WiCroft: Crowdsourced Approach for Performance Measurements and Scalability Testing in Dense environment

M.Tech Project Report

Submitted in partial fulfillment of the requirements
for the degree of
Master of Technology
by

Ratheesh K V
Roll No : 153050057

under the guidance of

Prof. Kameswari Chebrolu

Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Mumbai
June 18, 2017

Abstract

Smartphones are common nowadays. Smartphone based learning like watching video tutorials, giving an online quiz, referring to study materials etc. helps to improve the way of learning compared to the conventional approaches. A classroom with students accessing the WiFi network using their smartphones will create a dense WiFi environment. Dense WiFi networks are increasingly popular nowadys, e.g railway stations, airports, even in shopping malls. In dense setting, a single access point (AP) will serve a number of users. The installation of dense WiFi network is costly and the aim is to provide better WiFi experience.

IIT Bombay provides WiFi services to classes by common institute access points **IITB-Wireless** and **eduroam**. By using the **SAFE** (Smart Authenticated fast exams) Android application for conducting cheat free quiz using IITB-Wireless in a class of 60-80 students, it was observed that the AP is able to support only 30 students. Beyond this number, clients are experiencing WiFi disconnection issues and increased delay in getting a response to web request and for the associating with AP. The number of students is being supported are very less compared to the total strength of the class. So it is important to understand the underlying reasons behind these WiFi network problem and to find the solutions to scale the number.

Different methods can be employed to measure the client's performance in a dense WiFi network. Deploying static sniffers, AP based approaches, crowdsourcing are common among them. From literature study and from experimental results it was suggested crowdsourced based approaches give more accurate information about client's performance since the performance metric are collected by clients themselves. The **Loadgenerator** system used by SynerG lab of IIT Bombay uses crowdsource-based approach for collecting client side performance measurements. But this application is identified with lots of drawbacks.

We introduce Wicroft: **Wi**Fi based **Cro**wdsourced **F**ramework for **T**esting a more robust Android application for WiFi performance and scalability testing. This project deals with debugging the network related delay and connectivity issues from WiFi packet trace analysis, design and implementation of the WiCroft framework and use-cases.

One of the use cases of Wicroft was to simulate SAFE application by modelling it's traffic pattern. The experiment results show that Wicroft succeeds in giving similar results with SAFE, this shows Wicroft can simulate any application that uses WiFi service. This results really helped to perform scale testing using Wicroft. The results from WiFi packet traces captured during a SAFE quiz shows that the applications running on clients smartphones generate around 300 web requests in every 5 minutes and these requests sizes range from 40 bytes to 1.5 Kilobytes. These frames occupy 90% of total TCP traffic airtime in the channel and contribute towards the contention.

By adding a firewall to block all non-SAFE uplink TCP traffic brought down all the downlink non-SAFE TCP traffic and uplink reduces to 63.4%. The DHCP delay in getting IP address dominate in the connectivity delay faced by clients. The clients connected to an AP in bridged mode is appeared to take more delay than towards an AP in NAT mode.

Instructing students to install the Wicroft app for conducting scalability experiment, analysing non-decryptable packets from the IITB-Wireless network(uses secure EAP mode), power management and doze mode feature in the higher version of Android devices affected the smooth working of Wicroft app are the major challenges faced during this project.

Acknowledgement

My sincere thanks to **Prof.Kameswari Chebrolu** and **Prof.Bhaskar Raman** for their guidance and supervision which served to improve each step of the project. Thanks to **Swinky Mann**, my project partner, for extended discussion and support on this project. Furthermore, I would like to thank the CSE office staffs for their support in setting up the testbed for this project, students of CS224 Computer Networks and SynerG lab for using their smartphone to install Wicroft app and coorporating with us in conducting experiments.

Contents

List of Tables

List of Figures

1 Introduction

A wireless network deployment with one or more WiFi access points(AP) serving more number of clients forms a dense WiFi network. With increased popularity and use of smartphones, dense WiFi deployments are common nowadays, e.g railway stations, airports, large classrooms, shopping malls etc.

Educational institutions use student's smartphone as a platform for learning purpose. There are applications running on these devices to watch videos, giving online quizzes to improve the learning process. IIT Bombay's SAFE Android application used by the various department to conduct cheat free, auto-graded online quizzes. The institute provides WiFi service to all classrooms and research labs. The common SSIDs of the access point(AP) found in these networks are IITB-Wireless and eduroam.

A classroom with student accessing WiFi service for giving a SAFE quiz form a dense WiFi network. From past experience it was observed was, the IITB-Wireless AP can support only 30 students, beyond this number users are experiencing increased delay in associating with AP and getting a response to the web request, WiFi disconnection issues etc. Many class strengths are more than 30 students. So it was essential to debug these issues for finding reason and suggesting the solution to scale the number of users that can be supported.

A crowdsource-based approach always gives more accurate information related to client-side performance in dense environment. The Loadgenerator Android application developed by SynerG lab was a helpful in measuring client performance in dense WiFi network using crowdsourcing also helps to perform scale testing to identifying network performance with varying number of users. Issues like, unable to use clients in the NATed environment with this application for crowdsourcing, incompatibility with recent Android versions, difficulty in identifying active users etc. are identified from the previous usage of Loadgenerator. Details of Loadgenerator is mentioned in section 2.

The first challenge was to design a more robust architecture for crowdsourcing. The Wicroft framework was designed to solve the drawbacks of Loadgenerator. The deployment and experiments using Wicroft prove it's reliability and robustness. The framework includes an Android application and a server to monitor the clients and to conduct experiments. The architecture details of the framework are described in section 3. The Wicroft was designed to simulate any application by generating similar network traffic. In the initial phase of the project, we made a SAFE traffic model and given to Wicroft app to crowdsource the traffic generation to simulate an actual SAFE quiz environment. The response time calculated by Wicroft clients gave similar results with that of SAFE.

The second challenge was analysing WiFi packet traces from IITB-Wireless network since the AP uses EAP security(and we have no control) the packets cannot be decrypted above the mac layer, so it was unable to figure out the reasons behind network related issues mentioned earlier. An enterprise AP from vendor Airtight, another IITB-Wireless model AP from vendor Aruba in WEP mode was used to conduct SAFE quiz to debug the network related issues. The network related issues faced during a SAFE experiment using

IITB-Wireless is debugged by analysing the sniffed WiFi traffic trace. The trace analysis gives insight into the actual reason behind reduced client performance, the analysis details are described in section 4. The administrator of WiFi network will be interested in about how access point features like adding a firewall rule, enabling dynamic channel selection option impacts on client WiFi experience ?. These features are also analysed in this project.

The third challenge was using the Wicroft application on various Android smartphone models. Wicroft is basically a background running app, some models enabled with power saving mode was observed to disturb the proper functioning of the app, like not passing WiFi manager instance to background service and that make the server unreachable and this client cannot be used for the experiment. Running the app in the foreground during experiment solved this problem. Android versions of 6 and above provide the **doze** feature for optimising the battery usage. The smartphone dozes after some time when phone become idle since the WiFi will be turned off during this time running the app in the foreground will not solve this problem. Tackling doze problem is made as for future improvement to this app.

Video lectures are being part of flipped classroom-based models. From an app developer perspective, it will be interesting to explore different implementations to stream videos on Android applications. In a dense environment, many people are trying to utilise the WiFi service, there is increase channel contention, so a better video streaming method needs to deploy on apps. The Wicroft application supports DASH-based video streaming and a normal progressive download approach. Scalability experiments are conducted in a dense environment to compare these two design choices.

1.1 Problem statement

Our MTP deals with the finding a better solution to above mentioned problems in dense settings by the following approaches,

- Design and implement a WiFi based crowdsourced Framework to perform,
  - Client-side WiFi performance measurements
  - Simulate any application be generating similar network traffic
  - Debugging WiFi disconnection and performance issues
  - To conduct scalability testing and to analyse the WiFi performance with varying the number of users
- Identifying the underlying reasons behind connectivity and performance issues in a dense WiFi setting
- Finding the contribution of DHCP delay in connectivity issues in dense setting
- Compare the performance of a DASH based video download against a progressive download in a dense setting

- Identifying how Accesspoint configurations like dynamic channel selection and firewall rules affect the client performance
- Find the limit on the number of client associations possible with Aruba(IITB-Wireless) and a low cost Microtek(OpenWrt based) accesspoint.

1.2 Work done

This project is done in two stages in a group of two members. The project are equally divided into two parts and each of us contributed to their best towards the goal of the project, the work done in each stage and division of work is as follows,

1.2.1 Work done in stage-1

The work done in the first stage of MTP are listed below. The division of work done is given below and in Figure 1.

1. Analysed the sniffed WiFi packet traces during a SAFE based quiz in CS224m: Computer Network class of 50 students
    - Part-1 : Analysed the WiFi traces for identifying the channel utilisation by different 802.11 frames. Compared the channel utilisation by frames between with Firewall vs No-Firewall scenario. Calculated the DHCP delay in obtaining IP address to the client associated with AP in NATed mode.
    - Part-2 : Identified the client contribution towards the management traffic in the channel. Compared the WiFi channel utilisation with static versus dynamic channel selection modes an AP.
2. Designed and implemented a WiFi based crowdsourced framework for testing
    - Part-1 : Designed and implemented a server for the crowdsourcing system. Server facilitates connecting to clients and conducting the experiments to crowdsource network traffic generation. A dummy SAFE server also modelled to simulate the SAFE quiz environment for debugging network issues and for scalability test.
    - Part-2 : Designed and implemented an android application for WiFi crowdsourcing. The features include like it is a background app, client initiated and maintained TCP connection to the server, change access point on demand etc.
3. Setup a testbed setting and conducted WiFi scalability experiments
    - Part-1 : Experiments are conducted for calculating the DHCP delay in obtaining IP address to the clients. The delay is compared between NAT vs Bridged mode of APs. Also identified the maximum number of client association possible with IITB-Wireless and Microtek APs.

- Part-2 : Created probabilistic models for SAFE network traffic and background traffic generated by other application running on clients smartphones, to simulate an actual SAFE based quiz. Experiments are conducted using this traffic model for analysing the performance received by the clients and finding how many students can give the SAFE quiz for a particular network environment.

  A testbed setting is created by using the following,

  - 80 Vidyut laptops installed with Android 4.4 KitKat operating system. These devices were installed with Wicroft application
  - Two access points were used : Aruba(IITB-Wireless model) and Microtek
  - Server of the crowdsourcing system
  - A laptop configured with monitor mode to sniff packets on a particular channel

The part-2 of above three phases of projects are descibed in report [3], and part-1 are descibed in this report.

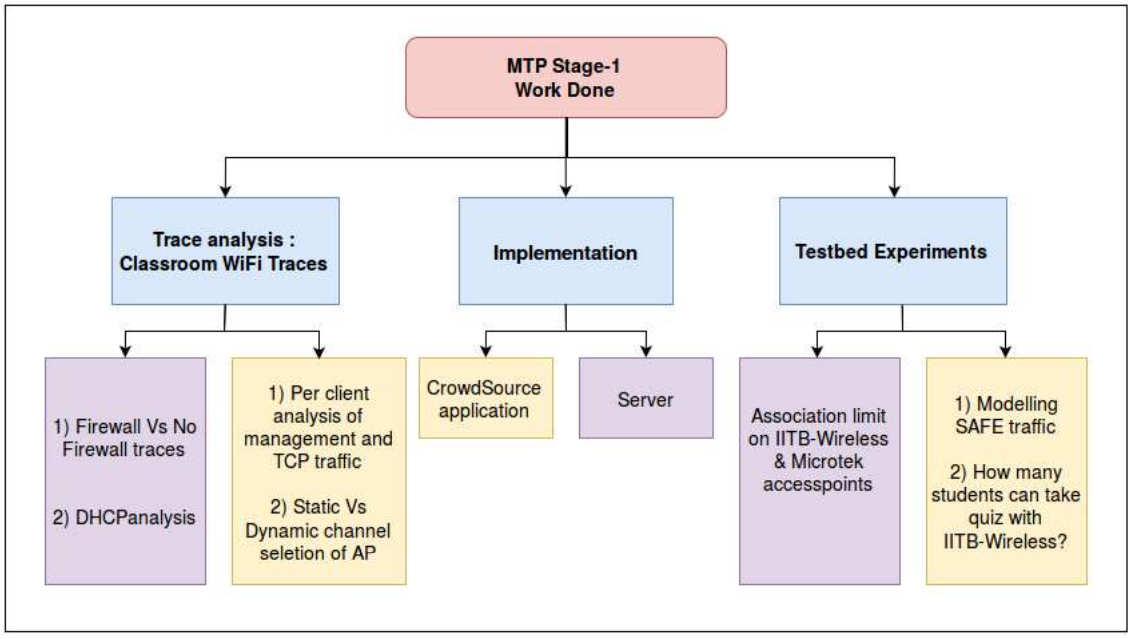

Figure 1: Distribution of work done in MTP stage-1

1.2.2 Work done in stage-2

The work done in second stage of MTP are listed below, also in Figure 2,

In the second stage of MTP, we improved the crowdsourcing system, compared the SAFE with the Wicroft application for mimicking the SAFE traffic, conducted classroom

experiments, video download experiments and analysed the hit ratio of clients connected to Wicroft server.

1. Improved and added features to the Crowdsourcing system

   - Part-1 : The Android application is a background running app, some device with power saving setting enabled disturbed the smooth functioning of the app. The app made to run in foreground to tackle this issue and is helpful for conducting an experiment. Added EXO player to stream DASH and progressive downloading for streaming videos.
   - Part-2 : Sending of control file, start experiment request, fetching log files are also done in batches to reduce the contention. A wake-up timer added to instruct the Wicroft app to run in the foreground.

2. Phase 2 : Wicroft versus SAFE quiz response time comparison

   Wicroft is provided with SAFE modelled control file to crowdsource the traffic generation. A dummy SAFE server also modelled to mimic the actual behaviour of the SAFE server. The performance of the Wicroft application to simulate the actual SAFE quiz is analysed.

3. Phase 3 : Conducted classroom experiments with Wicroft application

   Performed scale testing with the Wicroft app. The app is configured to generate simulated SAFE traffic. The response time of client's URL request with varying number of clients is identified and compared with the actual SAFE quiz with same client strength.

