COMP4336 Assignment - Jeremy Chen (z5016815) and Jerry Xu (z3475829)

Table of Contents

Introduction

This is the report for our COMP4336 Assignment. This report is split into two separate sections, with one section covering the Link-Layer Analysis and the other section covering the Mobility Analysis. Each section will discuss how we designed our experiments, the code that we used, how we executed these experiments and the analysis of our experimental results. The Mobility Analysis is further split into three smaller subsections covering L4 Connectivity, L3 Hand-Off and L2 Hand-Off.

Link-Layer Analysis

Design

For the Link-Layer Analysis, we were required to visit several locations across campus and collect data regarding the 802.11 Protocol, Signal Strength, Data Rate and AP Density of each location. In order for our experiment to be well-designed, we had to approach the experiment using traditional scientific methods.[1]

Firstly, we had to ensure that the data we collected was reliable. This meant that we had to take into consideration potential outliers and other factors that may have adversely affected the results we obtained. To do this, we decided to obtain the data multiple times over the course of a few days. This meant that we had a lot of data to work with, thus ensuring that the effects of certain variables like weather and other sources of interference had a minimal impact on our overall results.

Our next step was ensuring that the data we collected was valid. In our following link-layer code section, we will go over the code that we used for this experiment. Since our group had 2 members, we were able to cross-check the code with each other, to ensure that we weren't missing anything, as well as localise any potential bugs or errors. We set up a Bitbucket storage, where we were able to double-check commits made by the corresponding partner. Once we were sure that our code was up to scratch, we had to ensure that during our data collection, the position we gathered data from was always the same during each of our trials. This was important because the position of walls and our distance from the router could have a potential impact on our results (such as the Wi-Fi signal strength).[2] We will further discuss how we carried out these experiments in the execution section of this report.

Code For Link-Layer Analysis

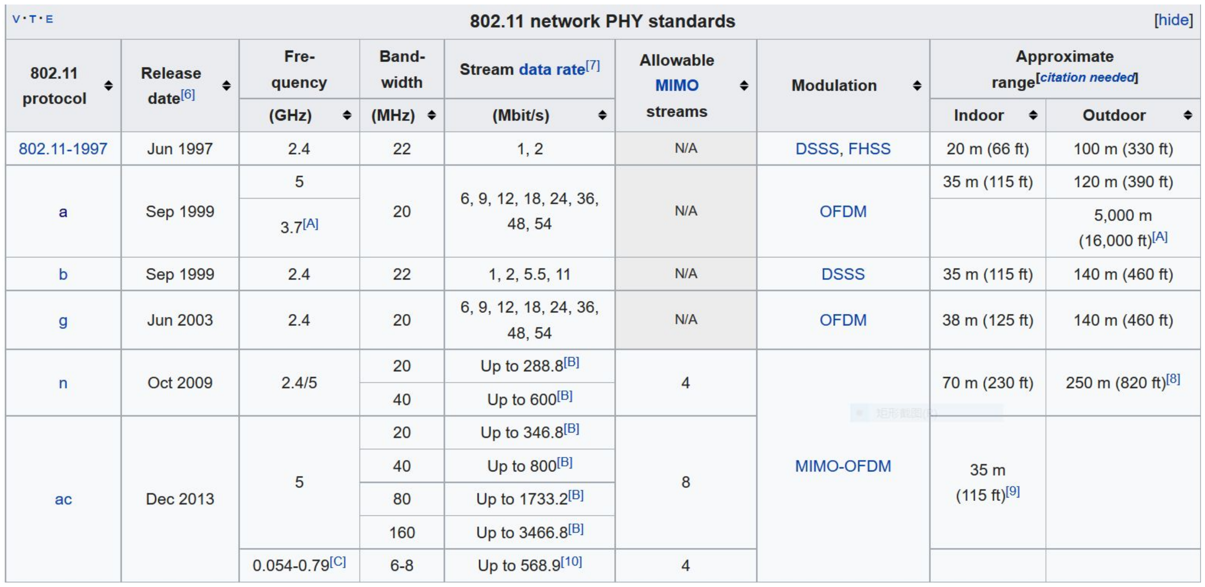
We are unable to show all our code for this section due to page constraints, but we will go over the key areas of the code that were used to obtain the data. This code was written in Java on Android Studio.

```
(ip >> 16 & 0xff), (ip >> 24 & 0xff));
                double freq = wifiInfo.getFrequency()*1.0/1000;
                txtIP.setText(wifiInfoView.getContext().getString(R.string.ip, decimalIP));
                txtBSSID.setText(wifiInfoView.getContext().getString(R.string.bssid_msg,
wifiInfo.getBSSID()));
                txtFreq.setText(wifiInfoView.getContext().getString(R.string.frecuency_msg,
String.format("%.1f", freq)));
                txtSpeed.setText(wifiInfoView.getContext().getString(R.string.speed,
String.valueOf(wifiInfo.getLinkSpeed())));
                if(Math.floor(freq) == 2) {
                    if(wifiInfo.getLinkSpeed() <= 11) {
                        protocol = "802.11 b";

txtProtocol.setText(wifiInfoView.getContext().getString(R.string.protocol, "802.11 b"));
                    }
                    else if(wifiInfo.getLinkSpeed() <= 54) {
                        protocol = "802.11 g";

txtProtocol.setText(wifiInfoView.getContext().getString(R.string.protocol, "802.11 g"));
                    }
                    else {
                        protocol = "802.11 n";

txtProtocol.setText(wifiInfoView.getContext().getString(R.string.protocol, "802.11 n"));
                    }
                }
                else {
                    if(wifiInfo.getLinkSpeed() <= 54) {
                        protocol = "802.11 a";

txtProtocol.setText(wifiInfoView.getContext().getString(R.string.protocol, "802.11 a"));
                    }
                    else if(wifiInfo.getLinkSpeed() <= 72) {
                        protocol =  "802.11 n";

txtProtocol.setText(wifiInfoView.getContext().getString(R.string.protocol, "802.11 n"));
                    }
                    else {
                        protocol = "802.11 ac";

txtProtocol.setText(wifiInfoView.getContext().getString(R.string.protocol, "802.11 ac"));
                    }
                }
                String s = "Connected wifi\n" + "SSID: " + wifiInfo.getSSID() + "\n" + "Freq: "
+ wifiInfo.getFrequency() + "GHz\n" + "Speed: "
                         + wifiInfo.getLinkSpeed() + "Mbps\n" + "Protocol: " + protocol + "\n";
```

