Final Report

Topic: Utilising Smartphone Application Data
Student: Wong San Yu (3035273022)
Supervisor: Dr T.W. Chim
2nd Examiner: Dr S.M. Yiu
Submission Date: 14th April, 2019
Abstract

Waiting for buses have become easier with bus companies introducing their own smartphone applications (apps) for passengers to check when do the buses arrive. However, apps from different companies do not read each other’s data so that passengers have to open two or more apps for checking bus arrival times. That could lead to undesired results such as missing a bus when opening another app to check arrival time. This paper describes the research and development process of an Android app that integrates bus arrival time data from different bus companies. The goal of the app is to retrieve arrival time data from different servers and show them to user in an all-in-one manner. The methods to retrieve the time data for Kowloon Motor Bus, Long Win Bus, New World First Bus and Citybus have been found and integrated into the app. In the future, this app might grow onto another level such as gathering all available public transport arrival time data and make users’ life easier.
# Table of Contents

**Abstract** ................................................................................................................................. 2

**List of Figures** .......................................................................................................................... 5

1 **Introduction** .......................................................................................................................... 7
   1.1 *Current Problem* .................................................................................................................... 7
   1.2 *Project Background* ............................................................................................................... 8
   1.3 *Related Works* ....................................................................................................................... 8
   1.4 *Route Map of the Project* ........................................................................................................ 9
   1.5 *Outline of the Progress Report* ............................................................................................... 10
   1.6 *Project Objective* ................................................................................................................. 10
   1.7 *Scope* .................................................................................................................................. 10

2 **Methodology** .......................................................................................................................... 11
   2.1 *Network Tracking* ................................................................................................................. 11
      2.1.1 *Tracking on KMB’s ETA Data* ............................................................................................. 12
      2.1.2 *Tracking on Citybus/NWFB’s ETA Data* ............................................................................ 14
   2.2 *Reverse Engineering* ............................................................................................................ 14
   2.3 *Project Timeline* .................................................................................................................. 15
      2.3.1 *Network Tracking on Kowloon Motor Bus’s Application* .................................................. 15
      2.3.2 *Testing the Results from KMB Tracking* ......................................................................... 18
      2.3.3 *Network Tracking on Citybus/NWFB’s Website* ............................................................... 19
      2.3.3.1 *Breakdown of Citybus/NWFB’s Website* ...................................................................... 20
      2.3.4 *Testing the APIs from Citybus/NWFB Website* ............................................................... 23
      2.3.5 *Reverse Engineering the Citybus/NWFB app* .............................................................. 26
      2.3.6 *Checking the APIs found from Citybus/NWFB app* ....................................................... 28

3 **Application** ............................................................................................................................ 35
   3.1 *Overview* ............................................................................................................................ 35
3.2 Contents of the Application ................................................................. 36
  3.2.1 Select Route Page ........................................................................ 36
  3.2.2 Select Destination Page ............................................................... 38
  3.2.3 Select Variant Page ...................................................................... 39
  3.2.4 Select Stop Page .......................................................................... 40
  3.2.5 Show ETA page ........................................................................... 41
3.3 Flowchart of the Application ............................................................ 43

4 Future Improvements ........................................................................... 44
  4.1 Adding new functionalities to ETA Combined .................................. 44
  4.2 Building ETA Combined for iOS ..................................................... 44

5 Difficulties and Challenges ................................................................. 45
  5.1 Difficulties ..................................................................................... 45
  5.2 Challenges .................................................................................... 45

6 Conclusion ......................................................................................... 46

Acknowledgments .................................................................................. 47

References ............................................................................................. 48
List of Figures