4. Phase 4 : Conducted and analysed video download experiment with DASH versus progressive download methods

5. Phase 5 : Analysis of clients connected to the Wicroft server during classroom experiments

   Among the clients connected to the server, it was observed that experiments are able to conduct with only 70-80% clients. The issues with remaining clients from not selecting for the experiment are analysed.

The part-1 of phase 1, phase 4, phase 5 are descibed in report [3], and rest of the phases are descibed in this report.

1.3 Outline of the report

This report is organised in the following way, section 2 deals with the working and drawbacks of current crowdsourcing application. section 3 describe working and architecture of newly designed crowdsourcing application, server architecture and implementation. section

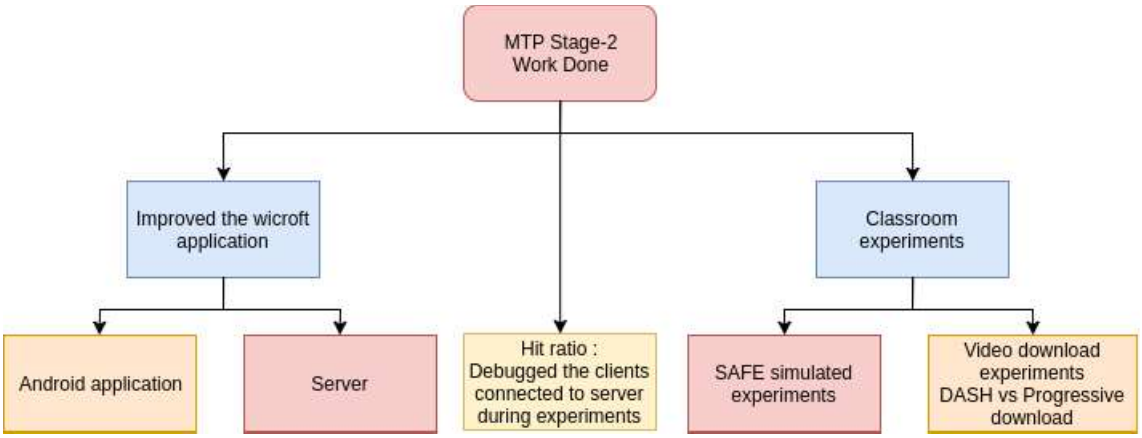

Figure 2: Distribution of work done in MTP stage-2

4 deals with the analysis of WiFi trace from a classroom, section 5 explains about the experiments conducted for finding the number of client associations possible with IITB-Wireless and Microtek access points, section 6 deals with DHCP delay in getting IP address to clients in testbed setup, section 7 compare SAFE and Wicroft, section 8 explains the scale testing conducted using Wicroft framework Finally, section 9 conclude the report. Section 10 suggest the future work of this project, the appendix in section 11 provided with extra details about server GUI, important events and message format in Wicroft client-server communication.

2 Existing Crowdsouce based system

The Loadgenerator application was used to measure the client performance in terms of response time of URL request and RSSI in the WiFi environment. Multiple scalability experiments are conducted using this application, from these experiences a lot of drawbacks were identified. This section tries to explain the working of current crowdsource application and server in brief also identified drawbacks. The detailed architecture and working are described in [6]. Figure 3 shows the crowdsourcing framework setup in a dense WiFi network.

2.1 Working of Loadgenerator in brief

- The server is a Java Servelet application and the client is an Android application.
- The server needs to start on Servelet container like Apache Tomcat web server. Once the server starts, it will wait for clients to connect and register with their IP, port, Mac address, SSID, BSSID details.

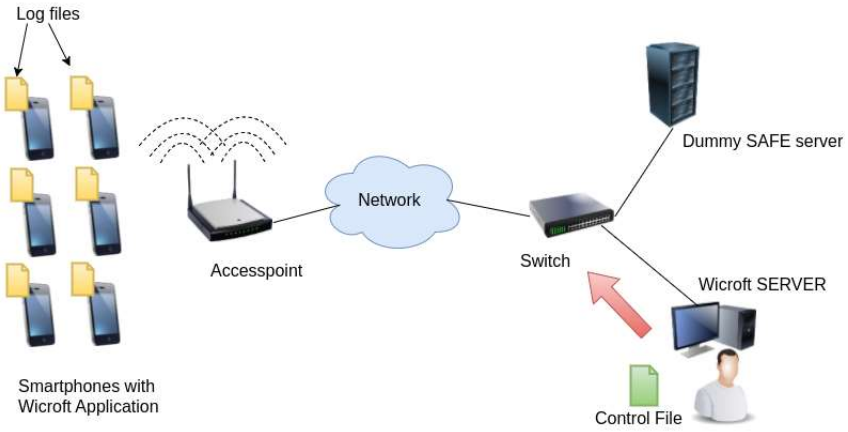

Figure 3: A network setting uses crowdsourcing application and servers

- The user on the client side will manually specify the server IP and port and try to connect to a server using available WiFi service.
- Each client needs to do this procedure for registration, after registration the client closes the connection.
- The server will have IP, port, Mac address details of registered clients. The user in server side is able to view how many clients are registered and registration details? Once there is enough number of clients, the user can proceed to conduct an experiment by selecting the required number of clients and a control file specifying the pattern of network traffic generation. Details about control file are explained in section 3.2.
- The control file contains a list of URLs to request and time of issue of these requests. The clients need to generate these URL requests in the specified time.
- The server will send the control file to selected client by connecting to it based on the IP and port information from registration and closes the connection. i.e the server crowdsources the traffic generation to these clients.
- Once the control file received by the clients, they will acknowledge to the server indicating experiment is starting. The clients can start issuing the URL request at the respective time specified and log the response or any error occurs, response time, RSSI information during the traffic generation.
- The response time of URL request is the performance metrics in the experiment.

- Once the experiment is over, the client will immediately send the log file by HTTP POST to the server. The log file contains the status of each request, the response time of each request, received signal strength at the time of URL request indicating the WiFi performance received.
- From the server side, we can collect the log file and can be analysed to understand the performance at each client.

2.2 Drawbacks of current system

From experiments conducted using Loadgenerator the following drawbacks are identified,

1. Drawbacks with crowdsource client application
    - Since the clients are closing the connection after registration, the server has no way of knowing which among the registered clients are still available with registered IP and port. The clients can move out of the place or WiFi may be turned off. In these cases, the sending of control file can fail and that client cannot be used for an experiment.
    - If the clients are associated with an AP in NAT mode, from the server side it will not be possible to connect to the client with registered IP and port.
    - User intervention is required during the client registration, which may disturb the user and may not cooperate for the experiment.
    - Application needs to run in foreground, there are chances to close this app accidentally by the user at any time and can disturb the experiment
    - The application is not tested on multiple Android versions for checking compatibility. App crashes were observed during installation on different Android versions.
    - Once the experiment is over, the client sends the log file to the server immediately, it can overlap with other clients SAFE traffic request and can affect the response time.
2. Drawbacks with server application
    - The server has no option to identify which all registered clients are still active, for conducting an experiment.
    - For sending control file, the server initiates a connection to clients with registered port and IP address, if the client is behind the NAT, sending control file will fail.
    - No information related to number of students connected to particular AP and how many are active.

- The control files are sent in a single round, it is a part of starting an experiment. Some clients may not receive the file due to contention on the channel. Sending file to all clients also increase the contention. This leads to lesser utilisation of client devices for the experiment.
- Log files are immediately sent to the server after completion of the experiment. If some client is failed to sent due to congestion, the file will not be received to the server later. So there are chances to obtain lesser number of log files compared to the number of clients actually participated in an experiment.
- If there were cases of server restart due to power failure or some reason, all registered IP, Port details will be no more valid to start an experiment, and all clients need to re-register manually.

2.3 Need for new Architecture

Based on the identified drawbacks, the current architecture is redesigned to meet the following requirements.

1. It should be able to use the client devices are connected to an AP in NAT mode
2. There may be required to conduct multiple experiments, so user intervention needs to be minimised by avoiding manually connecting to the server.
3. Log file sending should not overlap with existing experiment because it leads to increase the contention at the access point.
4. The clients need to indicate liveness so that the server can select the active clients for an experiment.
5. It may be required to conduct the performance test on a particular access point, there should some functionality at the server to request the clients to connect to specific AP programmatically, rather than instructing a large number of clients manually.
6. The server needs to have the functionality for identifying and selecting active clients for experiments.

3 Crowdsoucing application new Architecture

The architecture and working of newly designed crowdsource framework is explained here. Figure 4 shows the communication between client and server in this system.

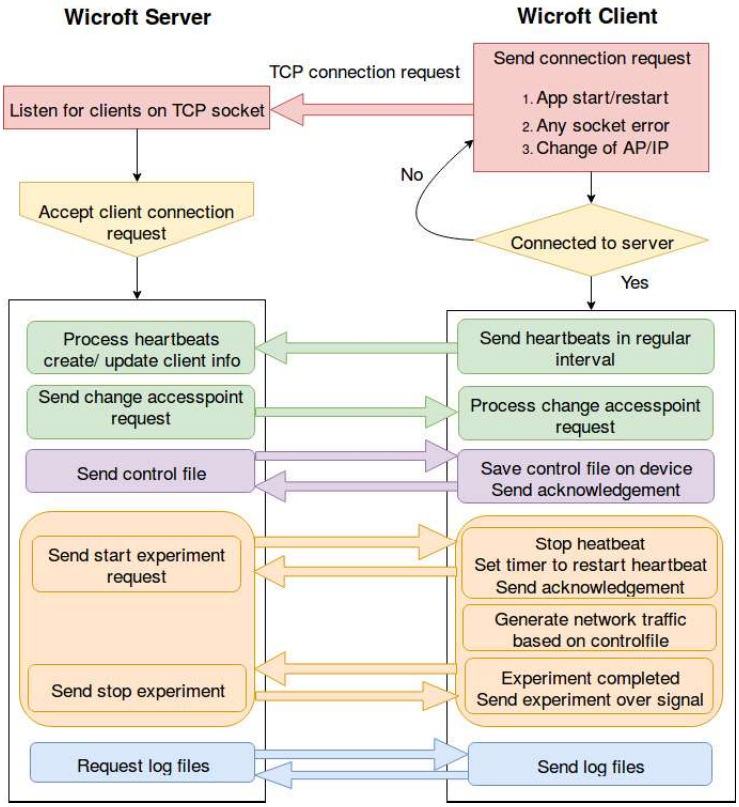

Figure 4: Crowdsouce application Architecture, message exchange between client and server

3.1 Working of the Crowdsoucing System

- In the new design, the server is a Java Servelet application and client is an Android application.
- The server needs to start on a Servelet container like Apache Tomcat web server.
- The server will start with creating a TCP socket and listen for clients to connect. Each connected clients will be handled by Java thread.
- The Wicroft app needs to be installed on the smartphone, it is a background application which will run with minimum user intervention (requirement 2) and tries to initiate and maintain a TCP connection to the server with provided server IP and port. This approach solves the NAT issue mentioned earlier (requirement 1). Now we can make use of clients running behind a NAT.
- The client application tries to keep the TCP connection active. There in only one TCP connection per client to server exists. All the communication between client and server is served by this active TCP connection. The client will send heartbeat

messages at regular intervals (every minute) to the server for indicating its presence (requirement 4).

- The heartbeat message contains IP, Port, Mac address, RSSI, SSID, BSSID, information about nearest access points etc. The server will update the information about client based on the data available in heartbeat. Table 1 describes the fields of the heartbeat message.

- From server dashboard, it is easy to understand the number of clients connected and how it is distributed across access points, how many clients are active in sending heartbeats and how many are passive (requirement 6), this information is updated with every heartbeat from client.

- Sometimes it may be required to conduct scalability testing on a particular access point, WiCroft system support the feature to programmatically connect to a specific AP by providing it's SSID, security and authentication details (requirement 5).

- Once there is enough number of clients connected to server, an experiment can be conducted by crowdsourcing the network traffic generation to these clients. A file called as the control file, that specifies the pattern of URL request need to generate along with the request time and size. The URL pattern in control file can be a model of any application traffic like SAFE. The control files can send to selected clients in batches to reduce the WiFi contention. Manual retry is also added to ensure to utilise all selected clients to receive the file. The clients will acknowledge for successfully receiving the control file.

- An experiment can be started by selecting those clients with a given control file. The start experiment request will be sent in batches, and with a single automatic retransmission to that client who didn't receive or acknowledge for first round request. Since the heartbeat traffic is not part of the traffic being simulated by Wicroft system, the client will temporarily stop the heartbeat during the experiment.

- The Wicroft app will log the status of URL requests like the success or failure, the reason for failure, response time of requests etc. Once all URL requests are done (success of failure) the client inform the server by an experiment over message.

- Once the experiment over, the user from server side can stop the experiment so that the client can stop any running experiment and start sending the heartbeat.

- The experiment log files can be fetched from server interface by requesting to the clients in batches. Since the log files are of size 900 bytes to 1 Kb. There is chance to increase contention at the AP during this transfer. So request to log file in batches will result in sending the log files also in batches and avoid losing the log files. The client will send all pending log file during this transfer. The process of fetching log files is made separate from being part of an experiment, this also avoids overlapping with experiment traffic (requirement 3).

- Once experiment log files are received, the user can analyse it for identifying the client-side performance during the experiment.

The new architecture for crowdsourcing system meet all the requirement and also have the additional feature to provide an efficient approach for client-side performance measurement in a dense setting.

3.2 Control file

The control file is one of the major components of a crowdsoucing system. It specifies the URL traffic need to be generated. It can be a model of any application traffic, like SAFE. Each entry of the file a specific format. Figure 5 shows one sample SAFE modelled traffic control file.

```
POST/GET url_request_time WEBVIEW URL size_of_request url_dependency
or
DASH/MP4 url_request_time EXO URL size_of_request url_dependency
```

The url_dependency equals to zero indicate this URL request is independent of previous URLs and this request can be generated after url_request_time from receiving the start experiment request at the client. A non-zero value indicates the URL request number on which this URL request depended on. i.e this URL request will be issued after url_request_time from completion of it's depended URL request. Control files for multiple clients can be specified in a single file with proper separation of 5 stars (∗ ∗ ∗ ∗ ∗). More details on modelling SAFE URL refer [3]

```
POST 60 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=quiz 340 0
POST 23 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=quiz-get 327 1
POST 178 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=partialsubmission 593 1
POST 290 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=add-log 480 1
POST 393 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=submit 586 1
POST 507 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=ldap-auth 341 0
POST 3 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=get-submission 314 6
POST 15 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=get-summary 357 6
*****
POST 69 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=quiz 340 0
POST 24 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=quiz-get 327 1
POST 170 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=partialsubmission 593 1
POST 179 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=submit 586 1
POST 348 WEBVIEW http://10.129.28.176:8080/TestServelet/testservelet?request=add-log 480 1
*****
```

Figure 5: Modelled URL traffic pattern for 5 minutes SAFE quiz for two clients

3.3 Log files from clients

The Wicroft Android application maintain following types of log files,

1. Experiment log file

   This file is associated with a particular experiment. Following information are logged into this file

   - URL request starting time and ending time, that gives the response time of URL request
   - URL request status, success of failure, reason for failure
   - Mac address, IP, Port number, SSID and BSSID of associated AP
   - Total playback time, number of stalls, video buffer time and average bit rate for video download experiments

2. Connection log file

   This file logs the WiFi connection and disconnection information of client device. The time of the event, SSID and BSSID of associated AP is also logged. This information helpful for finding the connection delay in a WiFi environment.