This was the area of the code that we used to get the majority of our Link-Layer Data. This snippet of code does not include the initialization of variables or other non-major areas of our overall code. Here the WiFi-Info depicts the results from using the getConnectionInfo function from the Wifi-Manager class[3]. In writing this code, we recycled a lot of data from our previous lab work.

```
wifiInfo = wifiManager.getConnectionInfo();
```

To obtain the 802.11 Protocol, we mainly used the link speed function from the getConnectionInfo function for the WiFi-Manager class. Using an IEEE 802.11 Table[4], we were able to obtain the corresponding link speeds of each protocol and thus deduce which protocol the connection was using. Since some protocols had similar link speeds (such as 802.11a and 802.11g), in order to differentiate them, we also compared the frequency, since they each operated on distinct frequencies (Around ~5 GHz and ~2.4 GHz respectively). We cannot directly obtain the Wifi Protocol from "WifiInfo", since it doesn't implement method such as "getProtocol()", the only useful methods we used are "getLinkSpeed()" and "getFrequency()" as defined in Android Developers Document.



V·T·E **802.11 network PHY standards** [hide]

802.11 protocol	Release date[6]	Frequency (GHz)	Bandwidth (MHz)	Stream data rate[7] (Mbit/s)	Allowable MIMO streams	Modulation	Approximate range[citation needed] Indoor	Approximate range[citation needed] Outdoor
802.11-1997	Jun 1997	2.4	22	1, 2	N/A	DSSS, FHSS	20 m (66 ft)	100 m (330 ft)
a	Sep 1999	5	20	6, 9, 12, 18, 24, 36, 48, 54	N/A	OFDM	35 m (115 ft)	120 m (390 ft)
		3.7[A]						5,000 m (16,000 ft)[A]
b	Sep 1999	2.4	22	1, 2, 5.5, 11	N/A	DSSS	35 m (115 ft)	140 m (460 ft)
g	Jun 2003	2.4	20	6, 9, 12, 18, 24, 36, 48, 54	N/A	OFDM	38 m (125 ft)	140 m (460 ft)
n	Oct 2009	2.4/5	20	Up to 288.8[B]	4	MIMO-OFDM	70 m (230 ft)	250 m (820 ft)[8]
			40	Up to 600[B]				
ac	Dec 2013	5	20	Up to 346.8[B]	8		35 m (115 ft)[9]	
			40	Up to 800[B]				
			80	Up to 1733.2[B]				
			160	Up to 3466.8[B]				
		0.054-0.79[C]	6-8	Up to 568.9[10]	4			

For the Data Rate, which is measured in Mbps (Megabytes Per Second), we again used the link speed function of the getConnectionInfo function of WiFi-Manager. We also obtained the frequency of the connection by using the getFrequency function of the getConnectionInfo function from the Wifi-Manager class.

For the Signal Strength, we used a separate snippet of code as shown below. The initialization of some variables have been removed due to page constraints. The Signal Strength was recorded in dBm (decibel milliwatts).

```
public void scanNetworks() {

       boolean scan = wifiManager.startScan();

       if(scan) {

            results = wifiManager.getScanResults();
            int i = 0;
            StringBuilder sb = new StringBuilder();;
            for(ScanResult s: results) {
                if(s.SSID.equals("uniwide")) {
                    sb.append(String.valueOf(s.SSID + "\n"));
                    sb.append(String.valueOf(s.BSSID + "\n"));
                    sb.append(String.valueOf(s.level + " dBm\n"));
                    sb.append(String.valueOf(s.frequency + " GHz\n"));
                    i++;
                }
            }
            String s = "#uniwide: " + i;
            counter.setText(s);
            sb.append(s);
            sb.append("\n");
            generateNoteOnSD("networks", sb.toString());
        }
```

In this section of the code, to obtain the signal strength, we use the getScanResults function of WiFi-Manager class to obtain a range of data from surrounding APs. We then read these results, filtering out non-uniwide SSIDs to obtain the dBm of the signal strength for the connection that we are currently using.

Similarly, we also used the snippet of code above to find the AP density of the location where we collected the data. Assuming that AP density referred to the number of Access Points in the vicinity of our location, we incremented a counter for each AP that we could detect in our surroundings, we then returned the counter value as a TextView after completion.

Execution of Link-Layer Experiments

We decided to conduct our link-layer experiments over a few trials during the course of a few days to improve the reliability of our data. We started out with a map of the UNSW campus[5] and set out a path that would allow us to visit each significant campus location. Overall, we plotted 38 different locations across UNSW, that would give us a holistic view of the Link-Layer data across the campus.

At the time, we would use our phones to record and screenshot the data, we would then transcribe this data onto an online spreadsheet. Each trial took us approximately 2-3 hours to traverse the campus. Altogether we completed 3 trials over the course of two days. These trials were conducted over distinct time periods. We then obtained the average of all these results to obtain the final data used for our analysis, while carefully removing any potential outliers. The results will be presented and discussed in the link-layer results section of this report.

There were a few variables we had to keep track of during this experiment. Since we were conducting multiple trials, we had to ensure that the location where we recorded the data was similar to the location in our previous trial. This was because distance and structures such as walls could have a potential effect on the strength of the Wi-Fi signal. We also tried to avoid days that were too rainy to lessen the effect of weather on our overall results. [6]

Shown below is a compilation of some of the photographic evidence taken during our experiment. These were some of the notable locations that we visited.