Figure 1. Users have to switch between NWFB (left) and KMB (right)......................... 7
Figure 2. Bus companies use standalone apps to provide ETA data.................................. 8
Figure 3. The KMB ETA searching site (left) and Web Inspector (right) ......................... 12
Figure 4. The Android virtual machine (left) and Wireshark interface (right)............... 13
Figure 5. Developer Tools in web browser was used to track Citybus/NWFB's website...... 14
Figure 6. Try to sniff on the requests and responses sent to and received from KMB server.
The right pane in the Web Inspector displays the request and response information......... 15
Figure 7. API for retrieving bus stops in a route with bound specified.......................... 16
Figure 8. Tracking result of the KMB app using Wireshark, showing the API for retrieving
actual ETA .............................................................................................................. 17
Figure 9. Retrieving all stops for 1A towards Tsim Sha Tsui ........................................ 18
Figure 10. Showing service types for 1A Tsim Sha Tsui bound ..................................... 18
Figure 11. Retrieving actual ETA at stop no.7 for 1A Tsim Sha Tsui bound ...................... 19
Figure 12. The Citybus/NWFB website (left) and corresponding tracking results (right).... 19
Figure 13. Rendered response from server after searching for route ................................ 20
Figure 14. Routing variation is shown ........................................................................... 21
Figure 15. The page shown to user (left) and the actual response from server (right)........ 23
Figure 16. Picking a random previous cookie .................................................................. 23
Figure 17. Retrieving directions of route B5 .................................................................... 24
Figure 18. Routing was found for B5 HZMB direction .................................................. 24
Figure 19. Response from server after searching B5 normal routing ......................... 25
Figure 20. The response was not "OK". That indicated the query was not successful .......... 25
Figure 21. ETA could not be shown. Query failed ......................................................... 25
Figure 22. Example of some code from the decompiled Citybus/NWFB app................. 26
Figure 23. Searching the codes with https://mobile.nwstbus.com.hk .......................... 27
Figure 24. Searching for usage of variable f3578c......................................................... 28
Figure 25. Part of the code to generate syscode ............................................................. 29
Figure 26. Response when rno is not specified. All routes of Citybus and NWFB are returned.
.......................................................................................................................... 30
Figure 27. Response with rno = 5B. Only route 5B is returned as this is exact match ....... 30
Figure 28. Response with rno = 78. All routes starting with "78" are returned ............... 30
Figure 29. The part where parameter id comes from ..................................................... 31
Figure 30. Showing the 6 variations of 788 Central bound..........................31
Figure 31. Compare parameter info with result from API 2. .........................32
Figure 32. Stops list for 788 Central bound, normal routing.........................33
Figure 33. Extract of stop list of 788 Central bound. ..................................34
Figure 34. Query result of 788 Central bound at Pottinger Street ..................34
Figure 35. ETA Combined running on an Android emulator with Android 6.0 ........35
Figure 36. Select route page of ETA Combined. ........................................36
Figure 37. Language menu. ........................................................................37
Figure 38. Toast message shown after choosing language............................37
Figure 39. Select destination page for NWFB route 14. ...............................38
Figure 40. Example of variants of NWFB 14, Stanley bound.........................39
Figure 41. Stops list for NWFB 14 via Stanley Village and Stanley Fort to Stanley Plaza....40
Figure 42. ETA of NWFB 14 at Shan Tsui Court.........................................41
Figure 43. ETA page for jointly operated route 307 towards Tai Po. ETAs are combined and sorted. ......................................................................................42
Figure 44. Flowchart of ETA Combined..........................................................43
1 Introduction

Smartphone usage have become more and more widespread in the past decade. Many smartphone applications are popular among users because they provide useful data. However, most of the companies do not provide open data and it is not convenient to get data from several applications in a short time. This project aims at relieving this situation by exploring methods to make data retrieval from multiple applications easier. In the project, the focus will be on data of Estimated Time of Arrival of buses, as it is a problem often faced by general public.