3. Debug log file

   The file log information like runtime exception, information about message sending to and received from a server. That information is helpful for debugging any issues with client devices while using the Wicroft app. This information is used in later part of the project to debug the hit ratio of clients who participated in experiments.

The complete architecture and added features of the client application are described in [3]. The server architecture and the implemented features are described in the following section.

3.4 Server Architecture

1. Start the server and listen for clients to connect

   The server will start by creating a TCP socket for listening to clients to connect. Each connected clients are handled by single Java thread. The server maintains a mapping between the mac address of the client to all details including reference to the thread. So whenever a client reconnects, the existing thread will be killed and a new thread will serve the client.

2. Receive Heartbeats from clients

   Once the client successfully connected to the socket, all communication between server and client will happen by using this TCP connection. The client will continue to send heartbeats at regular intervals (one minute) to indicate it's presence. The

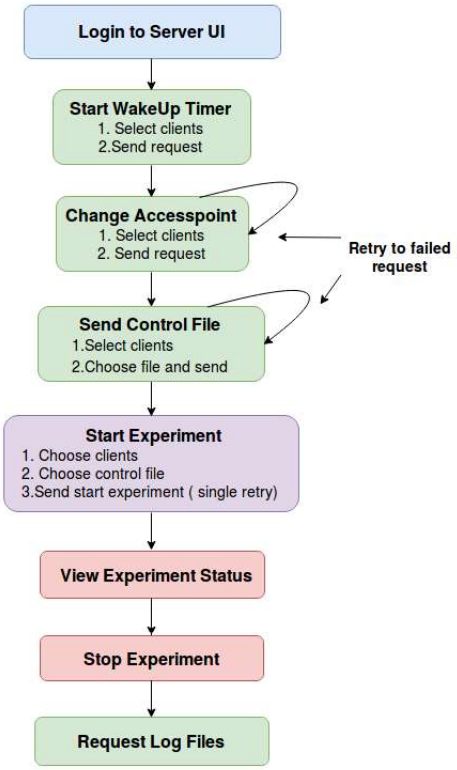

Figure 6: Flow chart for conducting an experiment

heartbeat contains the IP, Port, Mac address, received signal strength (RSSI), SSID and BSSID of associated AP, device hardware information, SSID and BSSID information of nearest APs, shown in Table 1. The server will maintain a map between client's macAddress to these details and updated with each heartbeat.

3. Start a wakeup timer

   The Wicroft is a background running android application. In some models of Android smartphone, it was observed that the WiFi manager instance is not passed to background services, power saving setting disturbs the functioning of background service, in the Android version of 6 and above the device will move to doze mode (with WiFi turned off) after some point of idle time. By moving the app to foreground the first two problems can be solved. A wake timer can be set from the server to instruct the Wicroft app to run in the foreground. Starting a wakeup timer is **recommened**, it will be helpful to maintain all connected clients for the duration of an experiment without loosing.

4. Request to change accesspoint

Table 1: Mac address to client information mapping

Key	Value
Mac Address	IP Address Port number SSID BSSID RSSI(dBm) LinkSpeed(Mbps) NearBSSList[] Processor Speed MemoryInfo Storge Space Number of cores Android OS version Device Model information LastHeatBeatTime Wicroft app version *Thread *Socket

Sometimes it may be required to conduct scale or performance testing based on a particular AP, it is also difficult to instruct the users to connect to a specific AP in a classroom. This module helps to avoid those difficulties, and the user can provide the target APs SSID, security and associated credentials to connect to it. There is no control over BSSID of AP. i.e If there are multiple BSSID with the same SSID, the user cannot force the client to connect to specific BSSID, it is not supported by Android. SSID level connection requests the user can make. The Wicroft app will programmatically try to connect to the specified AP.

5. Send controlfile

   The sending of a control file is done in batches to clients to reduce the WiFi contention. The Wicroft app will acknowledge successful receipt of the file, if not from the server this information are available, and file can be retried. There is an option provided to send a new control file or reuse already existing file.

6. Create and Start an experiment

   The user can create and save an experiment configuration and can be conducted at required time. While starting an experiment the user need to select the correct control file and also clients need to take part in the experiment. Sending start experiment request happens in batches to reduce the contention. There is a single automatic

retransmission to those clients who didn't receive the start experiment request in the first transfer. The start experiment request including retry is sent in such a way that all clients start the experiment synchronously. Once the client started with experiment it stops sending the heartbeat.

7. Stop a running experiment

   From the server side, the user can instruct to stop a running experiment. It may be for starting a new experiment or stopping a wrong experiment. After receiving the stop experiment command, the client will start sending the heartbeat.

8. Request log files

   The clients will maintain a log file, which includes the success of failure status of each URL request. The response time of each request. After completion of the experiment, the user sitting in front of the server can request for these log files. The request can be sent in batches so that the clients will send these logs in a way to reduce the contention. This module helps to fetch all pending log files from the client.

A flow chart describing the one sample flow of conducting experiment is shown in Figure 6. The format of messages exchanged between client and server are explained in Appendix.

3.5 Server implementation

The server with GUI for crowdsourcing application is implemented with a set of features discussed so far are described below. A flow chart for conducting an experiment is shown in Figure 6. A dummy server is also implemented to simulate actual SAFE server to handle the SAFE modelled request from clients.

3.6 Wicroft Server implementation [2]

The server for the Android client application is developed using Java, JSP, and Bootstrap. This application uses MySQL database to store the information related to the experiments, like start and end time, number of clients participated, client information (IP, PORT, Mac address, associated AP info), the status of start experiment request, experiment is finished or not, log file status etc. All architecture features are implemented and the following sections will describe the features implemented at the server along with the screenshots of the user interface.

3.6.1 Dashboard

The dashboard (Figure 7) is the front page where the user will login into, and shows the following information,

- The count of total number, active and passive clients connected to the server

  - Active client : The connected clients from which the server has received the heartbeat within expected heartbeat time window
  - Passive client : The connected clients from which the server has not received the heartbeat within expected heartbeat time window
- Individual client information like mac address, IP address, last heartbeat time etc.
- Accesspoint information, through which the clients are connected to the server
- Server timestamp

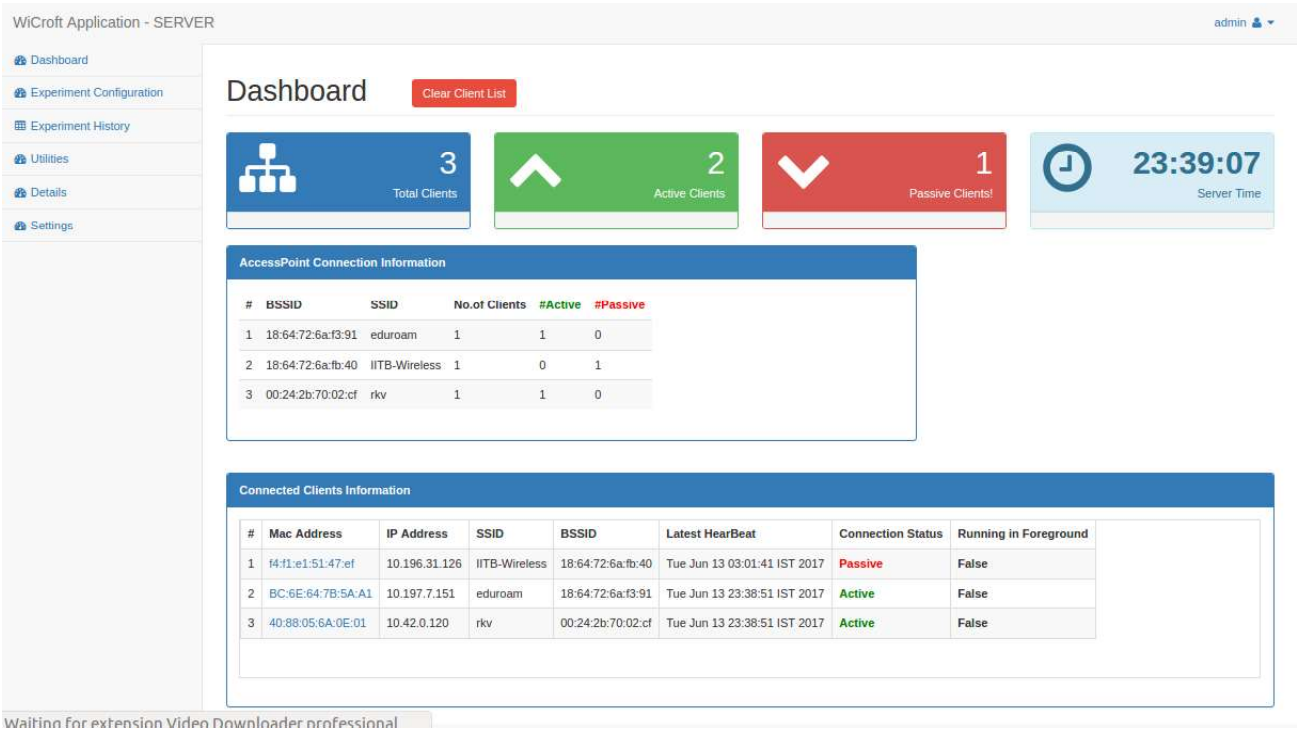

Figure 7: Dashboard of Wicroft server GUI

3.6.2 Request to change Accesspoint of clients

The server have option to send requests to each client for connecting to a specified accesspoint based on the experimenter's interest. This option is helpful to conduct scale test on a specific accesspoint. The Wicroft Android application is programmed to connect to a specific AP based on this request, which saves a lot of time from manually connecting to an accesspoint from each client endpoint. The Figure 8 shows the configuration to send the request, which has option to configure the following,

- SSID of the target accessoint
- Security type, Wicroft Android app currently support the eap, wep, open and wpa-psk security types
- Credentials based on the security type
- Timer : The time at when the client need to try to connect to the target AP, after receiving the request from server. This field can be used to connect a set of clients to a spectific AP at the same time.

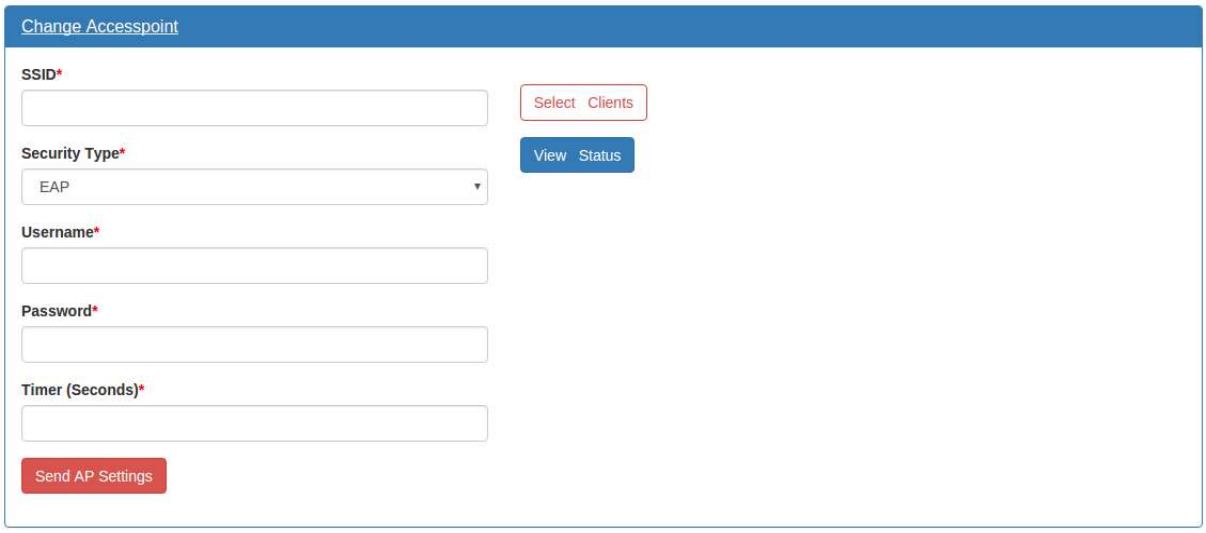

Figure 8: Create request to connect to a specific accesspoint

3.6.3 Select BSSIDs to choose clients

In order to view information about connected clients, the user need to select the BSSIDs from the available BSSID list obtained from clients heartbeat. The clients which are associated only with these selected BSSID lists will be displayed on the dashboard (shown in 7), also for conducting an experiment, sending control file and for fetching log files. The information is updated with every heartbeat. Last heartbeat time is also listed to view which all clients are active during a particular time. The Figure 9, shows option to choose the BSSIDs.

3.6.4 Control file information

The control files are the important components of the crowdsourced system (see section 3.2). The server has the following option to send and view the control files sent to the clients.