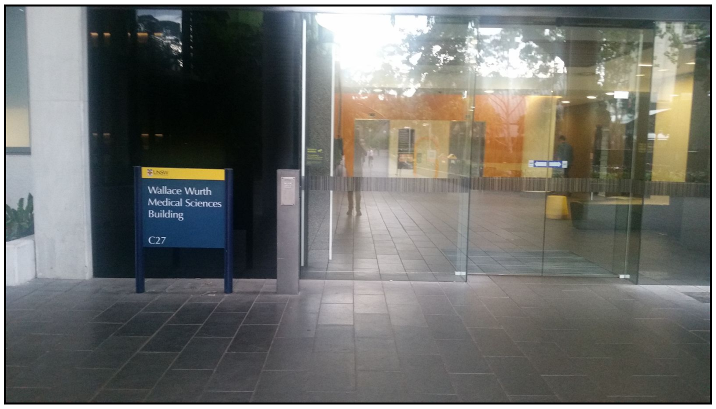
Taken at the Fitness and Aquatic Centre

Taken at The International Square

Taken at K17

Taken at Village Green

Taken at The Main Library Lawn



Taken at the Wallace Wurth Building

Link-Layer Experimental Results

Trial 1 Results

Location	802.11 Protocol	Signal Strength (dbm)	Data Rate (Mbps)	AP Density	Frequency (GHz)
Ainsworth Building	ac	-84	243	48	5.5
Barker Street Parking	ac	-89	270	16	5.3
Basser Steps	n	-74	52	21	5.7
Biological Sciences	ac	-87	81	21	5.7
Blockhouse	ac	-90	81	30	5.7
Botany Street Parking	ac	-88	324	13	5.8
Central Lecture Block	ac	-59	162	15	5.8
Chancellery	ac	-78	243	33	5.7
Chemical Sciences	ac	-88	81	20	5.3
Civil Engineering	ac	-78	324	28	5.7
Electrical Engineering	ac	-66	108	49	5.2
Fitness and Aquatic Centre	g	-70	52	8	2.4
Goldstein Hall	ac	-58	162	14	5.8
Hilmer Building	ac	-76	216	29	5.2
International House	ac	-79	108	2	5.7
International Square	ac	-84	81	48	5.3
J12 Newton	ac	-89	270	17	5.3
John Clancy Auditorium	ac	-88	108	25	5.2
John Niland Science	ac	-42	324	20	5.7
K17	ac	-83	243	18	5.7
Law Library	ac	-79	81	21	5.3
Main Library	ac	-83	360	18	5.3
Main Library Lawn	ac	-76	324	23	5.3
Mathews Theatre	ac	-87	108	17	5.2
Morven Brown	ac	-78	162	11	5.7

Physics Lawn	n	-77	54	19	5.7
Quadrangle	n	-53	162	13	2.5
Robert Webster	ac	-69	216	33	5.7
Science Theatre	ac	-66	324	26	5.7
Science Theatre Lawn	ac	-62	108	17	5.7
Scientia Lawn	ac	-45	162	18	5.7
The Red Centre	n	-64	54	26	5.8
Tyree Energy Technologies	ac	-88	243	21	5.7
University Terraces	ac	-74	324	9	5.2
UNSW Business School	ac	-65	216	34	5.2
UNSW Hall	ac	-70	162	21	5.2
Village Green	a	-88	26	25	5.8
Wallace Wurth	n	-85	108	14	2.4

Trial 2 Results

Location	802.11 Protocol	Signal Strength (dbm)	Data Rate (Mbps)	AP Density	Frequency (GHz)
Ainsworth Building	ac	-74	135	25	5.3
Barker Street Parking	ac	-66	162	19	5.7
Basser Steps	ac	-86	324	27	5.3
Biological Sciences	ac	-71	81	12	5.5
Blockhouse	ac	-82	108	20	5.2
Botany Street Parking	ac	-72	162	18	5.7
Central Lecture Block	ac	-82	216	12	5.2
Chancellery	ac	-66	243	44	5.5
Chemical Sciences	ac	-89	81	25	5.3
Civil Engineering	ac	-89	216	35	5.2
Electrical Engineering	ac	-59	162	32	5.7

Fitness and Aquatic Centre	ac	-46	324	7	5.8
Goldstein Hall	n	-57	104	27	2.5
Hilmer Building	ac	-85	162	17	5.7
International House	ac	-54	270	9	5.8
International Square	ac	-89	270	27	5.7
J12 Newton	n	-57	54	11	5.8
John Clancy Auditorium	ac	-83	104	19	5.8
John Niland Science	ac	-85	243	35	5.2
K17	ac	-85	162	20	5.3
Law Library	ac	-81	162	31	5.2
Main Library	ac	-81	324	31	5.8
Main Library Lawn	ac	-81	108	38	5.7
Mathews Theatre	ac	-81	144	27	5.5
Morven Brown	ac	-75	162	18	5.2
Physics Lawn	a	-75	26	19	5.2
Quadrangle	ac	-74	162	27	5.7
Robert Webster	ac	-73	162	25	5.7
Science Theatre	ac	-89	162	28	5.6
Science Theatre Lawn	ac	-79	162	28	5.2
Scientia Lawn	ac	-86	216	17	5.8
The Red Centre	ac	-73	162	32	5.8
Tyree Energy Technologies	ac	-80	108	13	5.2
University Terraces	n	-63	52	14	5.3
UNSW Business School	ac	-69	360	17	5.7
UNSW Hall	ac	-87	108	17	5.2
Village Green	ac	-71	108	11	5.2
Wallace Wurth	a	-86	27	19	5.8

Trial 3 Results

Location	802.11 Protocol	Signal Strength (dbm)	Data Rate (Mbps)	AP Density	Frequency (GHz)
Ainsworth Building	ac	-89	108	23	5.3
Barker Street Parking	ac	-66	130	4	5.7
Basser Steps	ac	-86	260	16	5.7
Biological Sciences	ac	-71	65	21	5.5
Blockhouse	ac	-81	87	25	5.2
Botany Street Parking	ac	-76	130	13	5.2
Central Lecture Block	ac	-82	173	12	5.2
Chancellery	ac	-79	195	43	5.5
Chemical Sciences	ac	-79	65	23	2.4
Civil Engineering	ac	-86	173	15	5.8
Electrical Engineering	ac	-89	130	19	5.3
Fitness and Aquatic Centre	ac	-55	260	5	5.8
Goldstein Hall	n	-48	84	22	2.5
Hilmer Building	ac	-86	130	19	5.3
International House	n	-63	216	8	2.4
International Square	ac	-77	216	23	5.3
J12 Newton	g	-70	44	20	5.8
John Clancy Auditorium	ac	-76	84	16	5.7
John Niland Science	ac	-89	195	42	5.2
K17	ac	-57	130	13	5.7
Law Library	ac	-89	130	12	5.3
Main Library	ac	-81	260	28	5.7
Main Library Lawn	ac	-66	87	31	5.5
Mathews Theatre	ac	-74	116	9	5.5
Morven Brown	ac	-71	130	26	5.2
Physics Lawn	a	-75	21	22	5.2
Quadrangle	ac	-74	130	27	5.7
Robert Webster	ac	-86	130	24	5.7
Science Theatre	ac	-66	130	11	5.7

Science Theatre Lawn	ac	-79	130	28	2.4
Scientia Lawn	ac	-81	173	25	5.8
The Red Centre	ac	-77	130	33	5.3
Tyree Energy Technologies	ac	-80	87	18	5.2
University Terraces	a	-54	42	12	5.3
UNSW Business School	ac	-82	288	24	5.2
UNSW Hall	ac	-87	87	17	5.2
Village Green	n	-71	87	21	2.4
Wallace Wurth	a	-83	22	14	5.8