1.1 Current Problem

The everyday commute has always been a problem for working people and students. In the past, passengers have to wait for unknown time at stations for buses. With the introduction of Estimated Time of Arrival (ETA) of buses in bus companies’ apps, commuters know when does the bus come in advance and that helps in time management. However, people often have to enquire ETAs from different apps for different bus routes. Figure 1 serves an example. A user checks ETA for a cross harbor route jointly operated by KMB and New World First Bus. Then the NWFB app (left) says it is now non-NWFB hours. Therefore, the user has to open KMB app (right) and check ETA again. Changing apps back and forth could be annoying. A universal tool gathering ETAs from different bus companies will be favourable for commuters.

Figure 1. Users have to switch between NWFB (left) and KMB (right)
1.2 Project Background

Although users can get ETAs from the bus companies’ apps, the data are not open to public. Bus companies claim that the ETA data are their private property [1] so that they will not provide any means (e.g. tools accessing real time data and/or static data) for users to retrieve ETA data other than using their app. However, the bus company apps are often slow and in some sense not very user friendly. Figure 2 shows three different apps from three bus companies. They have different layouts and UI/UX. Users have to learn how to use the three apps, which is not user-friendly. As a heavy user of bus services, a transportation enthusiast and CS student, the abovementioned situation of having to switch apps always appear in daily life. Therefore this project is an interesting one which is a way to put knowledge gained in university into practice and benefit the general public.

![Figure 2. Bus companies use standalone apps to provide ETA data.](image)

1.3 Related Works

There are few integrated transportation applications available on the market. They provide one-stop route suggestions and mixed real time and non-real time ETA. However, as a user, non-real time ETA is not preferred. Following is a general comparison of those apps:


  Google Maps is a popular way-finding app which provides route suggestions and basic schedules of bus routes and MTR lines. However, the information provided is often not accurate and it does not provide any real time ETA data for the user.
- Citymapper ([https://citymapper.com/hong-kong](https://citymapper.com/hong-kong))

Citymapper provides route finding function in several locations over the world including Hong Kong. It integrates most of the public transport in Hong Kong including red minibuses into their service. However, users can only get real time ETA for KMB and LWB routes. For other bus companies, the app still provides ETA but they are only calculated based on the official schedule from bus companies. The calculated ETA are not accurate.

- Moovit ([https://moovit.com](https://moovit.com))

Moovit is a service similar to Citymapper. Moovit also provides route suggestions for most kinds of public transportation in Hong Kong. It also supports push notifications to alert users on incidents. The ETA system is the same as Citymapper with only KMB and LWB routes having real time ETA.

This project is not going to compete with the above apps, but try to improve their weakness, which is not providing real time ETA for companies other than KMB and LWB.

Other than mobile apps, there exists a bot (a program that responds to messages user send to it automatically) on the instant messaging app Telegram which does similar work as this project. As the bot works, there must exist some methods to retrieve ETA data from Citybus/NWFB which the above apps do not implement. The bot can be accessed at [https://t.me/HKBusesETABot](https://t.me/HKBusesETABot). (Need to be a Telegram user)

### 1.4 Route Map of the Project

For this project, the main point is to explore how do the existing applications get ETA data from bus companies. Therefore, the first thing to do will be studying the behaviour of the applications. After that, the focus will turn to finding techniques to stimulate the applications and integrating the techniques into the final product of this project.
1.5 Outline of the Progress Report

This report proceeds as follows. First, the methodology is presented in chapter 2. That enables better understanding of the network tracking and reverse engineering techniques being used in the project. Then the process of finding the methods to get ETA data from KMB and Citybus/NWFB’s servers and developing the Android app for real world usage are presented in chapter 3. Next, some difficulties encountered and possible plans for expansions are discussed in chapters 4 and 6.

1.6 Project Objective

This project aims at finding ways to centralise the process of ETA data request and retrieval from different bus companies within one single application. As the ETA data are not public, different approaches have been tried in order to retrieve data from companies’ servers. An app is built for users to view real time bus ETAs retrieved from different companies, eliminating the need of switching apps to get real time ETA. This is a big step forward as this is a new channel for users to get required data easily, which makes their life easier.