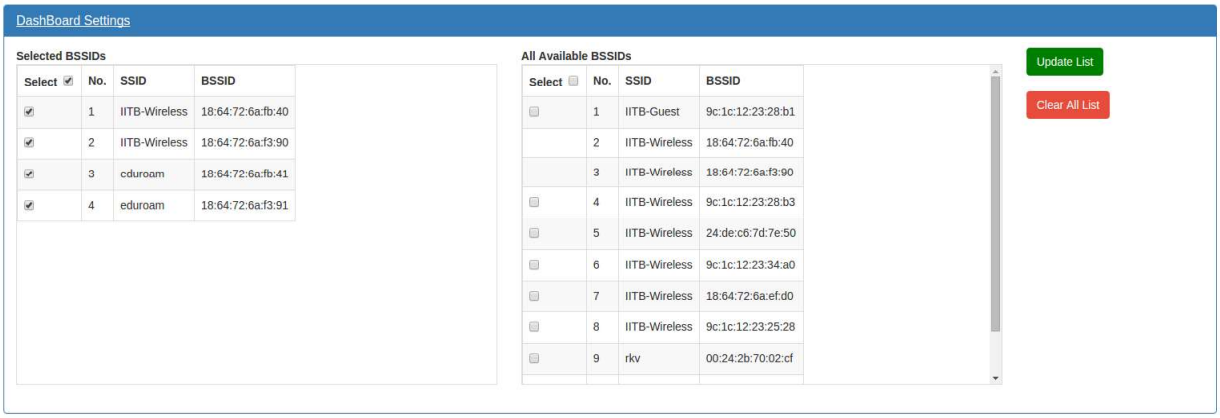

Figure 9: Select accesspoints to filter clients

1. Send new control files to the clients

   The Figure 10, shows the option to choose new control file and list of clients to send the file. An option is provided to send the file to set of clients in rounds, in order to reduce the traffic/congestion on the WiFi channel, due to bulk transfer. This is to maximize the successful reception of the file by the clients. We can specify the number of clients to send the files in a round and the time gap between the rounds.

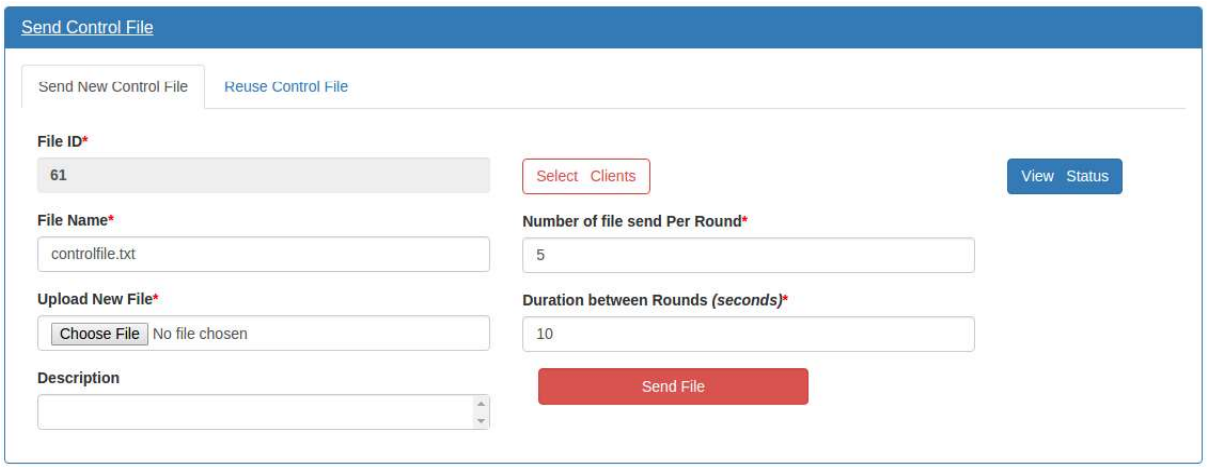

Figure 10: Configuration to send new control file to clients

2. Send previously used control files to the clients

   The Figure 11, shows the option to choose previously used control files and list of clients, to which the file was received earlier. Also an option is provided to send the file to set of clients in rounds, in order to reduce the traffic on the WiFi channel,

due to bulk transfer. This is to maximize the successful reception of the files by the clients. We can specify the number of clients to send the files in a round and the time gap between the rounds.

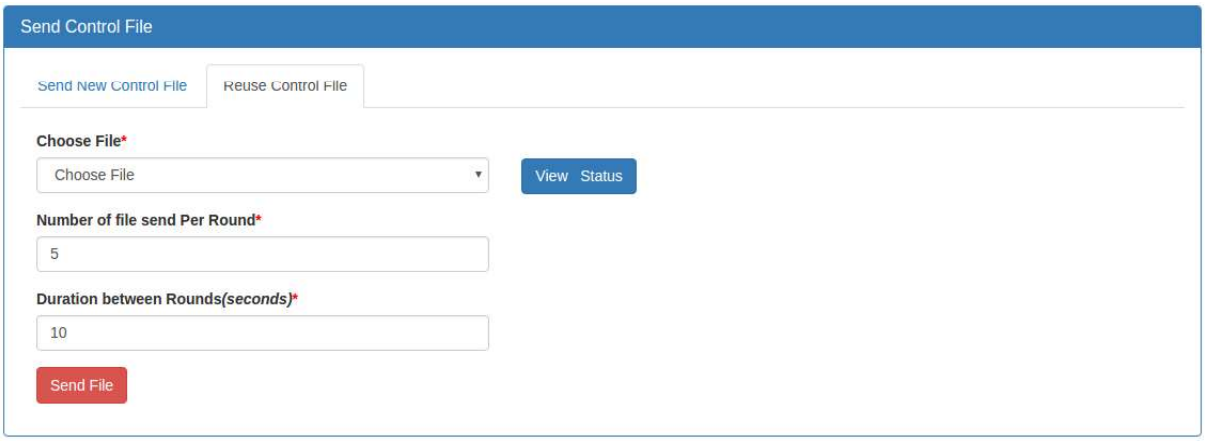

Figure 11: Configuration to reuse older control file

3. To view information about the transferred control files

   The Figure 12, shows information about already sent control files and number and details (link provided) of clients who has received the file successfully.

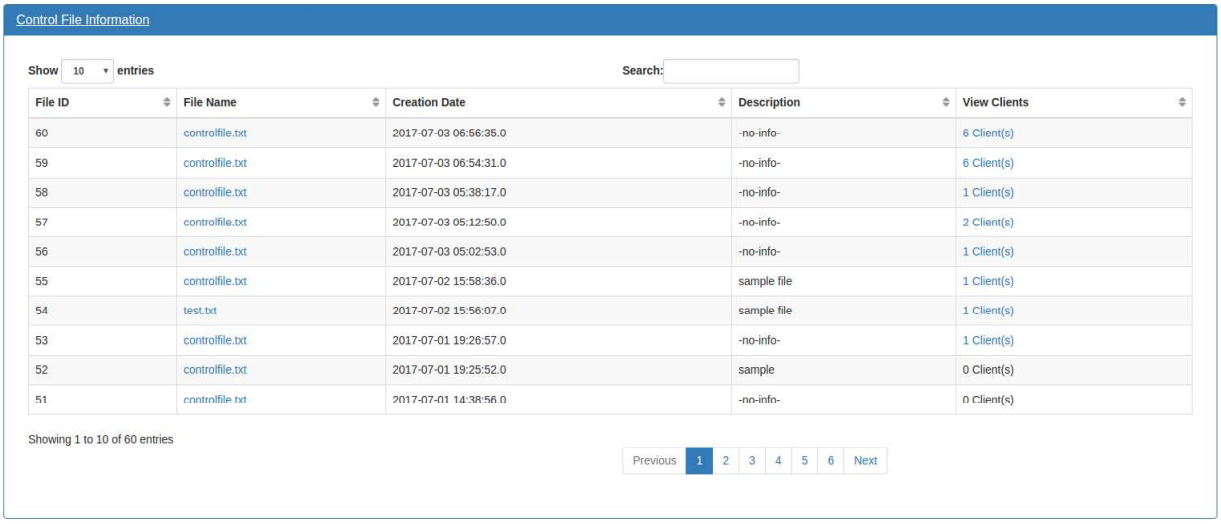

Figure 12: Information about transferred control files

3.6.5 Start a wakeup timer

The Figure 13 shows option to specify the duration of the timer and list of clients to choose for sending the request, refer section 3.4 for more information.

Figure 13: Option to start a wakeup timer for conducting an experiment

3.6.6 Create and save experiment

The Wicroft server provide the options to start with an experiment directly with available clients or to save and start the experiment later on.

1. Start an experiment

   The user can start an experiment by specifying the following information on the user interface (Figure 14)

   - Experiment details
   - Experiment timeout, at which the client stop the experiment, this is the expected experiment duration.
   - Start experiment request sends in rounds : The request per round and time gap between the rounds. This is to maximize the successful reception of the requests by the clients
   - Experiment acknowledgement waiting time : The maximum time at which the server is expected to receive the ACK for the experiment request received from the clients.
   - Experiment schedule time for retry request : If no ACK received from a client, server will have this duration of time to retry the request, the retry to start experiment will be send to automatically to these clients.
   - Control file
   - Select clients in the next step

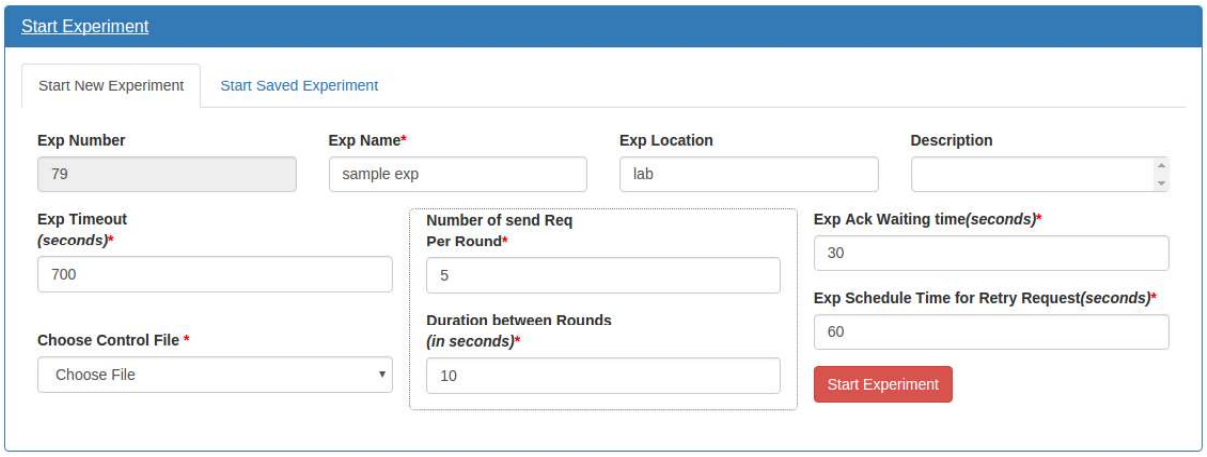

Figure 14: Configurations to start a new experiment

2. Save and load an experiment

   The user can create and save an experiment with or without a control file by providing experiment details. This saved experiment can be load and start whenever required. The Figure 15, shows the options to create and save an experiment. The user can specify the experiment details and optionally choose the control file.
   The Figure 16, shows option to load an already saved experiment, and option to choose the control file will be provided if the experiment was not configured with any. An option will be provided to select the clients once the experiment is loaded/selected.

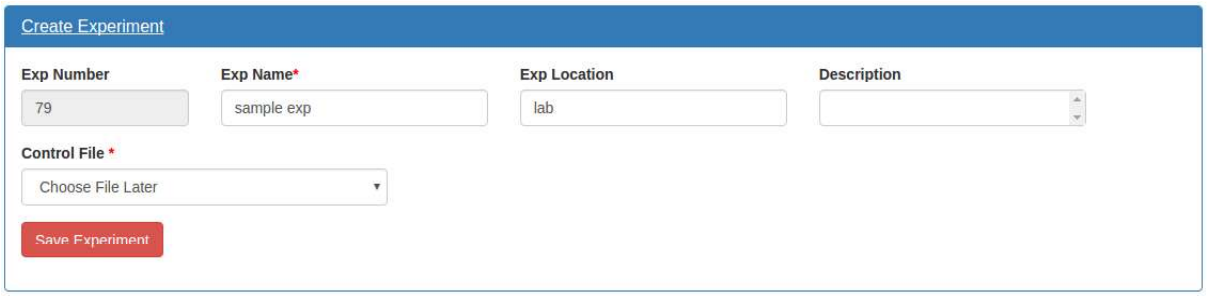

Figure 15: Configuration to create and save an experiment

3.6.7 View Experiment details or history

The Wicroft server maintain the information about the saved and executed experiments. The UI provide the information to view the experiment details (Figure 17)

The following information related to past experiment can be viewed in this module,

- Experiment status : Saved, running and completed experiments

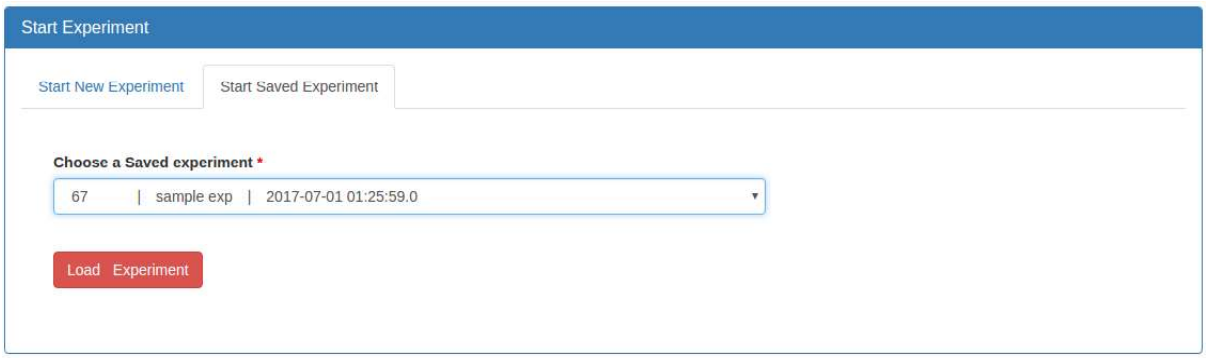

Figure 16: Option to load a saved experiment

- Experiment details like, creation time, location and control file used
- Duration of each experiment by start and end time
- IP, MAC, SSID, BSSID etc. details of clients participated in each experiment