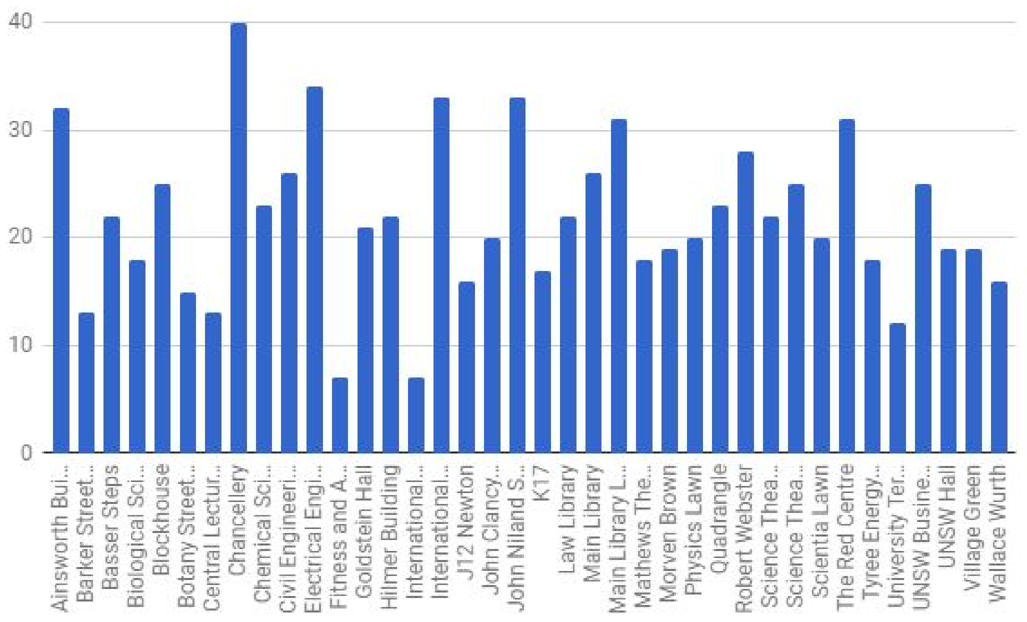
Final Average of Results

Location	802.11 Protocol	Signal Strength (dbm)	Data Rate (Mbps)	AP Density	Frequency (GHz)
Ainsworth Building	ac	-83	162	32	5.4
Barker Street Parking	ac	-74	188	13	5.6
Basser Steps	ac	-82	212	22	5.6
Biological Sciences	ac	-77	76	18	5.6
Blockhouse	ac	-85	92	25	5.4
Botany Street Parking	ac	-79	206	15	5.6
Central Lecture Block	ac	-75	184	13	5.4
Chancellery	ac	-75	227	40	5.6
Chemical Sciences	ac	-86	76	23	4.4
Civil Engineering	ac	-85	238	26	5.6
Electrical Engineering	ac	-72	134	34	5.4
Fitness and Aquatic Centre	ac	-57	212	7	4.7
Goldstein Hall	ac	-55	117	21	3.6
Hilmer Building	ac	-83	170	22	5.4
International House	ac	-66	198	7	4.7
International Square	ac	-84	189	33	5.5
J12 Newton	ac	-72	123	16	5.7

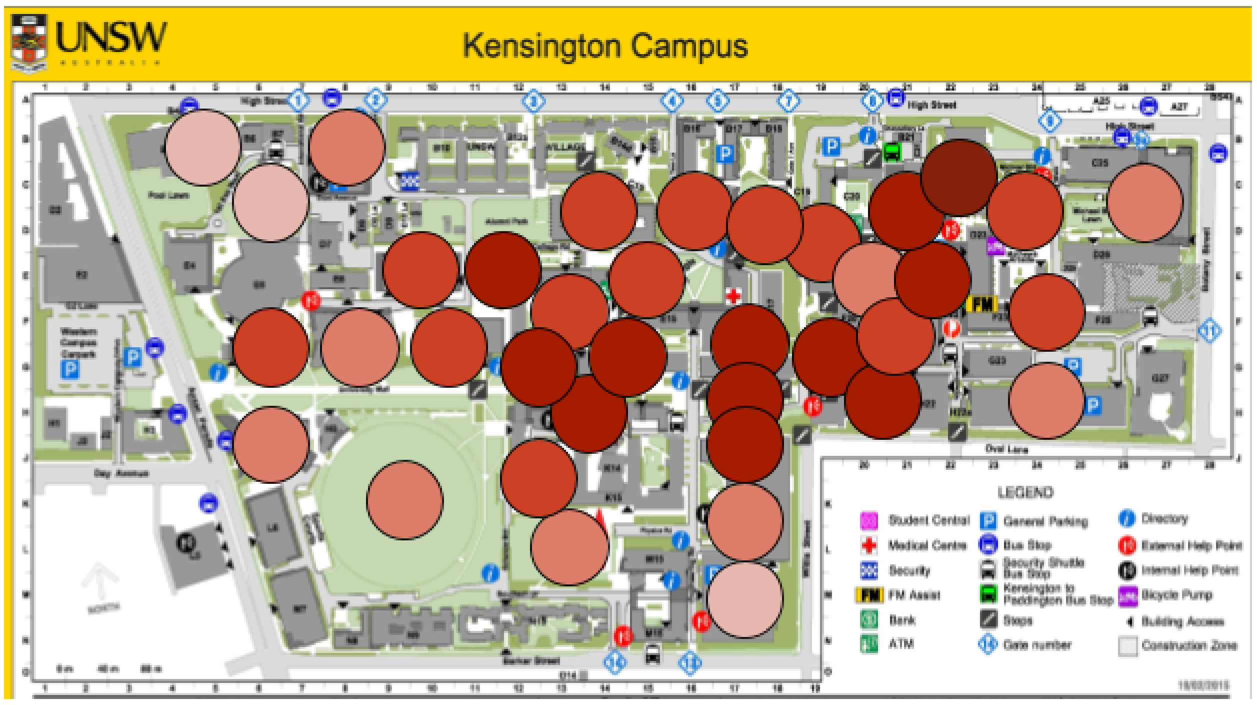
John Clancy Auditorium	ac	-83	99	20	5.6
John Niland Science	ac	-72	254	33	5.4
K17	ac	-75	179	17	5.6
Law Library	ac	-83	125	22	5.3
Main Library	ac	-82	315	26	5.6
Main Library Lawn	ac	-75	173	31	5.5
Mathews Theatre	ac	-81	123	18	5.4
Morven Brown	ac	-75	152	19	5.4
Physics Lawn	a	-76	34	20	5.4
Quadrangle	ac	-67	152	23	4.7
Robert Webster	ac	-76	170	28	5.7
Science Theatre	ac	-74	206	22	5.7
Science Theatre Lawn	ac	-74	134	25	4.5
Scientia Lawn	ac	-71	184	20	5.8
The Red Centre	ac	-72	116	31	5.7
Tyree Energy Technologies	ac	-83	146	18	5.4
University Terraces	ac	-64	140	12	5.3
UNSW Business School	ac	-72	288	25	5.4
UNSW Hall	ac	-82	119	19	5.2
Village Green	ac	-77	74	19	4.5
Wallace Wurth	n	-85	53	16	4.7