1.7 Scope

This project manipulates ETA data from bus companies’ apps and combines them into one single application. As the focus shall be on the techniques to get data, not the breadth of data, only ETA data from Kowloon Motor Bus (KMB), Long Win Bus (LWB), New World First Bus (NWFB) and Citybus (CTB) are covered as there are actually only two data sources. Other public transport are not covered. The main target of this project is to develop a tool for checking ETA easily. The product focuses more on functionality than the user interface and user experience.
2 Methodology

For this project, the underlying request/response cycle of bus companies’ apps have to be understood first with the help of some existing tools. There are two main approaches to understand the process, namely Network Tracking and Reverse Engineering.

2.1 Network Tracking

In the early stage of the project, network tracking is first tried. Network Tracking means studying the real time activity of a computer network [2]. The apps must send and receive data packets to/from bus company servers in order to get the ETA data. If the content of the data transmission can be seen, we can then study the logic behind the connections, send similar requests to bus companies and get the ETA data needed.

For network tracking, two major tools are used. First one is the Developer Tool in browsers and the other is the software Wireshark. Developer Tool built in the network browsers such as Apple Safari and Google Chrome provide simple but necessary tools for users to study network activity. For example, users can view what resources have been downloaded and which request sent to server they correspond to. The Developer Tool is used to do simple checks on network activity from the web browser. Developer Tool is used because no separate installation is needed and it provide essential functionality. User can quickly analyse network activity relating to web pages.

The software Wireshark is used when we are not working in a web browser. Wireshark is a free open source software for analysing data packets in a computer network [3]. Wireshark monitors all the network activity going in and out of the network interface card, where the computer physically connect to the internet. With the help of Wireshark, data packets transmitted by apps can then be traced. Contents and destinations of the packets can then be explored and used in future work.

The tracking on smartphone apps runs on a computer with Android virtual machine installed at first. Android is used because it is free, open source and easy to obtain. Later, BlueStacks software is used to run apps because the Android VM is too slow. BlueStacks is a software which runs Android apps on a PC or Mac. It is chosen because it is free and it is an easy way to run mobile app on computer. Bus companies’ apps are
installed in BlueStacks and their network activity will be logged by Wireshark on the host machine. The logged data can then be studied later.

2.1.1 Tracking on KMB’s ETA Data

As the KMB app works with an offline database of bus routes and stops, we first tried tracking on the web page provided by KMB for passengers to check ETA data to check how to obtain those data. Tracking was done using built-in Developer Tool in Apple Safari (called Web Inspector) for simplicity (see Figure 3, which shows the interface of Web Inspector). When actions were done on the web page, e.g. entering desired route to check ETA, choosing the bus stop you wish to get ETA, the Tool logged what request had been sent to the server and what did the server returned. From tracking on the website, the important function for querying bus stop list of a route was found.

*Figure 3. The KMB ETA searching site (left) and Web Inspector (right)*
The function for querying actual ETA was also found, but it sent HTTP POST requests to server which the contents were not readable as it is the nature of POST requests. Therefore tracking has to be applied also on the KMB app (see Figure 4, which shows the tracking on KMB app) to investigate more on how to get ETA. By recording the network activity as soon as ETA was queried in the KMB app, the remaining function of retrieving actual ETA was found.

Figure 4. The Android virtual machine (left) and Wireshark interface (right)
2.1.2 Tracking on Citybus/NWFB’s ETA Data

Tracking Citybus/NWFB’s data was done in a similar way of what have been done on KMB’s one. Developer Tools in web browser (see Figure 5) was used and the requests and responses were examined in the Developer Tools window. It was different from KMB’s one because all the query parameters were very explicit and values required could be found in the responses by reading the URL.

