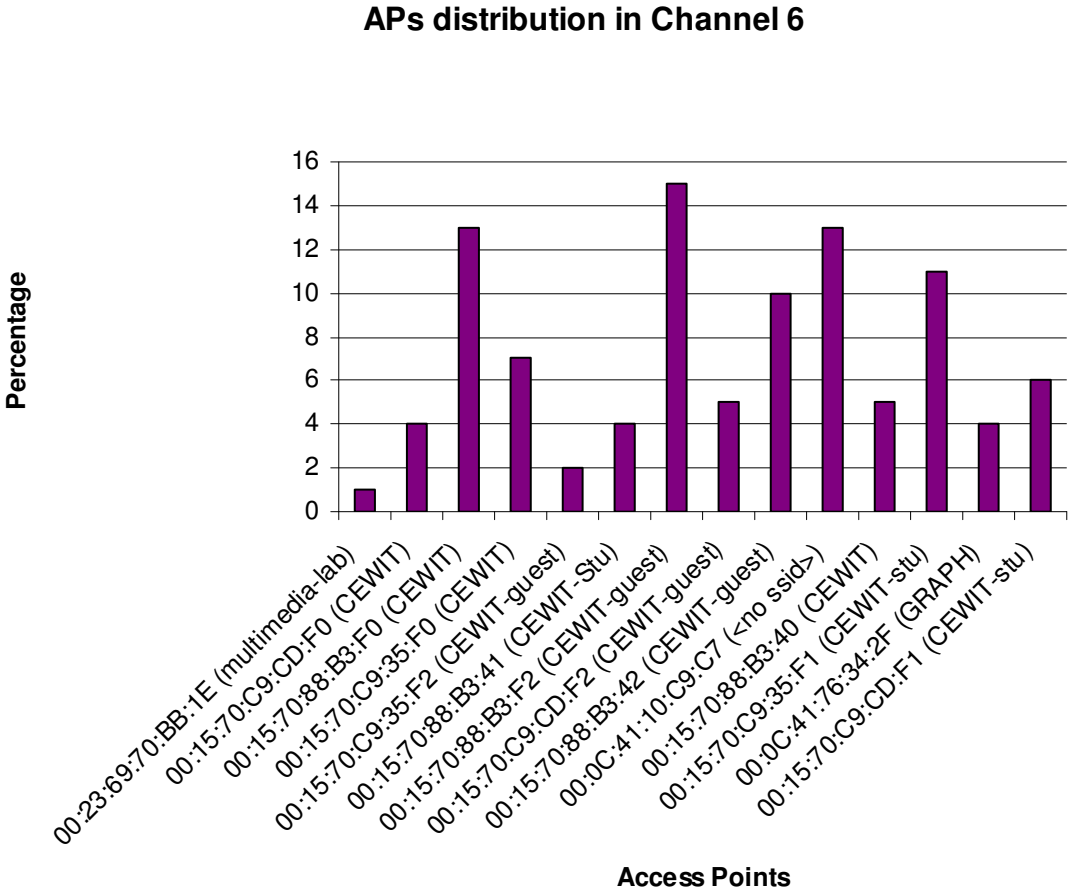


Figure 16: ***AP's present in a Channel***

Figure 17: ***Various Security Alerts which are described in Chapter 5's output***

Chapter 8

Conclusion and Future Work

In this thesis, management and real visibility of the WLAN was presented. The recurring issues in Wireless Network has been stated and most important area within have been identified and was solved. The traffic load estimation algorithm will play an important factor to unroll all uncertainties involved in the wireless network. NMIF, GUI gives a user perceivable way of reaching out to solve the wireless network problems.

Future Work includes,

1. Two most important advantages from the traffic load estimation algorithm is to control the wireless network,

- As the algorithm depicts channel's usability, the Access Points in the channel can be controlled to move across the channels.
- As well as if the Access Points are heavily loaded then the stations can be associated with other Access Points nearby.

2. MAC address spoof detection by signal strength indicator can be added to the system.

3. Virtual jamming, Denial-of-service attack can also be implemented in the system.

Bibliography

[1] C. C. Ho, K. N. Ramachandran, K. C. Almeroth, E. M. Belding-Royer; "A scalable framework for wireless network monitoring;" Proceedings of the 2nd ACM international workshop on Wireless mobile applications and services on WLAN hotspots, 2004

[2] Symbol Technolgies Inc. "SpectrumSoft: Wireless Network Management System;" http://www.symbol.com

[3] AirWave; "AirWave Management Platform;" http://www.airwave.com/

[4] AirMagnet; "AirMagnet Inc." http://www.airmagnet.com/

[5] AirDefense; "AirDefense Inc." http://www.airdefense.net/

[6] Kismet; http://www.kismetwireless.net/

[7] D. Kotz K.Essien; "Analysis of a campus-wide wireless network;" Proc ACM MobiCom 2002