Link-Layer Results / Performance Analysis

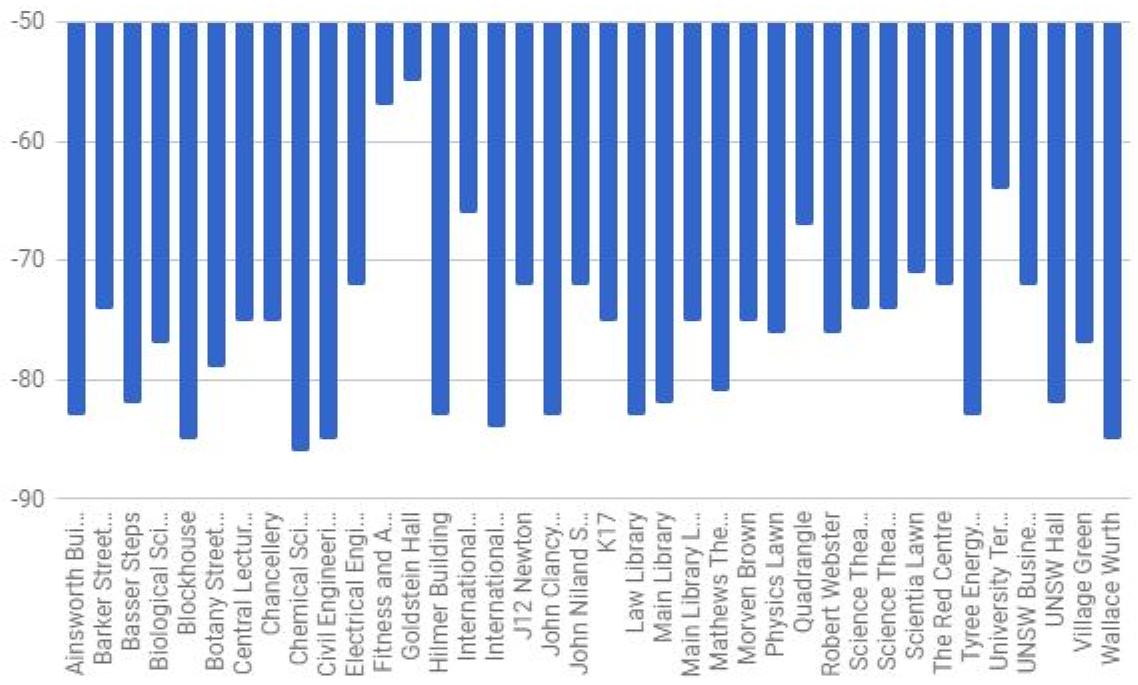
Comparison of AP Density Across Locations (Bar Graph)



AP Density Map Analysis (Heat Graph)



Signal Strength Comparison in Dbm (Bar Graph)



From the results obtained, it is evident that there is a lot of variation in regards to some aspects of the data. Data Rate seems to fluctuate quite significantly over all the trials. On some days it can reach upwards of 300 Mbps at a location, on other days, it can drop to around 50 Mbps. As a result, we are unable to draw any conclusive inferences based on the data collected.

In terms of the signal strength, a bar graph is shown above depicting the data that we obtained. We can deduce that generally, the signal strength has a low standard deviation and tends to not fluctuate too widely across locations and trials. The only notable exceptions occur in more remote locations such as the Fitness and Aquatic Centre, International House and University Terraces. This is an anomaly, and we likely infer that this is due to less interference and noise in remote areas. It is also likely that the signal strength is stronger because the area is not as densely populated as other areas across the university.[7]

In regards to AP Density, a heat map is shown above, where a gradient of colours are used to indicate low-high AP density areas. We can ascertain that as we approach more inner locations on the campus such as the Red Centre, the AP Density tends to increase considerably. It is also noticeable that the area surrounding the Ainsworth Building also has a high AP concentration. Another location of interest is the Chancellery and Student Office area which seems to possess an extremely high AP Density. As we move further away from the inner locations of the campus, we begin to notice that the AP Density slowly dissipates. This is likely due to the fact that the APs are spread out more in regions where there are more students and a denser population

In terms of the 802.11 Protocol, we can notice that for the majority of locations, they all tend to follow 802.11ac / 802.11a Protocol. As a result, we can also notice that the corresponding frequencies for each location generally hover around 5 GHz. The SSID for all these locations was Uni-Wide.

Mobility Analysis

Design

L4 Connectivity

In checking L4 Connectivity, our approach was to send HTTP Requests to a CSE Web Server on a timer based setup. These HTTP Requests would be sent every millisecond and we would retrieve a subsequent Response straight after sending the request. If the HTTP Responses were received continuously without failure after switching APs, that would indicate that TCP was preserved. However, if there was a failure in the Responses upon switching APs, that would indicate that the TCP connection had been disconnected / reconnected.

The HTTP Requests would be sent with the assistance of the Volley Library. This was a HTTP Library that made it very convenient to send consecutive HTTP Requests and reading Responses received. [8]

L3 Handoff Delay

For our L3 Handoff, our plan was to walk around Campus while continuously monitoring for any changes to our IP address. When the IP address was dropped, we assumed that the new IP address would thus become “0.0.0.0” for a brief period of time. We would thus start a timer, when the IP became “0.0.0.0”. This timer would end when the IP address was refreshed with a new one after breaking TCP connection. The value returned by the timer would thus be the L3 Handoff.