![Figure 5. Developer Tools in web browser was used to track Citybus/NWFB's website](image)

2.2 Reverse Engineering

Reverse Engineering is used when network tracking does not work, or the results gathered are not sufficient. Reverse Engineering means cracking down the application back to lines of code. Android applications, in the form of APK files, are in fact simple compressed ZIP files containing binary codes. Binary codes can then be converted back to source codes for reading or modification. With the source code, the ultimate logic of the app can be understood better. Non-public APIs might also be found. With the API, we can directly contact with bus companies’ servers. Binary codes can be converted to different other forms using tool like `apktool` and `dex2jar` as they are free tools and provide sufficient functionalities. After understanding the program logic, the ETA data request/retrieve process can be emulated for later use.
2.3 Project Timeline

In this section, the progress history of the project will be shown in chronological order.

2.3.1 Network Tracking on Kowloon Motor Bus’s Application

In the beginning of the project, network tracking on KMB’s app and website was done. Some APIs were found. Figure 6 shows the interface of Web Inspector of Apple’s Safari web browser.

![Figure 6. Try to sniff on the requests and responses sent to and received from KMB server. The right pane in the Web Inspector displays the request and response information.](image)

These APIs can be accessed to obtain several types of data. All of the APIs return JSON objects, which are widely used in web applications as they are easy to manipulate, as response. Most of the APIs are obtained from the KMB website. Types of data available are listed below:
1. Retrieving all stops in a route, specifying the bound

![Summary]

**Figure 7. API for retrieving bus stops in a route with bound specified**

By reading the URL part in Figure 7, it can be seen that in this API, request is sent to [http://search.kmb.hk/KMBWebSite/Function/FunctionRequest.ashx](http://search.kmb.hk/KMBWebSite/Function/FunctionRequest.ashx) (“base URL”) with parameter “action” equals “getstops”. “Route” corresponds to the requested route, “bound” means direction and “serviceType” is related to some special departure types, for example, special routing variations in the morning peak.

2. Retrieving special departures information of a route

The base URL is same as API 1 in Figure 7. The parameter “action” changes from “getstops” to “getSpecialRoute”. “serviceType” is not needed.

3. Retrieving timetable of a route

Still the same base URL from Figure 7, with “action” changing to “getschedule”, together with “route” and “bound” equals the same route number and bound number.

4. Retrieving bus-bus interchange information

Same as API 3, with “action” equals “getbbiforroute”.

5. Retrieving announcements related to the route

Same as API 3, with “action” equals “getAnnounce”.

6. Retrieving directions information of a route

Same Base URL, with “action” equals “getroutebound” together with “route” equals to the desired route number.
7. Retrieving ETA of a bus route at a specific stop

There is difference with the above 6 APIs. The request is sent to http://etav3.kmb.hk instead. This is obtained by checking the requests and responses from the KMB app. This request cannot be found in the KMB website.

In Figure 8, it can be seen that there are 7 useful parameters sent to the server:

- “action” = “geteta” tells server to fetch ETA
- “lang” defines language in the response
- “route” selects a route number
- “bound” selects a direction
- “stop” is some number unique to a bus stop
- “stop_seq” is the stopping sequence number of that stop in the route
- “serviceType” is related to special departures

Among them, APIs 1,2 and 7 are the top 3 important APIs relating to the project. The stop list API is used to help specifying which stop we would like to get ETA. Then we can get ETA using API 7. The special departures list helps to ensure all the possible stops are in coverage. By changing parameters in the URL, different results are expected from the server when the requests are sent.

![Figure 8. Tracking result of the KMB app using Wireshark, showing the API for retrieving actual ETA.](image)
2.3.2 Testing the Results from KMB Tracking

The abovementioned 3 important APIs in chapter 2.3.1 were tested and the result was positive. Desired data can now be retrieved without accessing KMB’s app or website. Using route 1A as example:

1. Retrieving all stops for Tsim Sha Tsui bound

Corresponding to the parameters breakdown in 2.3.1, the testing request was sent to http://search.kmb.hk/KMBWebSite/Function/FunctionRequest.ashx?action=getstops &route=1A&bound=1&serviceType=1. Bound number is either 1 or 2 as routes have 2 directions. ServiceType is usually 1 for normal routes and it changes for special departures. Figure 9 shows that desired results were returned.