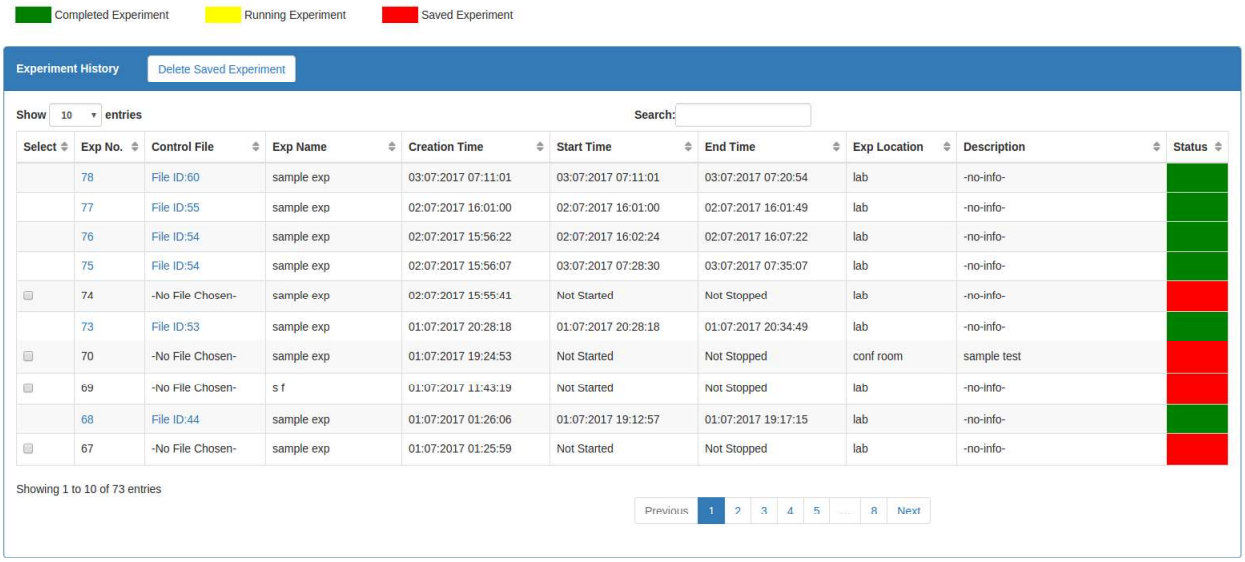

Figure 17: Details of experiments

3.6.8 Request for experiment log files

The Figure 18, show the option to send requests to clients for the experiment logs files, which is not transferred to the server yet. The user can select the clients and the requests

will be send in rounds, such that the clients will send back the log files in rounds, to reduce the WiFi channel traffic and to reduce the number of retries.

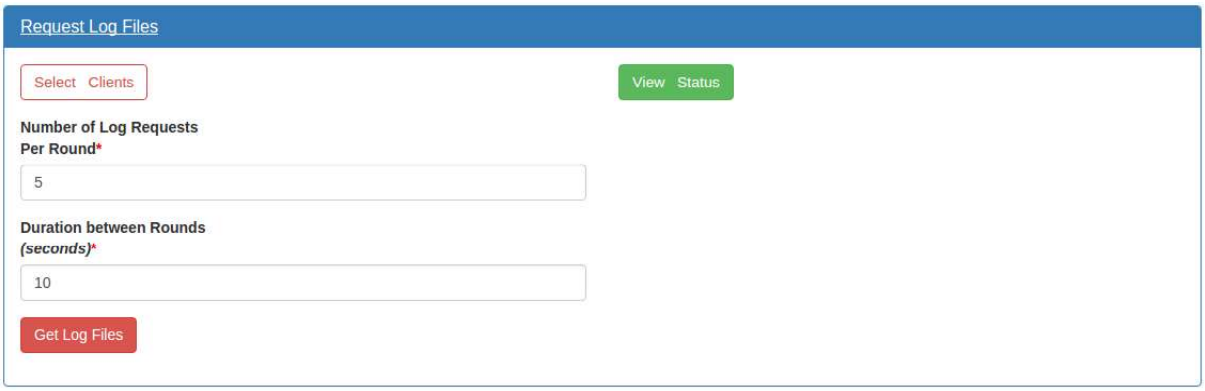

Figure 18: Configuration to send request to collect log files from clients

3.6.9 Wicroft App User informations

The server GUI is also provided with client information (MAC address and Gmail address, device name) who installed Wicroft Android app, and the version of the App they have. The last received heartbeat time also tracked to identify the non-cooperating client devices during an experiment. Figure 19 show the client information.

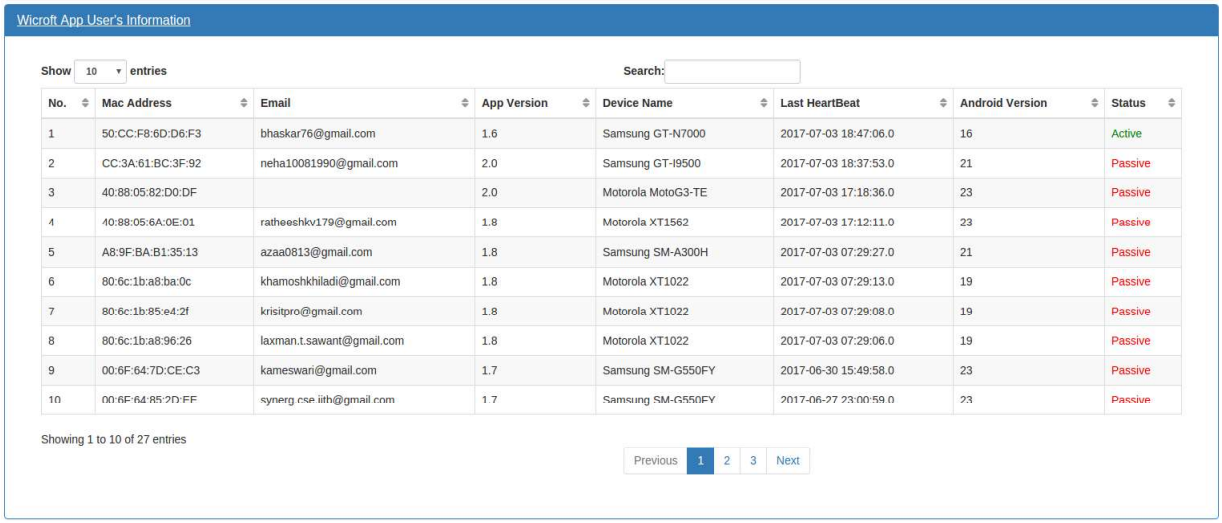

Figure 19: Information about users having installed the Wicroft Android application (Android version is the Android API level)

3.6.10 Create User account

The Wicroft server provides an admin account to user. In addition to that the user can create additional account from this section. Each user can perform all functionalities mentioned earlier. The UI provide the following option to handle the user accounts.

1. Create a new user account

   The user need to provide the new username and password (Figure 20)

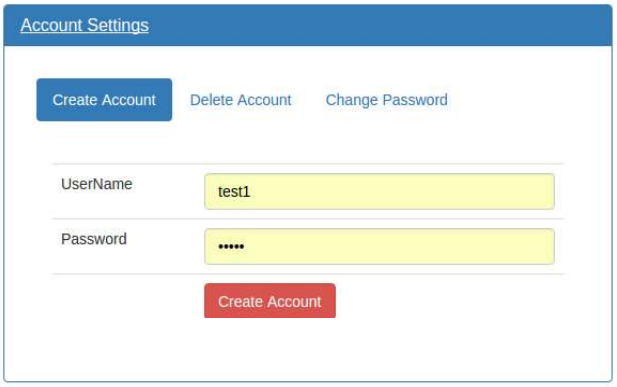

Figure 20: Option to create new user account

2. Delete an user account

   The user need to provide the username and password of the account (Figure 21)

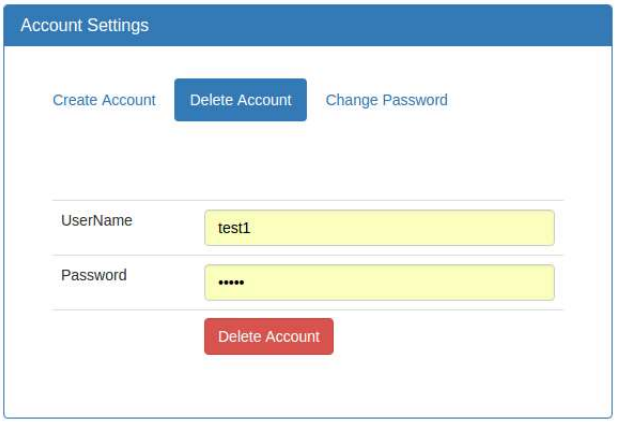

Figure 21: Option to delete a user account

3. Change password for an user account

   The user need to provide the username, current password and new password for password change (Figure 22)

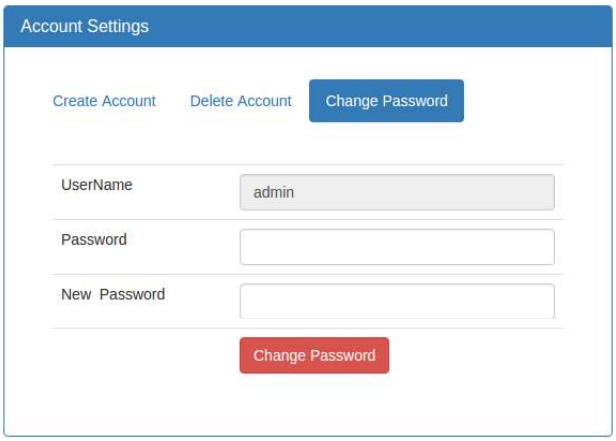

Figure 22: Option to changes password for an user

3.6.11 Send Wicroft Android app update notification

The Figure 23, shows the option to select the clients and send a message to notify the user about the availability of the new version of the Wicroft Android application on the Google PlayStore. The message once received will show a notification on the user's smartphone to inform the user to upgrade the existing app [3]. This option is helpful to notify about the availability of a new version of Wicroft application to the user.

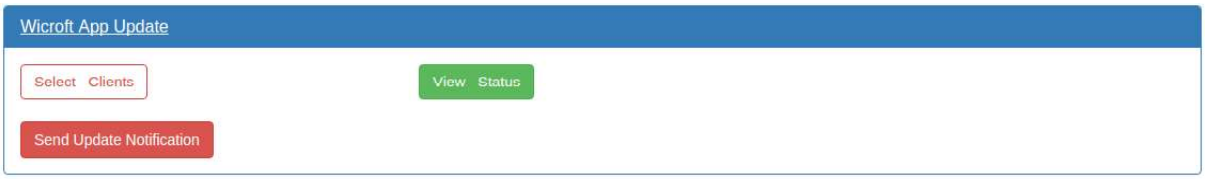

Figure 23: Option to send notification for Android app update to clients

3.7 Dummy SAFE Server implementation [2]

A dummy server is implemented to mimic the actual SAFE server. This server will respond with HTTP 200 OK message to all received HTTP GET/POST requests. Dummy server is implemented using JSP. The HTTP response messages have the following format to avoid additional *favicon.ico* GET request from a client.

```
<!DOCTYPE html>
<html><head>
<meta charset="UTF-8">
<link rel="icon" type="image/x-icon" href="data:image/x-icon">
```

```
</head>
<body></body>
</html>
```

The response time to each request and size of the response is modelled probabilistically based on analysis of actual SAFE server responses. The comparison between actual SAFE experiment delay versus Wicroft experiment delay is described later in this report in Section 7.

4 WiFi Trace Analysis[4][5]

The WiFi packets are captured using a laptop in monitor mode, during an actual 5 minutes SAFE quiz from the same classroom on two different days. An airtight access point in WEP configuration is used to serve the clients. The collected traces are parsed using the below **tshark** command and the required fields are filtered from the trace.

```
tshark -nr pcap_file -o wlan.enable_decryption:TRUE -o ""uat:80211_keys:
""wep",""wep_key"" -Tfields -e field_name1 -e field_name2
```

- **Goal of trace analysis**
  By trace analysis, we are trying to find answers for the following questions,
  Identifying how the channel is utilised during the SAFE quiz?. Which type of frame is occupying a major fraction of airtime? what is the pattern for SAFE traffic? Identifying methods to control irrelevant traffic. How the DHCP delay is contributed towards the connectivity problem faced during associating with AP.

The classification of IEEE 802.11 frames are illustrated in Figure 24, this classification is used to reach the goal of trace analysis. The fraction of airtime of frames is used to describe the channel utilisation in trace analysis.

Note : Bodhitree is the server side of SAFE client application also called as SAFE server

4.1 Experiment setup 1 (No Firewall)

WiFi packets are captured during a 5-minute SAFE quiz from **CS641** Computer Networking class with a strength of 50 students. A single access point(vendor Airtight) with WEP security is used to serve all the clients. The access point is serving the clients in NATed mode. The WiFi traffic in the channel is sniffed using TCP dump. No firewall rules are added to the AP.

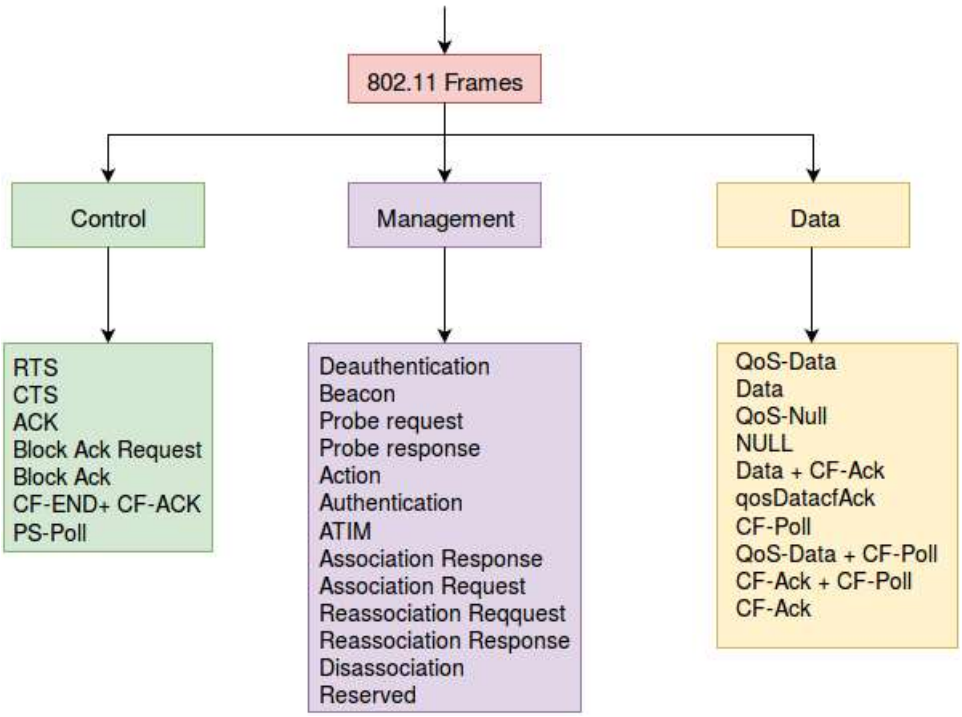

Figure 24: Classification of IEEE 802.11 frames

4.1.1 Analysis of WiFi traffic

The following questions are trying to answer by trace analysis.