However, upon further research, we noticed that since the campus was composed of a single large subnet, there would be no changes to the IP address, regardless of how far we walked across campus.[9] As a result, we could not obtain the L3 Handoff since there was no change in the IP address. However, we were able to change our IP address by switching from Uniwide to a WiFi Hotspot created by a phone. As a result, we do have some data that we can share in regards to L3 Handoff, although it does not apply for switching APs that share the same SSID (Uniwide).

L2 Handoff Delay

Initially, our approach for the L2 Handoff Delay involved measuring the time between losing the current BSSID and then gaining a new BSSID. This was the method that we thought most appropriate. Upon executing this design, our final results ended up hovering around 1-10ms. Unfortunately, did this not correlate with the data that we obtained from our research. [10]

Returning to our research, we discovered that the L2 handoff delay should consist of 3 primary components. This involved the probe delay, authentication delay and reassociation delay. The L2 handoff delay was in fact the combination of all these 3 minor delays, with the probe delay constituting 90% of the actual delay. [11] This meant that the average L2 handoff delay obtained from the experiments that we researched would usually deviate around a few hundred milliseconds.

Since this was in contrast to our original results of 1-10ms, we decided to conduct further experimentation with various other approaches. Our next approach involved an assumption that during the dissociation/reassociation period there would be a brief period where we would be unable to obtain an internet connection as our device was transitioning between networks. So we set up a continuous stream of HTTP requests that were sent every millisecond and then we monitored for any breaks in between requests where we did not receive any replies from the requests we sent. Unfortunately, this method did not work out as the replies were mostly persistent even during reassociation/disassociation, indicating TCP Persistence.

Our next approach involved using Android API and Wifi Manager/ Wifi Info/ Network Info [12]. We would attempt to manually call disconnect / reconnect and then wait for the duration it would take for the device to reassociate to its network. This method did not work as effectively as we had hoped and the delay became far too long (usually over 2 seconds) compared to the results of other experiments we researched. Next we tried to follow the implementation outlined in the research we found. The users in that particular scenario had created an 802.11 Sniffer that could detect and record Probe Requests and Reassociation Requests, they would then measure the amount of time between obtaining the initial Probe Request and subtract that from the time of the Reassociation Request to obtain the L2 Handoff Delay. [13] Unfortunately this proved to be too difficult, so ultimately, we decided to keep our original data from switching BSSIDs.

Code for Mobility Analysis

L4 Connectivity

```
public void sendRequest() throws IOException {

   counter2++;
   StringRequest stringRequest = new StringRequest(Request.Method.GET, url,
           new Response.Listener<String>() {
               @Override
               public void onResponse(String response) {
                   t1.setText(response.substring(0, 1000));
                   time = System.currentTimeMillis();
                   MyString = Long.toString(time);
```

```
                    t2.setText(MyString + "\n" + wm.getConnectionInfo().getBSSID());

                    if (counter == 1) {
                        difference = time - updated;
                        String aString = Long.toString(difference);
                        t3.setText(aString);
                        counter = 0;
                    }


                }
            }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            t1.setText("Disconnected" +"\n");
            updated = time;
            counter = 1;
        }
    });
    queue.add(stringRequest);
}
```

This was the main code used for L4 Connectivity. This is just the key area of the code, and does not include the initialization of several variables and the timer. The Volley HTTP Library was used to send a HTTP Request to a CSE Web Server [14]. These requests were sent every millisecond through a timer. Upon sending a Request, a Response would be received. If a Response was not received when switching between Access Points, that would indicate that the TCP connection had been broken. This would result in an Error Response, which would output “Disconnected” and record the time of the disconnection. However, if Responses are still received steadily without interruption during an AP switch, then that would indicate that the TCP connection was preserved during the switch. The difference in time was recorded for any broken TCP connections.

L3 Hand-Off Delay

```
public void checkIP() throws IOException {

   IP = Formatter.formatIpAddress(wm.getConnectionInfo().getIpAddress());
   String base = "0.0.0.0";
   if (IP.equals(base) && (counter == 0)) {
       time = System.currentTimeMillis();
       counter++;
       counter2 = 0;

   }

   if (!(IP.equals(base)) && (counter2 == 0)){
       difference = System.currentTimeMillis() - time;
       String aString = Long.toString(difference);
       t2.setText(aString);
       counter = 0;
       counter2++;
   }
   counter3++;
   t1.setText(IP + "\n" + counter3 + "\n");
}
```

This was the written code used for the L3 Hand-Off. Since L3 Hand-Off was the time between losing an IP address to gaining a new one, our understanding was that during the losing process, the IP would temporarily become 0.0.0.0, before accepting its new IP. As a result, we set up this function in a timer to record the moment an IP address is lost (becomes 0.0.0.0) until the moment an IP is regained. Unfortunately, this code did run into some complications, which we will discuss further in our Results section.

L2 Hand-Off Delay

```
public void checkBSSID() throws IOException {

   WifiInfo wifiInfo = wm.getConnectionInfo();

   time = System.currentTimeMillis();
   t3.setText(Long.toString(time));

   if ((counter == 0)) {
       time = System.currentTimeMillis();
       MyString = wifiInfo.getBSSID();
       counter++;
   }

   if (!(wifiInfo.getBSSID().equals(MyString)) && (counter == 1)){
       difference = System.currentTimeMillis() - time;
       String aString = Long.toString(difference);
       t2.setText(aString);
       counter = 0;
   }
   counter3++;
   t1.setText(wifiInfo.getBSSID() + "\n" + counter3 + "\n");
}
```

This was our original approach for our L2 Hand-Off. We operated under the assumption that we could set a timer and calculate the difference in time it takes to switch between BSSIDs. We monitor for changes to BSSID and then calculate a difference (in ms) when a change in BSSID has occurred.

Prior to this, our hypothesis was that for a brief moment, the BSSID would be “00:00:00:00:00:00” as it disconnects from its original AP, our plan was to measure the amount of time the BSSID was “00:00:00:00:00:00” and then use that time as the L2-Handoff. However, this did not go according to plan, as the BSSID appeared to transition nearly instantly in every scenario that was tested. We also tested using the Wifi Manager and Network Info API, with commands such as isConnected(), disconnect(), reassociate() and reconnect(). [15]

Execution of Mobility Layer Experiments

L4 Connectivity

When testing L4 Connectivity, we followed our original design and code. Our approach was to walk around the entire campus a few times and measure for changes in TCP connectivity. This was done by monitoring changes to HTTP Requests. For additional reliability, to account for outliers and other random variables, we decided to conduct our experiments multiple times over the course of a single day.