![Figure 9. Retrieving all stops for 1A towards Tsim Sha Tsui](image)

2. Retrieving special departures

In Figure 10, a “getSpecialRoute” request was sent. In the response, “ServiceType” value was found and that would correspond to special departures. Special departures have “ServiceType” value not equal to 1. Value “CountSpecial” gives the total number of special departures.

![Figure 10. Showing service types for 1A Tsim Sha Tsui bound](image)
3. Retrieving ETA of bus stop no. 7 (example) for Tsim Sha Tsui bound

For this query, what should be put in “stop” parameter was not clear. Therefore, try to skip that parameter first:

![Screen capture of search.kmb.hk/KMBWebSite/index.aspx?lang=en](image)

From Figure 11, ETA data could still be retrieved without specifying “stop” parameter.

The test results have showed that the APIs are providing expected results so that they are usable in the project.

### 2.3.3 Network Tracking on Citybus/NWFB’s Website

The general working logic of Citybus/NWFB’s website for querying ETA was found, also by network tracking.

![Screen capture of Citybus/NWFB website](image)

Figure 12 above shows the structure of the response sent from Citybus/NWFB’s server. It shows that the ETA data is presented in the form of a web page (which is HTML, file
extension not shown in the figure). That is different from KMB which uses JSON as response format. The logic of retrieving ETA is as follow:

1. User enters desired route number
2. Server returns with route bounds and variations of routing to user
3. User picks a direction and variation of routing
4. Server returns with bus stop list of that direction and routing
5. User then picks the desired bus stop
6. ETA of that specific bus route and bus stop is shown

2.3.3.1 Breakdown of Citybus/NWFB’s Website

This subsection explains the APIs found in Citybus/NWFB’s website, using route A21 between Airport and Hung Hom Station as example.

1. Retrieving route bounds

A request containing desired route is sent to the server.


In the request, parameter skey is the route entered into the searching box. Parameter ssid is one of the cookies saved in user’s computer and will be used throughout the whole querying process. Parameter sysid is a fixed value. The server will then respond with a HTML file which shows buttons for choosing route bounds, as shown in Figure 13.

![Figure 13. Rendered response from server after searching for route.]

2. Retrieving variations of routing

A request asking for routing variance is sent to server.

In the request, it can be seen that *ssid* is the same with API 1 above. It is reasonably to guess that the Citybus/NWFB server uses this cookie ID to save what did the user search. *lid* specifies bounds with 0/1 for different directions. Figure 14 shows the rendered response from server.

![Open/Close route input](image)

**Figure 14.** Routing variation is shown.

3. Retrieving bus stops in specified routing

After selecting routing, the server returns with the list of bus stops in that routing, just like the KMB one. For the website, there are two requests doing similar function, the one with simpler response would be used.

```
https://mobile.nwstbus.com.hk/nwp3/showsingleroutestops2.php?info=1%7C%7CG%7C%7CA21-HUH-1%7C%7CNormal%20Routeing%7C%7CCTB***A21-HUH-1***1***20***10661***I&l=1&ssid=5bff8b1254156&sysid=6
```

In the request, the following parameters are present:

1. info: 1||G||A21-HUH-1||Normal Routeing||CTB***A21-HUH-1***1***20***10661***I
2. l: 1
3. ssid: 5bff8b1254156
4. sysid: 6

Again, *ssid* is the cookie ID since API 1. The long string in *info* points to the desired routing. The response looks like this:
In the `addstoponmap()` function, the number in the first parameter is the stop ID for a specific bus stop. Followed by coordinates in longitude and latitude, and some other parameters specifying stop number, stop name and which routing it belongs to. These parameters are used in retrieving the actual ETA.