1. How is the channel utilised during the quiz?

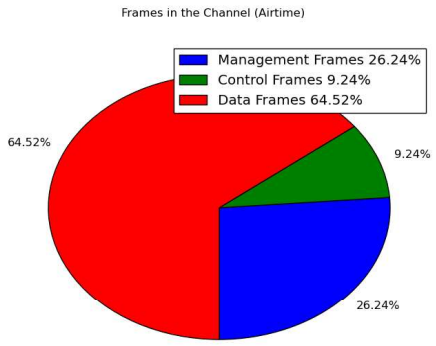

Figure 25: Fraction of airtime by different frames in the channel

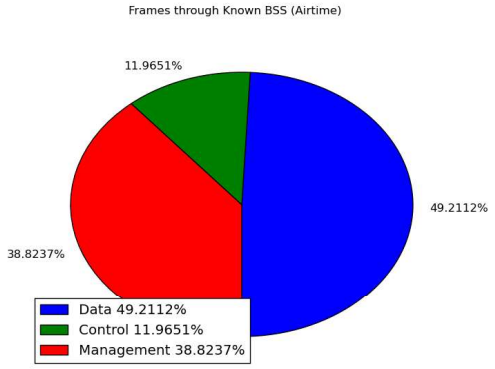

Figure 26: Fraction of airtime by different frames through Airtight AP

- From collected trace, it was observed that the access point used for quiz contributed 43.49% of total sniffed packets, there were two IITB-Wireless and two

IITB-Guest access points were also using the same channel, they contribute around 42.81% of total packets in the trace.

- From figure 25, the data frames occupies major portion 64.52% of airtime during the quiz
- From figure 26, the data frames occupies major portion 49.21% of airtime through our BSS
- Contribution of control frames and management frames are also significant

2. What is the contribution of control frames in the channel?

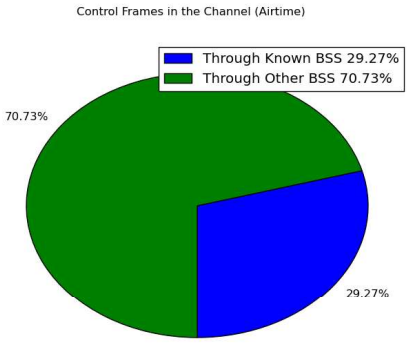

Figure 27: Fraction of airtime by control frames in the channel

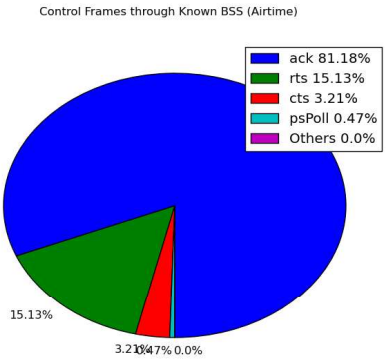

Figure 28: Fraction of airtime by control frames from known BSS

- Since there are multiple access point using the same channel, the control frames associated with our access point is around 29.27% of total airtime (figure 27).
- The major portion of control frames through our BSS is acknowledgement frames(81.18%), they are the unicast acknowledgement to received data frames (figure 28).
- There are 50 clients associated with access point, the RTS frames generated by these devices for sending data contributed to 15.13% of airtime (figure 28).
- Acknowledgement and RTS frames occupies major portion of airtime, CTS frames are occupies 3.21 % of airtime.
- Remaining control frames contribute negligible traffic in the channel.

3. What is the contribution of management frames in the channel?

- The management frames associated with known BSS contribute 33.45% of total management frames in the channel (figure 29)

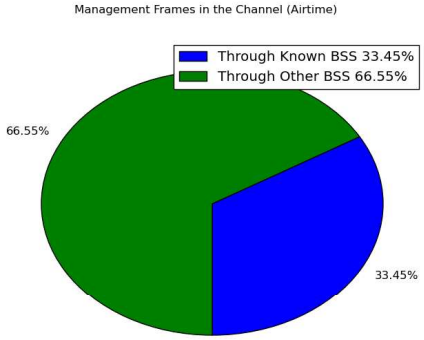

Figure 29: Fraction of airtime by management frames in the channel

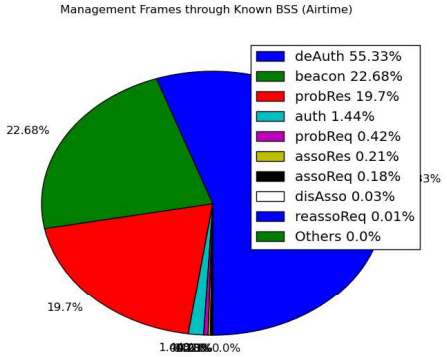

Figure 30: Fraction of airtime by management frames from known BSS

- The deauthentication frames occupies major portion of management frames 55.33%, it was due to configuration problem associated with specific access point. Deauthentication frames are used to terminate the authentication of a station, but from trace analysis it is observed that it does it cause any client disconnection from access point (figure 30).
- Beacons are broadcast frames having default frequency of one in 102.4 milliseconds, they occupies 22.68% of airtime
- Probe responses to client's probe request occupies 19.7% of airtime. The client devices having less signal strength, the probe requests frames might have lost due to interference and they occupies only 0.42% of airtime (figure 30).
- Remaining management frames contribute negligible traffic in the channel.

4. What is the contribution of data frames in the channel?

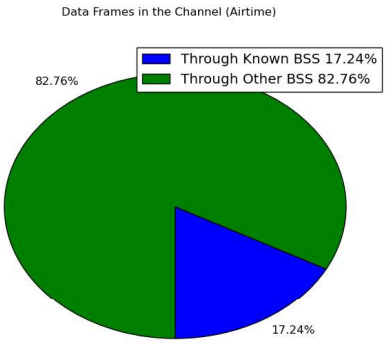

Figure 31: Fraction of airtime by data frames in the channel

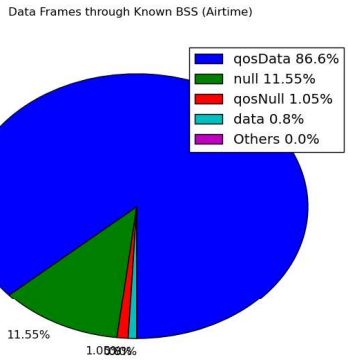

Figure 32: Fraction of airtime by data frames from known BSS

- The data frames through known BSS contribute around 17.23% of airtime in the channel (figure 31).

- The major portion of data frames are QoS versions of data and occupies 86.6%. It include all connection oriented SYN, FIN packets, DHCP, DNS, HTTP, TLS and other TCP, UDP packets with data payload (figure 32).
- The null data frames occupies 11.55% of airtime, these data frames does not carry any data payload. The only purpose is to carry power management bit information from station to access point[1] .
- Rest of types of data frames are negligible in number.

5. Analysis of TCP data frames

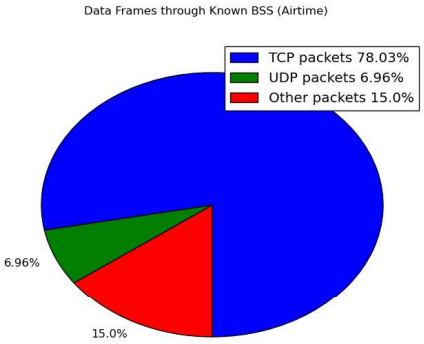

Figure 33: Data frames classified as TCP, UDP and Others associated with known BSS

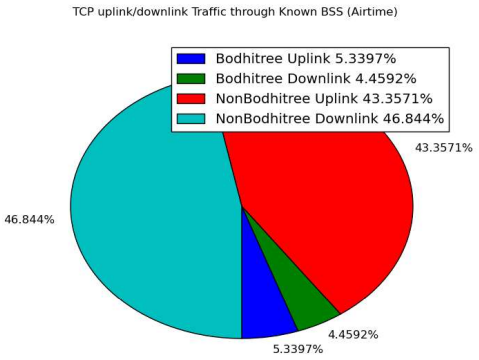

Figure 34: Bodhitree vs Non-Bodhitree TCP traffic associated with known BSS

From Figure 33,

- It is observed that TCP data frames are major component of data frames through known BSS. These TCP frames include SYS, FIN, data payload, TLS, SSL data frames.
- UDP frames contribute 15% of airtime, these include mainly DHCP and DNS packet.
- The frames that are classified under **other** frames are described in the following Table 2.

From Figure 34,

- Other than TCP and UDP data packet, 802.11 null data packet contribute more towards the traffic, these frames are used to carry power management information from stations to access point. For the following cases, a station used to generate 802.11 null data frames to inform the current access point about buffering the frames,
  - Due to power save more of operation of station

Table 2: Data frames other than TCP, UDP through known BSS

Frame Subtype	Fraction of airtime	Packet Type
Null Data	77.01%	802.11
QoS Data	13.78%	ARP, ICMPv6, IGMPv3
Data	2.19%	ARP, ICMPv6
QoS Null data	7.01%	802.11

  - Station tries to scan and associate with other access point

- From Figure 34, it is clear that only 10% of total TCP traffic is contributed towards Bodhitree traffic while using SAFE App, ramaining 90% of traffic are destined towards non-Bodhitree server. The Non-Bodhitree traffic are generated by the background applications running on client's smartphones.
- If we are able to reduce the non-Bodhitree traffic, the channel contention can be reduced.

4.1.2 Conclusion

The channel used by Airtight AP is shared by multiple APs, 43 % sniffed packets are from other APs. The total SAFE traffic is only 10 % of TCP traffic, remaining 90 % TCP traffic are generated by other application running on student's smartphones. This indicate the amount of contention in the channel. So the main reason for contention are the other APs and background applications on smartphones.

By adding firewall rules to access point to drop all outgoing non-Bodhitree TCP packet, the traffic in the channel can be reduced. Since other APs are part of Institute WiFi infrastructure and also use dynamic channel selection. So we don't have control to reduce the other APs traffic. The next quiz is conducted with adding the firewall rule and trace analysis done in the next section.

4.2 Experiment setup - 2 (with Firewall)

From the same **CS641** Computer Networking class with 50 students, WiFi packets are captured for a 5 minutes SAFE quiz. A single access point with WEP security is used to serve all the clients. The AP uses NAT configuration and a firewall rule was added to drop all outgoing non-Bodhitree traffic. The WiFi traffic in access point's channel is sniffed using TCP dump.

4.2.1 Analysis of WiFi traffic

The observations are made in comparison with previous trace analysis with no firewall rules. The following questions are trying to answer by trace analysis.

1. How the channel is utilised during this experiment?

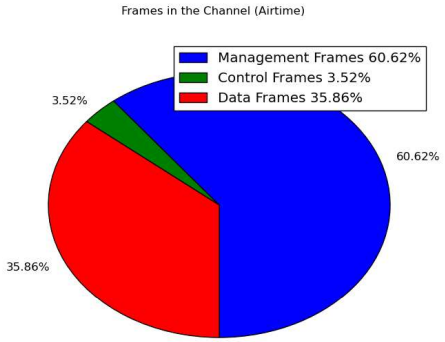

Figure 35: Fraction of airtime by different frames in the channel, with firewall

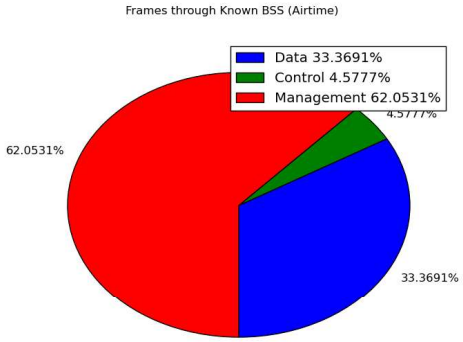

Figure 36: Fraction of airtime by different frames through known BSS, with firewall

- From trace it was observed that 26.12% total packets are contributed by known BSS, the same channel is used by IITB-Wireless and IITB-Guest access point and they contribute around 40% total packets.
- From Figure 35, the fraction of airtime of data frames reduced to 35.86% from 64.52%, with the use of firewall.
- From Figure 36, the fraction of airtime of data frames through our BSS is reduced to 33.36% from 49.21%, with the use of firewall.

2. How the data frames uses the channel during the quiz duration?

   Since the Firewall rule was added to control the non-Bodhitree TCP data frame in the channel, the following comparison made between with amount of data frames with Firewall and without firewall scenarios. (Figure 37, 38, 39 and 40).

4.2.2 Conclusion

This section summarizes how much TCP data traffic reduced by introducing firewall.

Table 3: Data frames through known BSS in the channel, Firewall vs No-Firewall

Parameter	**Without Firewall**	**With Firewall**	**Comments**
Total data frames in the channel	17.24% of channel	10.61% of channel	38.45% reduced
Total data frames through known BSS	49.21% of BSS frames	33.37% of BSS frame	32.18% reduced
Total QoS data frames through known BSS	86.6% of data frame	41.35% of data frame	52.25% reduced
Total TCP data frames through known BSS	78.03% of data frame	28.78% of data frame	63.12% reduced
Total Bodhitree traffic (uplink and downlink)	9.79% of TCP data frame	14.30% of TCP data frame	46% increment in Bodhitree data frames

Table 4: Data frames through known BSS in the channel, Firewall vs No-Firewall

Parameter	**Without Firewall**	**With Firewall**	**Comments**
Total non Bodhitree traffic (uplink)	43.357% of TCP data frame	85.41% of TCP data frame	Due in TCP SYN and it's retranmission
Total Packet size	4318480 bytes	1220754 bytes	71.73% reduction
Total packet count	31537	11519	63.47% reduction
Total non Bodhitree traffic (downlink)	46.84% of TCP data frame	0.2838% of TCP data frame	Small downlink traffic from local WiFi network
Total Packet size	8357416 bytes	5296 bytes	99.93 % reduction
Total packet count	10648	56	99.47% reduction

- The addition of firewall nullifies the non-Bodhitree downlink data frames fraction from 46.844% to zero. The 0.28% downlink traffic contributed by TCP connection reset packet generated by the client in the wireless network in response to TCP connection request from another client in the same network.
- The bodhitree traffic with firewall increased to 14% from 10% in without firewall case.
- The non-Bodhitree uplink traffic airtime is increased from 43.357% to 85.4%.