L3 Handoff

For the L3 Handoff time, we followed the original design outlined in our design section above. The strategy was to walk around campus and check for any particular changes to the IP address with the App running. Upon any changes to the IP address, we would be alerted and the transition time recorded on the screen.

After realising that the IP address was constant throughout the entire university campus, we tried switching between EduRoam and UniWide, but surprisingly, the IP address was the same for both SSIDs. However, we did obtain changes in IP address when switching between UniWide and Mobile Hotspots. So we decided on using and recording the handoff delay time experienced in that particular scenario, details will be discussed further in the results section

L2 Handoff

Execution for L2 Handoff was quite complicated and many trials/experiments were conducted with several changes made for each trial. Our first trial with L2 Handoff, we tested for the amount of time it took to transition between BSSIDs. The results from this initial trial opposed the sample data we had obtained from our research, our hypothesis was that we were not factoring in the probe delay.

For our second trial, we used WifiManager, WifiInfo and NetworkInfo to manually disconnect from the AP, and then attempt to reconnect again instantly. [16] Using this new code, we then walked around the entire campus a few times recording our data. The results obtained in this scenario were quite lengthy in duration, oftentimes amounting to more than a few seconds for each campus area.

Our next trial involved creating a sniffing application that could detect outgoing probe requests and incoming responses. Each individual request and response would have their corresponding time recorded. Upon obtaining all the data, subtraction would give us the total L2 Delay. Initially, we tried downloading applications that served this purpose such as ‘Packet Capture’ [17] as well as trying to connect WireShark [18] to our android device. These attempts did not work, we also tried our hand at creating our own android program for this particular application, but it proved to be too difficult for us to manage. Ultimately, we settled on using the data obtained from the first trial (measuring switches between BSSIDs).

Results (Average)

L4 Connectivity

L4 Connectivity	Preserved / Disconnected (Duration)
K17 - J17	Disconnected - 6318 ms
J17 - J18	Disconnected - 8312 ms
J18 - Elec Building	Preserved
Elec Building - E15	Preserved
E15 - Food Court	Preserved
Food Court - E12	Preserved
E12 - F10	Preserved
F10 - Hilmer Building	Preserved
E19 - F20	Preserved
F20 - F22	Preserved
F22 - Main Lib	Disconnected - 2557 ms
Main Lib - Mathews Theatre	Preserved
Mathews Theatre - C24	Preserved
C24 - C25	Preserved
C25 - C27	Preserved
C27 - Pavilions	Disconnected - 1427 ms
Pavilions - F23	Disconnected - 2327 ms
F23 - C20	Disconnected - 6230 ms

L3 Hand-Off

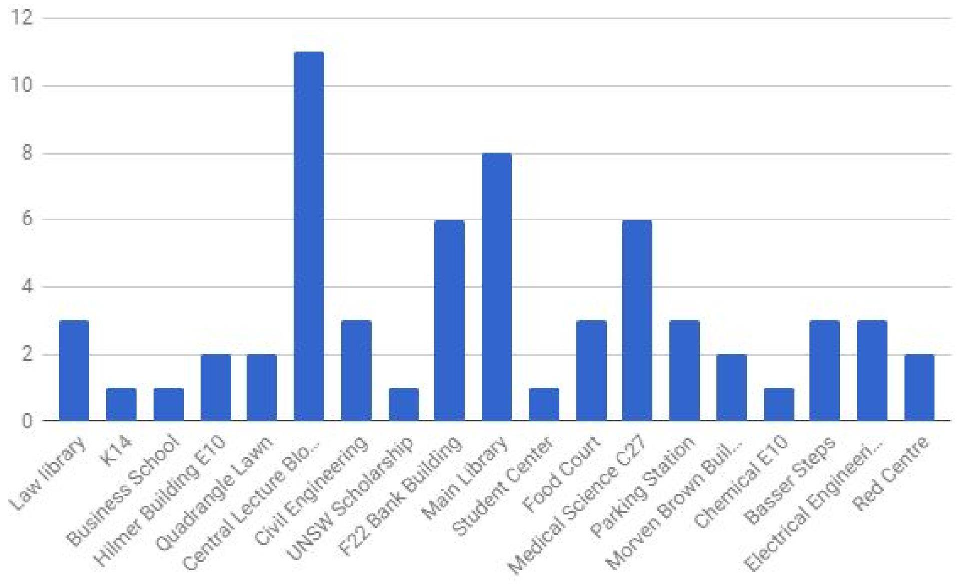
L3 Handoff Delay	Uniwide -> HotSpot (ms)	HotSpot -> Uniwide (ms)
K17	1117	3576
J18	1162	3278
J17	1127	3985
Electrical Building	1230	6850
E15	1190	3345
Food Court	1123	3056

E12	1176	4320
F10	1078	3020
Hilmer Building	1120	6750
E19	1143	6340
F20	1165	7080
Main Library	1176	3823
Mathews Theatre	1154	3145
C24	1132	3245
C25	1188	3344
C27	1195	6650
Pavilions	1190	6170
F23	1162	6030
C20	1155	6230
Richie Theatre	1184	7430
Law Library	1133	7250
E10	1180	6850

L2 Hand-Off Delay

Location	Hand-Off Delay (ms)
Law library	3
K14	1
Business School	1
Hilmer Building E10	2
Quadrangle Lawn	2
Central Lecture Block E19	11
Civil Engineering	3
UNSW Scholarship	1
F22 Bank Building	6
Main Library	8
Student Center	1
Food Court	3
Medical Science C27	6
Parking Station	3

Morven Brown Building C20	2
Chemical E10	1
Basser Steps	3
Electrical Engineering	3
Red Centre	2



Results Analysis

L4 Connectivity

From the results obtained, it is apparent that for most locations across the UNSW campus, the TCP connections seemed to be preserved. This means that there does not seem to be any losses in connectivity to an internet connection while switching between Access Points. This would likely ensure a very stable internet connection when transitioning between different locations across the campus.

In locations where TCP is disconnected, there appears to be a brief period of 1 - 10 seconds where there is no internet connection (as the HTTP Requests are unable to retrieve responses). This may be due to a delay in re-establishing a new TCP connection or some other miscellaneous delay.