4. Retrieving ETA from server

After retrieving all the required parameters, the final step is requesting for actual ETA. In this step, two requests are sent in order to get the actual time.

```
https://mobile.nwstbus.com.hk/nwp3/auto_af08b1e236b44c28ac962835e7e674ce.php?info=001559||A21-HUH-1||8||I&ssid=5bff8b1254156&sysid=12&_=1543473940341
```

This is the first request and it is for the server to log the details of the ETA query. The file name (auto_xxxxxx.php) is different every time starting a new search from scratch (i.e. accessing https://mobile.nwstbus.com.hk) and the naming method is not found. This request returns an OK message if the parameters are valid. In the `info` part, the stop ID, routing name, stop number are sent to the server. Again, `ssid` is the cookie ID used since the first step.

```
```
This is the second request. The file name is similar to the first one but not the same. Also, the naming method is not found and the name will change. ssid is present again and the server will send some HTML code containing the actual ETA in response, as shown in Figure 15. In the Figure, the right half is the rendered result of the HTML code sent from the server. It corresponds to the part showing ETA to users on the actual website (left half).

![Figure 15. The page shown to user (left) and the actual response from server (right)](image)

### 2.3.4 Testing the APIs from Citybus/NWFB Website

In the testing part, cookie ID was needed as it was required in the request parameters. For simplicity, we tried to randomly pick one previous cookie first. Figure 16 shows some cookie saved in the computer before. Try to use the third one, 5bff892e280c2, first.

![Figure 16. Picking a random previous cookie](image)
First step was retrieving route bounds. Route B5 between Hong Kong-Zhuhai-Macau Bridge Hong Kong Port and Sunny Bay MTR Station was taken as example. A request was sent to the server:


And the server returned with expected response, shown in Figure 17. Correct directions were found.

![Figure 17. Retrieving directions of route B5](https://mobile.nwstbus.com.hk/nwp3/routesearch.php?rtype=&skey=B5&l=1&savecookie=1&ssid=5bff892e280c2&sysid=3)

Figure 17. Retrieving directions of route B5

Then we tried to retrieve bus stop list:


Routing was found, shown in Figure 18.

![Figure 18. Routing was found for B5 HZMB direction](https://mobile.nwstbus.com.hk/nwp3/getvariance.php?lid=0&l=1&cur=0&rdv=&bound=&ssid=5bff892e280c2&sysid=4)

Figure 18. Routing was found for B5 HZMB direction

After selecting routing, next step was to retrieve bus stops list. The info part was found in the HTML code in the response of last step. Response shown in Figure 19.
Final step was to retrieve ETA. However, it seemed that the session has expired already and there was no result. The two requests sent:


**Figure 19. Response from server after searching B5 normal routing**

ETA was **NOT** shown, as in Figure 20 and 21.


**Figure 20. The response was not "OK". That indicated the query was not successful.**

Estimated Time of Arrival is currently not available. Please refer to timetable.

**Figure 21. ETA could not be shown. Query failed.**
Therefore, it is reasonable to claim that we have succeeded in most of the steps in querying Citybus/NWFB’s ETA except the last part where the actual ETA was queried.

2.3.5 Reverse Engineering the Citybus/NWFB app

As the API found on Citybus/NWFB’s website is not usable to get the actual time data, another method has to be trialed. Therefore, we tried to break down the app and find whether some other APIs exist.

In the beginning, decompiling the app using apktool and dex2jar was proposed. Later, a new tool, JADX, was found and there exists an online decompiler using JADX [5]. JADX is better because it provides Java source code after decompiling.