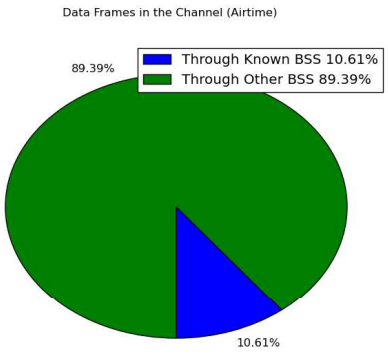

Figure 37: Fraction of airtime by different data frames in the channel, with firewall

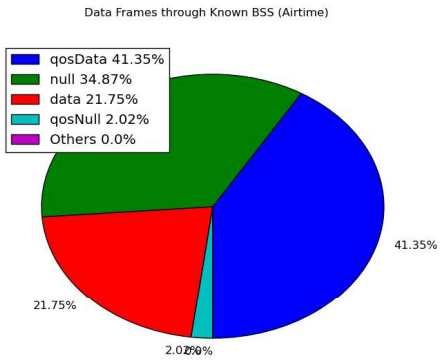

Figure 38: Fraction of airtime by different data frames of known BSS, with firewall

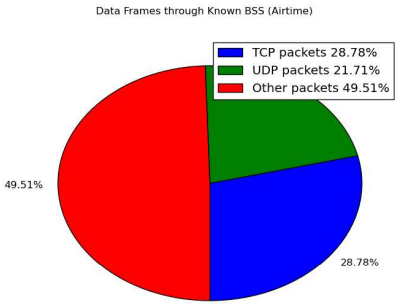

Figure 39: TCP, UDP, other data frames from known BSS, with firewall

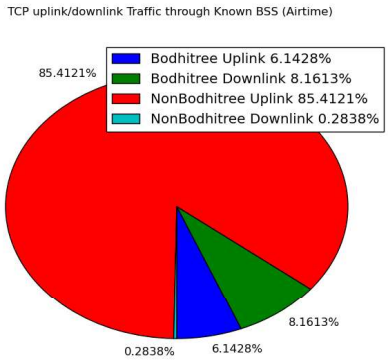

Figure 40: TCP Uplink and Downlink frames from known BSS, with firewall

From traces, it is observed that since firewall is not blocking any DNS packets, clients are able to resolve IP address of non-Bodhitree web requests and trying for TCP connection. TCP connection requests are dropped at the access point, results in SYN retransmission. It was observed from analysis that out of all non-Bodhitree uplink packet generated, 99.88% count of total packet is SYN and of which 59.25% are SYN retransmissions.

- The bodhitree frames are still only 14% of total TCP data frames.
- It is not possible to control 100% of web requests from the client device because it was generated by different background applications running on it. The current reduction in uplink traffic helps to reduce the channel contention
- Figure 41 compare a number of data frames in bytes through Airtight AP in firewall versus no-firewall scenario. The green arrow indicates the percentage of reduction with firewall, values in bracket shows a number of bytes retransmitted. There is 97.42 % reduction in downlink non-SAFE traffic, since all connection

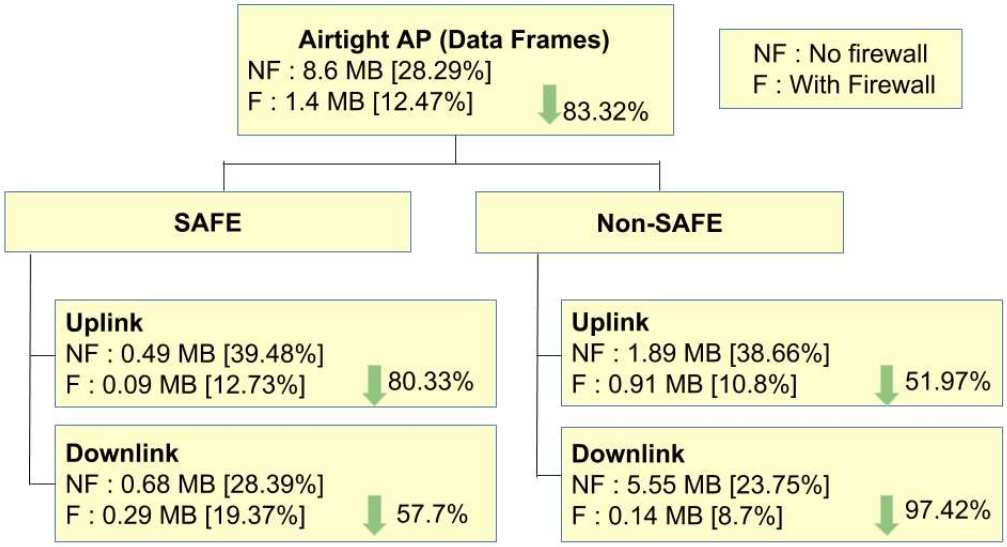

Figure 41: Firewall vs No-Firewall Data frames comparison

request are dropped at the AP. The decrease in SAFE traffic can be explained by the reduction in the transmission percentage. The amount of retransmission is less compared to the no-Firewall case, so the channel contention is reduced. There is 51 % reduction in non-SAFE uplink traffic size, because these traffic contains only SYN and SYN retry and no big size data packet going outside other than towards the SAFE server.

5 Experiments on client association limit of access points

The experiment is conducted for identifying the maximum number of clients can be associated with given access point.

5.1 Experiment setup

This experiment is conducted in the testbed for finding the maximum number of clients can be supported with two different access points, IITB-Wireless (Aruba access point) and Microtek access point. The Vidyut Laptops with Android Kitkat 4.4 the platform is used as clients for the experiment. The crowdsourcing application is pre-installed on these devices.

The experiments are conducted in two scenarios,

1. Simultaneous connection by clients without any background traffic

   By using the feature in crowdsourcing application and server, it possible to programmatically connect a set of clients to specific access point simultaneously. The experiment is conducted without any background traffic.

2. Simultaneous connection by clients with background traffic

   In this scenario, some clients are generating background traffic to create contention at the access point, while remaining clients trying to associate with the same access point.

5.2 Results

1. With IITB-Wireless (Aruba access point)

   In all above two scenarios, the IITB-Wireless is able to associate maximum 64 clients. When more than 64 clients are trying to connect, the association response indicates **Error : Malformed Packet(Exception occurred)** message.

2. With Microtek access point

   All the 80 devices are able to successfully associate with the access point without any problem. We don't know about the maximum limit, but 80 devices are server by this AP.

6 Experiment for finding DHCP delay

From real-time experience, it was observed that client devices face increase delay in associating with IITB-Wireless in the large classroom. IITB-Wireless used bridged mode operation where DHCP server in the network handle IP requests. Since this access point uses EAP security, it is not possible to decrypt the WiFi traces for debugging this issue.

A testbed setting with an access point having the same configuration as IITB-Wireless is used to understand DHCP delay. Using the Wicroft's change AP feature, we instruct a group of clients to connect to the AP Simultaneously. The network traffic is captured

by a sniffer. Since it not possible to decrypt traces, the time difference between WiFi disconnection and WiFi connection information logged by the crowdsourcing application is taken as connection delay. This connection delay includes authentication, association and other Android related delays. From sniffed data, it was observed that association and authentication contributed to a maximum of hundreds of milliseconds. So the majority of time is spent for obtaining IP address, ie DHCP delay.

The same experiment is conducted using another access point with WEP security, with an NATed mode of operation. A comparative study between DHCP delay in NAT vs bridged mode of operation is also done. Finally, DHCP delay is also calculated from real WiFi trace using an access point with WEP security, to compare testbed setting with real network situation.

The aim of this experiment is to understand the contribution of DHCP delay to connectivity issues with an AP

6.1 Testbed Experiment setup

The components of testbed are mentioned earlier. Vidyut laptop devices are used as clients. The DHCP delay is calculated for clients trying to connect simultaneously. Two access points Microtek with NAT and IITB-Wireless in bridged mode of operation is used.

6.1.1 Accesspoint with NAT mode, Table 5

Table 5: DHCP delay, in NAT mode

No. of clients	**Minimum delay (seconds)**	**Average delay (seconds)**	**Maximum delay (seconds)**
20, with no traffic	4	7.5	11
30, with no traffic	3	7.33	12
40, with no traffic	3	6.25	11
50, with no traffic	3	7.2	11
60, with no traffic	3	9.08	32 [only one client]
39 clients generating traffic and 35 trying to connect	3	11	25

- In NAT mode, with increase in the number of clients, there is no significant increase in DHCP delay.
- Only few clients have maximum delay of 32 seconds, and the rest of the clients have delay close to average value.

- Experiment with background traffic, maximum delay for two clients are 25 and 17.5 seconds respectively. The average delay is same as with no background traffic scenario.

6.1.2 Accesspoint with Bridged mode

Table 6: DHCP delay, in BRIDGED mode without background traffic

No. of clients	Minimum delay (seconds)	Average delay (seconds)	Maximum delay (seconds)
20	3	7.88	16
30	3	9.96	18
40	11	13.45	25
50	11	14.22	34
60	6	18.4	86 [two clients]

Experiment without background traffic, Table 6

- In bridged mode, the delay in getting IP address increases with increase in number of clients.
- With 60 clients the average delay obtained is doubled compared with 30 clients, (figure 27).
- Only two clients have maximum delay of 86 seconds, and for rest of them significant increase is not observed.
- Figure 29 shows comparison between connection delay in NAT vs bridged mode

Experiment with background traffic, Table 7

- The DHCP delay is not significantly increased with background traffic for same number of clients, compared with no background traffic scenario.
- Few clients received a significant delay of 88 seconds for obtaining IP address.

Figure 42 compare the DHCP delay with AP in NAT versus bridged mode. Minimum, average and maximum delay values are also shown in the graph.

6.2 DHCP delay from real WiFi trace

From the WiFi traces collected from a classroom of 50 students, the DHCP delay in obtaining IP address to the clients are calculated by the following method,

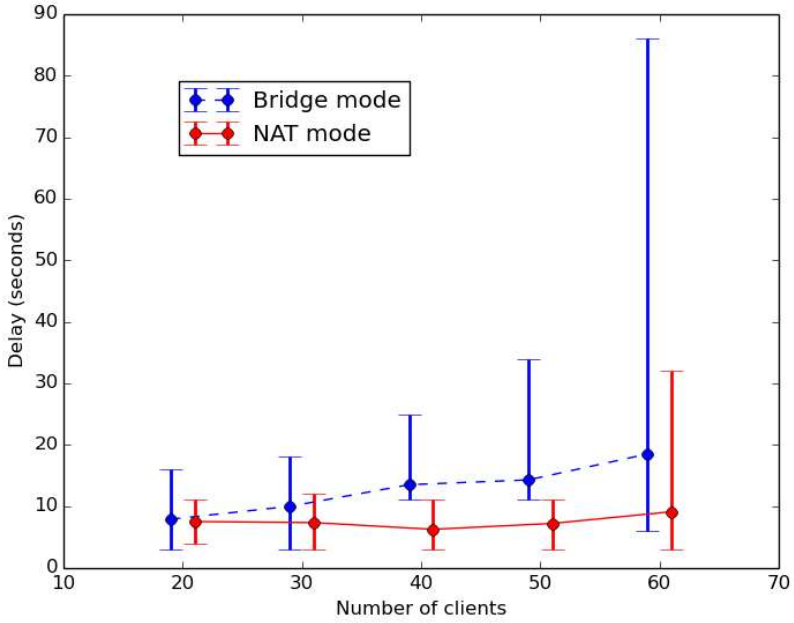

Figure 42: DHCP delay in NAT vs Bridged mode, without background traffic

Table 7: DHCP delay, in BRIDGED mode with background traffic

No. of clients	**Minimum delay (seconds)**	**Average delay (seconds)**	**Maximum delay (seconds)**
44, with background traffic	7	12.878	31
53, with background traffic	11	14.7143	34
10 connected and 50 trying to connect with 3 times background traffic	12	17.408	88
10 connected and 50 trying to connect connect with 5 times background traffic	11	14.102	28

DHCP delay = Time of DHCP ack frame - Time of corresponding association request

by client

Table 8: DHCP delay in NAT mode, WiFi trace from a classroom of 50 students

No. of clients	**Minimum delay (seconds)**	**Average delay (seconds)**	**Maximum delay (seconds)**
50	0.24	14.05	62.51

Experiment with background traffic, Table 7

- The DHCP ACK packets collected in trace are less in number.
- The average DHCP delay in real trace obtained was double the value observed in testbed setting for the same number of clients.
- The maximum delay is one minute, which is comparitively large. The increased delay can be due to high channel contention and co-channel interference, since multiple access points are using the same channel.

7 SAFE versus Wicroft comparison

Here we are trying to simulate a 5-minute SAFE quiz using the Wicroft app with 20 clients. A 5 minutes SAFE quiz traffic are captured using a laptop in monitor mode in a network with an AP in WEP configuration. The class strength was 20. The packet traces are analysed and identified the types of URL request generate by the client, interdependency of URLs, size of URL request etc. based on this information a probabilistic model for SAFE client traffic is modelled and created a control file.

To capture the packets from the SAFE server, a TCP dump runs on the SAFE server. The size of responses, delay of response is identified. Using this information a probabilistic model for SAFE server also developed. This model act as a dummy SAFE server to handle modelled SAFE client requests.

The Wicroft experiment was conducted with 20 clients with modelled SAFE traffic. The experiment was repeated 4 times. Figure 43 shows the CDF of the response time of URL requests. From the graph, it can be seen that the delay values are close Or they are following the same pattern. Hence Wicroft succeeds in simulating actual SAFE quiz traffic. These results encourage to conduct SAFE scalability experiment with varying number of clients using WiCroft. The average, 10th percentile, 90th percentile values of response time obtained are SAFE [105, 10, 116] and WiCroft [126, 18, 200].