L3 Handoff

For L3 Handoff, we did not notice any changes to our IP address as we moved across campus. This was after visiting more than 38 distinct areas across the UNSW campus. The likely reasoning behind the constant IP is due to the fact that UNSW is composed of one big subnet (129.94.8.*). [19] This was also hinted at in our research as well, which suggested that large campus areas are unlikely to experience L3 delays due to being composed of one big subnet only. As a result, we did some additional experimentation and trials by switching between a mobile hotspot network and Uniwide (Eduroam did not yield a change in IP address). From the data provided, it seems evident that it takes a significantly longer time to connect to and obtain a new IP address from Uniwide, than from a Mobile Hotspot. This may be likely due to the fact that the client is re-entering a large subnet, whereas entering a new smaller network (not a subnet) is a lot more easier and faster in comparison. [20]

L2 Handoff

From the results given, it is evident that there does not appear to be a significant level of L2 Handoff Delay observed from our experimental results, and variation between each location amounts to roughly a few milliseconds. This is in contrast to other experimental data which seems to indicate an average of a few hundred milliseconds. The likely explanation is that since we are only measuring the amount of time it takes to transition between BSSIDs, we do not take into consideration Probe Delay. Another possibility is that UNSW incorporates a cache of surrounding APs that share the UNSW SSID, our research suggests that this could potentially increase the speed of L2 Hand-Off delays very significantly. [21]

We did notice a spike in the Handoff delays that we recorded in locations such as the Main Library and the CLB. A possible reasoning is due to the higher number of Access Points at that location, a Scanner or Cache may have taken slightly longer than normal. In retrospect, designing a 802.11 sniffer that works for Android would be optimal and allow us to detect the release of Probe and Re-Association / Authentication Requests, to measure the total time taken for re-association, however, that task ultimately proved too difficult for us.

References

1. Dummies. 2017. Designing Experiments Using the Scientific Method - dummies. [ONLINE] Available at: http://www.dummies.com/education/science/designing-experiments-using-the-scientific-method/. [Accessed 3 October 2017].
2. Broadband: What affects your Wi-Fi signal (block, setup, signal, strength, weak, wi-fi, wireless) . 2017. Broadband: What affects your Wi-Fi signal (block, setup, signal, strength, weak, wi-fi, wireless) . [ONLINE] Available at: https://support.zen.co.uk/kb/Knowledgebase/Broadband-What-affects-your-WiFi-signal. [Accessed 4 October 2017].
3. WifiManager | Android Developers . 2017. WifiManager | Android Developers . [ONLINE] Available at: https://developer.android.com/reference/android/net/wifi/WifiManager.html. [Accessed 4 October 2017].
4. "Wi-Fi Alliance: Organization". Official industry association web site. Retrieved September 23, 2017.
5. Campus Maps | UNSW Current Students. 2017. Campus Maps | UNSW Current Students. [ONLINE] Available at: https://student.unsw.edu.au/maps. [Accessed 4 October 2017].

6. Factors Affecting Wireless Signals | Network+ Exam Cram: Wireless Networking | Pearson IT Certification. 2017. Factors Affecting Wireless Signals | Network+ Exam Cram: Wireless Networking | Pearson IT Certification. [ONLINE] Available at: http://www.pearsonitcertification.com/articles/article.aspx?p=1329709&seqNum=3. [Accessed 5 October 2017].
7. University of New South Wales - UNSW IT - Accessing UniWide. 2017. University of New South Wales - UNSW IT - Accessing UniWide. [ONLINE] Available at: https://www.it.unsw.edu.au/students/uniwide/access.html. [Accessed 5 October 2017].
8. Transmitting Network Data Using Volley | Android Developers . 2017. Transmitting Network Data Using Volley | Android Developers . [ONLINE] Available at: https://developer.android.com/training/volley/index.html. [Accessed 5 October 2017].
9. Forte, A., Shin, S. and Schulzrinne, H. (2017). Improving Layer 3 Handoff Delay in IEEE 802.11 Wireless Networks. [online] http://citeseerx.ist.psu.edu. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.122.4145&rep=rep1&type=pdf [Accessed 8 Oct. 2017].
10. http://www.cs.umd.edu. (2017). An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process*. [online] Available at: http://www.cs.umd.edu/~waa/pubs/handoff-lat-acm.pdf [Accessed 8 Oct. 2017].
11. http://www.cs.umd.edu. (2017). An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process*. [online] Available at: http://www.cs.umd.edu/~waa/pubs/handoff-lat-acm.pdf [Accessed 8 Oct. 2017].
12. WifiInfo | Android Developers . 2017. WifiInfo | Android Developers . [ONLINE] Available at: https://developer.android.com/reference/android/net/wifi/WifiInfo.html. [Accessed 10 October 2017].
13. http://www.cs.umd.edu. (2017). An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process*. [online] Available at: http://www.cs.umd.edu/~waa/pubs/handoff-lat-acm.pdf [Accessed 10 Oct. 2017].
14. Transmitting Network Data Using Volley | Android Developers . 2017. Transmitting Network Data Using Volley | Android Developers . [ONLINE] Available at: https://developer.android.com/training/volley/index.html. [Accessed 11 October 2017].
15. WifiManager | Android Developers . 2017. WifiManager | Android Developers . [ONLINE] Available at: https://developer.android.com/reference/android/net/wifi/WifiManager.html. [Accessed 11 October 2017].
16. WifiManager | Android Developers . 2017. WifiManager | Android Developers . [ONLINE] Available at: https://developer.android.com/reference/android/net/wifi/WifiManager.html. [Accessed 12 October 2017].
17. Packet Capture - Android Apps on Google Play. 2017. Packet Capture - Android Apps on Google Play. [ONLINE] Available at: https://play.google.com/store/apps/details?id=app.greyshirts.sslcapture&hl=en. [Accessed 14 October 2017].
18. Wireshark · Go Deep.. 2017. Wireshark · Go Deep.. [ONLINE] Available at: https://www.wireshark.org/. [Accessed 14 October 2017].
19. 129.94.8 - edu.au - Australia - Unsw - Search IP addresses. 2017. 129.94.8 - edu.au - Australia - Unsw - Search IP addresses. [ONLINE] Available at: https://db-ip.com/all/129.94.8. [Accessed 15 October 2017].
20. http://www.cs.umd.edu. (2017). An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process*. [online] Available at: http://www.cs.umd.edu/~waa/pubs/handoff-lat-acm.pdf [Accessed 15 Oct. 2017].
21. Forte, A., Shin, S. and Schulzrinne, H. (2017). Improving Layer 3 Handoff Delay in IEEE 802.11 Wireless Networks. [online] http://citeseerx.ist.psu.edu. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.122.4145&rep=rep1&type=pdf [Accessed 15 Oct. 2017].