[8] http://www.rfc-archive.org/getrfc.php?rfc=1157

[9] http://www.net-snmp.org

[10] http://www.net-snmp.org/wiki/index.php/Containers#Introduction

[11] http://openhpi.sourceforge.net/subagent-manual/c53.html

[12] http://www.linuxjournal.com/article/5616

[13] http://www.linuxvirtualserver.org/docs/scheduling.html

[14] http://www.jung.sourceforge.net

[15] http://www.swixml.org/

[16] http://www.sqlite.org/

[17] http://www.zentus.com/sqlitejdbc/

[18] http://en.wikipedia.org/wiki/Simple_Network_Management_Protocol

[19] http://en.wikipedia.org/wiki/Management_information_base

[20] http://en.wikipedia.org/wiki/ASN.1

[21] http://en.wikipedia.org/wiki/AgentX

[22] www.ecsl.cs.sunysb.edu/tr/TR214.pdf

[23] A Adya, P. Bahl, R. Chandra, L. Qiu; "Architecture and Techniques for Diagnosing Faults in IEEE 802.11 Infrastructure Networks;" Proceedings of ACM MobiCom 2004.

[24] IEEE P802.11 – TASK GROUP K. http://grouper.ieee.org/groups/802/11/Reports/tgk_update.htm

[25] http://www.kismetwireless.net/documentation.shtml [12th question]

[26] A Balachandran, G. M. Voelker, P. Bahl and P.V.Rangan; "Characterizing User Behavior and Network Performance in a Public Wireless LAN;" Proceedings of ACM SIGMETRICS, 2002.

[27] D. Kotz and K. Essien. "Analysis of a Campus-wide Wireless Network;" Proceedings of ACM Mobicom, 2002.

[28] A.P. Jardosh, K. N. Ramachandran, K.C. Almeroth, and E. M. Belding-Royer. "Understanding Congestion in IEEE 802.11b Wireless Network," Proceedings of ACM IMC, 2005

[29] R. Mahajan, M. Rodrio, D. Wetherall, and J. Zhorijan; "Analyzing the MAC-level Behavior of Wireless Networks in the Wild;" Proceedings of ACM SIGCOMM, 2006.

[30] J. Yeo, M. Youssef, and A. Agrawala; "A Framework for Wireless LAN Monitoring and its Applications;" Proceedings of ACM WiSe, 2004.

[31] P. Bahl, J. Padhye, L. Ravindranath, M. Singh, A. Wolman, and B. Zill. Dair; "A framework for managing enterprise wireless networks using desktop infrastructure;" Proceedings of the Fourth Workshop on Hot Topics in Networking (HotNets), Nov 2005.

[32] Y. C. Cheng, J. Bellardo, P. Benko, A.C. Snoeren, G. M. Voelker, and S.Savage; "Jigsaw: Solving the puzzle of enterprise 802.11 analysis;" Proceedings of the ACM SIGCOMM Conference, Pisa, Italy, Sept. 2006.

[33] Aditya Dhananjay and Lu Ruan; "PigWin: Meaningful load estimation in IEEE 802.11 Based Wireless LANs"

[34] Ioannis Papanikos and Michael Logothetis; "A study on Dynamic Load Balance for IEEE 802.11b Wireless LAN;" Proceedings of COMCON, 2001

[35] Shiann-Tsong Shue and Chih-Chiang Wu; "Dynamic Load Balance Algorithm (DLBA) for IEEE 802.11 Wireless LAN. Tamikang Journal of Science and Engineering, Vol. 2, No. 1pp. 45-52 (1999)

[36] Balazinska, M and Castro, P. 2003. "Characterizing Mobility and Network Usage in a Corportae Wireless Local-area Network;" Proceedings of the 1st International Conference on Mobile Systems, Applications and Services (San Francisco, California, May 05-08, 2003). MobiSys, '03.

[37] Mathieu Lacage, Mohammad Hossein Manshaei, and Thierry Turletti. "IEEE 802.11 Rate Adaptation: A Practical Approach" MSWiM04, October 2006, Venezia, Italy.

[38] http://www.opennms.org/wiki/Main_Page

[39] Chen, Deng and Varshney, "Protecting Wireless Networks against a Denial of Service Attack Based on Virtual Jamming"

[40] Guo and Chiueh; "Sequence Number-Based MAC address spoof detection"

[41] WEPWedgie. http://sourceforge.net/projects/wepwedgie/

[42] http://www.wi-fiplanet.com/tutorials/article.php/1457211

[43] Guo and Chiueh; "Scalable and Robust WLAN connectivity using Access Point Array"

[44] Wu and Chiueh; "Passive and Accurate Traffic Load Estimation Algorithm"