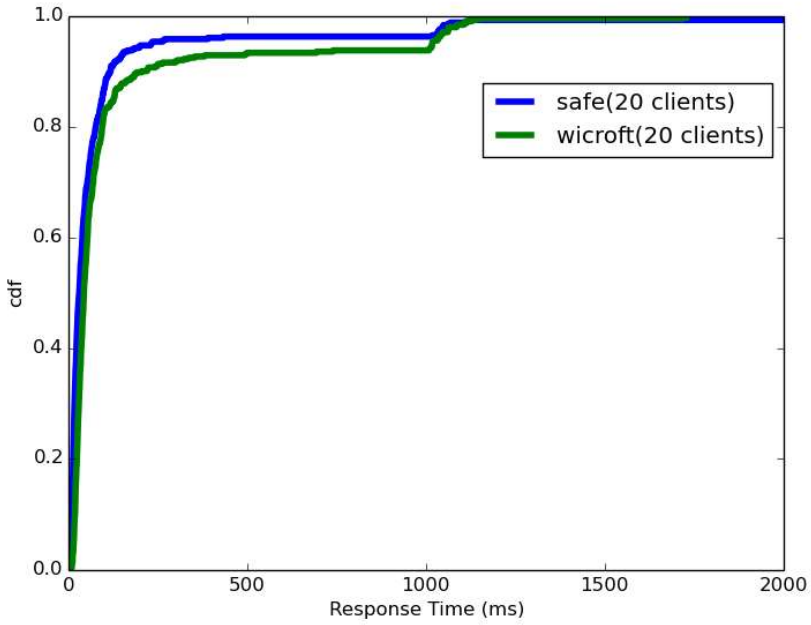

Figure 43: CDF of SAFE versus Wicroft Response time

Table 9: SAFE versus WiCroft Response time in milli seconds

	Average	**10th Pecentile**	**90th percentile**
SAFE	105	10	116
WiCroft	126	18	200

8 Scalability experiments

We observed that Wicroft with SAFE modelled traffic can mimic an actual SAFE quiz. Next thing is the scale testing. If SAFE quiz is conducted in a WiFi enviroment how much it can scale, i.e how many maximum number of clients can be suppported to give SAFE quiz. Experiments are conducted in three location as shown in Table 10.

- In location SL2-lab, experiments are conducted using Institute WiFi IITB-Wireless, there are multiple BSSIDs with the same SSID, and these APs supports dual bands. The number of clients participated in experiments and their association to different AP are also described in the table. With increasing the number of clients the response time obtained is also increases. Since the load are uniformly distributed across different BSSIDs, response time obtained with 58 and 47 clients have not much difference.

- The SIC201 location has high value for response time, with 40 clients the average response time obtained is more than 20 times that of the values obtained from the circular hall. From SIC201 it was observed that there are lots of packets from other BSS [3], channel reuse and co-channel interference results in increased delay at this location.

- Circular Hall is appeared to be a lesser congested environment. In the circular hall with 64 clients perform better than compared with SL2 Lab with same client strength.

- In Table 11, describe the response time obtained from scale test in the same location on two different days. We can see with 40 clients the average response time obtained on day 3 is more than 20 times than that obtained on day 2. These results show the dynamic nature of WiFi environment. The CDF of the response of locations described in Table 10 is shown in Figure 32. The more congested nature of SIC201 environment are clearly understood from CDF in terms of larger delay compared to other locations. The less congested Circular Hall environment gives comparatively lesses delay to request is also shown in the CDF.

- It is also interesting to know how many clients have the good experience is using WiFi. For this, a threshold can be defined for each URL response. If the response time of all URL requests is less than a threshold value then the client said to have the good experience. Table 12 shows the percentage of clients have a bad experience, i.e if the response time of at least one request exceeds the delay threshold value. The maximum value of request delay is also given, if the threshold value is maximum then we can support all the clients.

- For a threshold of 10 seconds, Circular Hall and SL2Lab can support around 63 clients.

- For a threshold of 5 seconds, Circular Hall can still support 64 clients and SL2Lab can support around 58 clients.

- For a threshold of 2 seconds, Circular Hall can support 40 clients and SL2Lab can support the same number of clients.

- A delay threshold of 5 sec is reasonable in the case of a SAFE quiz. With this value SIC201 location cannot support more than 30 students, this is also observed from previous SAFE quiz experience..

Table 10: SAFE modelled Scale test Response time in milli seconds

Location	AP used	**No. of clients**	**AP split**	**RT [Avg, Stdev, Max]**
SL2 Lab (Day 1)	IITB-Wireless (Aruba)	63	6,9,10,3,7,6,12,10	[625, 2960, 54026]
		58	9,9,11,4,4,4,8,8,1	[273, 537, 6396]
		47	1,11,9,5,9,1,6,5	[249, 484, 5640]
SIC 201	Aruba	40	40	[2890,6789,42739]
		25	25	[471,1626,16672]
Circular Hall	Aruba	64	64	[228,814,4463]
		40	40	[140,285,1601]

Table 11: SAFE modelled Scale test Response time in milli seconds, with Airtight AP

Location	AP used	**No. of clients**	**AP split**	**RT [Avg, Stdev, Max]**
SL2 Lab (Day 2)	Airtight	40	40	[494, 3090, 32948]
		30	30	[301, 1054, 11160]
SL2 Lab (Day 3)	Airtight	42	42	[8897, 19739, 146868]

Table 12: SAFE User (Bad) Experience

Location	**Clients**	**Max Delay (seconds)**	**Delay Thresholds (seconds)**				
			30	**10**	**5**	**2**	**1**
SIC201	40	42.7	10 %	25 %	45 %	70 %	78 %
	25	16.7	0 %	4 %	8 %	16 %	40 %
Circular Hall	64	4.5	0 %	0 %	0 %	6 %	26 %
	40	1.6	0 %	0 %	0 %	0 %	20 %
SL2 Lab	63	54.0	2 %	3 %	13 %	22 %	30 %
	58	6.4	0 %	0 %	2 %	9 %	26 %
	47	5.6	0 %	0 %	2 %	6 %	17 %

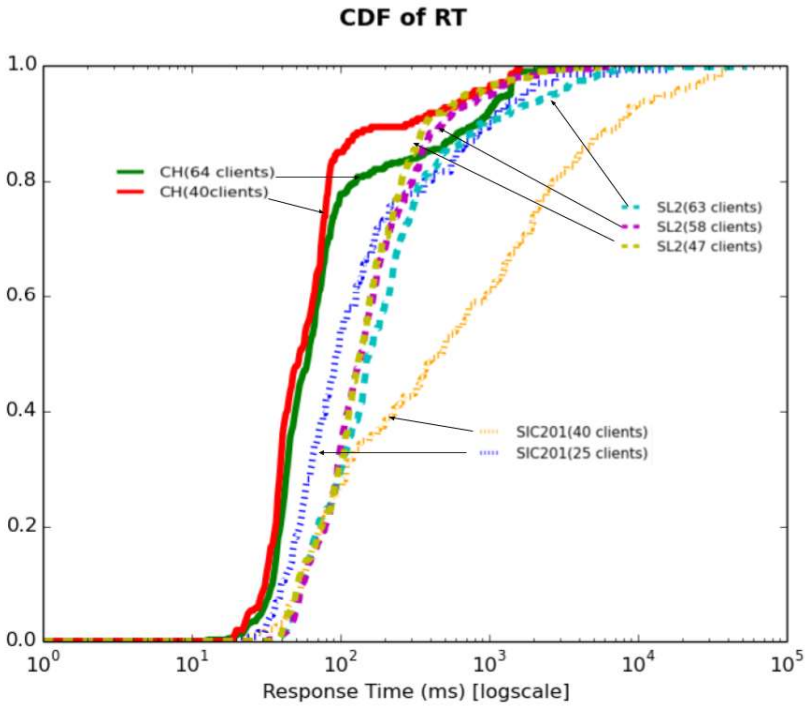

Figure 44: CDF of SAFE response time (different locations)

9 Conclusion

Following conclusion are made from trace analysis and observations from experiments,

- Wicroft provide a more reliable and robust framework for client-side performance measurement
- Wicroft is successful in mimicking SAFE traffic. So it is a generic framework to simulate any application that uses WiFi service
- In dense setting the response of URL request increase with number of users
- The traffic from neighbouring APs (in the same channel) and traffic generated from background applications from client device are the major reason behind contention.
- The channel contention can be reduced to some extent by adding firewall rule to block all irrelevant traffic
- The Bodhitree traffic is contributing only 10-14% of TCP traffic.The TCP data traffic generated by the background applications running on client devices contribute towards 90% of total TCP traffic through an access point.
- The IITB-Wireless(Aruba) AP support maximum 64 client associations and Microtek AP is able to support all 80 clients used in the testbed experiment.
- DHCP delay for obtaining IP address increased with client count and results in connectivity problem.
- The DHCP delay in obtaining IP address in NAT mode is less compared with a bridged mode of an access point.
- DHCP delay dominate in the connection delay experienced by clients in dense environment

10 Futurework

- Find number of clients can be supported by an AP with single frequency band.
- Improving the app to handle doze mode feature providede by android OS

11 Appendix

11.1 Wicroft server GUI is explained

Login with credentials, an alert box will pop up to notifying that new clients are connected. Inorder to see those clients in a dashboard, go to setting, dashboard setting. Choose or check required BSSIDs from all available and update the list. The selected bssid will be shown in the corresponding list. The dashboard will reflect only those clients which are connected to these selected BSSIDs. This will be helpful once we need to conduct an experiment and need only those clients within than environment.

11.1.1 Dashboard

1. From dashboard, the user can able to understand how many clients are connected, among them how many are passive and active

   (a) Active clients

   The clients with Wicroft app installed will send heartbeats in every minute, those heartbeats may not be reached at the server in the same interval due to WIFi disconnection issues due to congestion or clients may switch off the WiFi after sending few heartbeats. So in the design of the server, a threshold is kept to classify clients whether a heartbeat is received within last threshold value of intervals, if yes, it is an active client else, passive clients. Passive does not always mean the clients left the environment, there can be cases like socket connection error, WiFi disconnection so the heartbeats may not be reached in regular intervals.

   (b) Passive clients

   Those clients from which server didn't receive heartbeats during the threshold interval. Current threshold value in 90 seconds for heartbeat duration of 60 seconds.

2. Distribution of clients among BSSIDs

3. Heartbeat information of connected clients

11.1.2 Experiment configuration

1. Create experiment

   (a) This module helps to create and save an experiment. The user need to provide some details of the experiment like name, location, description and also the user can choose or not choose a file for this experiment.

   (b) The saved experiments are able to see in Experiment History module.

2. Change accesspoint

   When the experiments are needed to conducted using a specific access point. It is very difficult to instruct every client to instruct to do so. This module will help in this scenario, from server side we can request to selected clients to connect to a specific access point by providing it's SSID, security and required credentials. We have no control over BSSID. i.e if there are multiple BSSIDs with the same SSID we cannot force the clients to connect based on BSSID. The clients may choose an AP based on signal strength it receives.

   (a) Using Wicroft the clients able to connect to desired AP. The security types can be WEP, OPEN, WPA-PSK and EAP.

   (b) A timer also can be specified to indicate, start trying to connect after these many time.

11.2 Events and message format

The server uses the existing TCP connection from clients for sending and receiving messages. The event types and corresponding message format are described below. The **"action"** key in JSON message identifies the event type of the message like heartbeats, control file etc.

1. HeartBeat from client application

   The heartbeat contains connection information like IP, BSSID, SSID of the associated access point, a list of BSSIDs of nearest AP, device information like storageSpace, processor speed etc. The heartbeat message format is,

```
{
"action"       : "heartBeat",
"macAddress"   : "mac address",
"ip"           : "ip addr",
"port"         : "port no",
"bssid"        : "bssid ",
"ssid"         : "ssid ",
"bssidList"    : "ssid1, bssid1, rssi1; ssid2, bssid2, rssi2;...",
"numberOfCores": "4",
"memory"       : "882",
"storageSpace" : "1371"
}
```

   Once the heartbeat is received from a client, the server will maintain a map between mac address of client with information available from heartbeat and it will be updated

on each heartbeat, see Table 1. The reference to thread and input and output stream from TCP connection is also maintained in the map. The heartbeat will also help the server to maintain the list of active clients.

2. Sending control file from the server as part of starting an experiment, this message contains control file data, server time for executing the URL request at a specified time.

```
 {
"action"        : "controlFile",
"serverTime"    : "SERVER TIME",
"message"       : "CONTROL FILE DATA "
 }
```

3. Change AP settings request from server

   This message is a request for connecting to a specified target access point using the AP change feature in the client application. This helps in remotely connecting the clients to specific AP. The target APs SSID, BSSID, security and authentication details need to be specified in the message.

```
  {
"action"  : "apSettings",
"username": "user name ",
"password": "password ",
"bssid"   : "bssid value",
"ssid"    : "ssid value"
"security": "wep/eap"
}
```

4. Request log files from server

   This request is sent from the server for getting pending log files from a client. Request for pending log file can be done at any time. The client will send the pending log file to the server by HTTP POST method

```
{"action" : "getLogFiles"}
```

5. Experiment over message from client This message is to indicate the experiment is over for that client. So that the server can request for log files.

```
{
"action"    : "expOver",
"exp"       : "experiment_number",
"macAddress": "mac address"
}
```

6. Acknowledgement for controlfile received from client

   The client will send an acknowledgement to indicate the control file for the respective experiment is received and it is ready to start the experiment. This is a level of application reliability added so that from server side it is able to show the status of control file sending.

```
{
"action" : "ack",
"exp"    : "exp_number"
}
```

References

[1] WiFi Data frames. http://www.my80211.com/home/2009/12/5/80211-null-data-frames.html.

[2] Ratheesh kv. Server source code. https://github.com/ratheeshkv179/MTP.

[3] Swinky Mann. Dense WiFi To test scalability of SAFE app. http://www.cse.iitb.ac.in/synerg/lib/exe/fetch.php?media=public:mtp1-oct16:dense_wifi_mtp1_153050021.pdf.

[4] Cisco Networks. WiFi Frame Types. https://supportforums.cisco.com/document/52391/80211-frames-starter-guide-learn-wireless-sniffer-traces.

[5] Cisco Networks. WiFi Frame Types. https://supportforums.cisco.com/document/101431/80211-sniffer-capture-analysis-management-frames-and-open-auth.

[6] Ashish Sonone and Sanchit Garg. Network Load generator. https://github.com/ratheeshkv179/Loadgenerator/blob/master/Doc/sanchit-RnD-II-Report.pdf.