Figure 22. Example of some code from the decompiled Citybus/NWFB app

From Figure 22, it can be seen that the codes are scrambled. The file names of the java files are also in a unreadable way (see the left part). Following the design of the website, it is proposed that the URL that the app request data from should be similar to those on the website. So, try to search with the base URL first:
Figure 23. Searching the codes with https://mobile.nwstbus.com.hk

As seen in Figure 23, there are several URLs starting with base https://mobile.nwstbus.com.hk. Strings f3578c and f3581f are likely to be the targets because of its path name (api6 and etaapi).

Start with f3578c, we repeatedly find for codes that utilize this base URL, as shown in Figure 24, and found the following APIs which looks similar to those found on Citybus/NWFB’s website or looks useful by its name:

2.3.6 Checking the APIs found from Citybus/NWFB app

After finding the APIs, testing is needed to discover how they work. We can have confidence in these APIs for their usability because they are found in a production app. Upon further searching in the scrambled codes, we confirmed that the four APIs found is sufficient for querying ETA for a route at a particular bus stop.

**Something in common – “syscode”**

There exists a parameter called *syscode*, which is needed in every request to the bus company’s server. During exploration in the codes, there are some more parameters like *syscode2* and *syscode3* sent together to the server. After testing, only *syscode* is essential for the requests. Syscode is constructed with two parts. The first part is some numbers generated with current system time and random integer within 1000. The second part uses result of the first part. A phrase “firstbusmwymwy” is added to the numbers’ tail, and the whole string is then hashed with MD5. In another way to express it, *syscode* = (6 digits number generated from current time) + (4 digits random integer) + MD5(first two numbers added with “firstbusmwymwy”).
Figure 25. Part of the code to generate syscode.

Figure 25 shows part of the actual code for generating *syscode*. This *syscode* generated is added to all of the requests mentioned below as a parameter.

**API 1 – Retrieving route list**

For this API, request is pointed to 
[https://mobile.nwstbus.com.hk/api6/getroutelist2.php](https://mobile.nwstbus.com.hk/api6/getroutelist2.php). As the filename suggests, this is for retrieving the route list. There are two parameters sent to the server together:

1. **rno** – route number

This parameter can be optional. If route number is specified, it will return routes starting with rno. If **rno** is not specified, all routes operated by Citybus or NWFB currently in the database will be shown. Figures 26, 27 and 28 show the examples of not specifying **rno**, **rno** = 5B and **rno** = 78.
Utilising Smartphone Application Data

Final Report

CTBHI1FEV12Happy Valley (Upper)→Felix Villas,21H→Felix Villas,102291111

CTBHI1FUJ22Felix Villas→Happy Valley (Upper)→21H→Happy Valley (Upper)→102279111

CTBHI1PCEBH1Wong Nai Chung Road (Broadwood Road)→Central,21H→Central,103981103

CTBHI1PHP11Happy Valley (Upper)→21HP→Happy Valley (Upper)→109971103

NWFBi21GKL22Central (Macau Ferry)→Grand Promenade,412→Grand Promenade,103111111

NWFBi21MAF122Grand Promenade→Central (Macau Ferry)→212→Central (Macau Ferry)→103821111

NWFBi21AIHEC12Yiu Tung Estate→Wan Chai North,412A→Wan Chai North,110331111

NWFBi21AYTIT22Wan Chai North→Yiu Tung Estate,412A→Yiu Tung Estate,103411111

NWFBi21HEC12Shau Kei Wan→Wan Chai North,412X→Wan Chai North,110041111

NWFBi21XOE12Wan Chai North→Shau Kei Wan→Alidrick Bay,412X→Shau Kei Wan,110381111

NWFBi21XOE12Wan Chai North→Shau Kei Wan→Alidrick Bay,412X→Shau Kei Wan,110381111

NWFBi21AFTT12Wan Chai,413A→Felix Villas,103881101

NWFBi21AFTT12Felix Villas→Central (Star Ferry)→413A→Central (Star Ferry)→110381111

NWFBi21WS322Wah Fu (South)→Central,110411111

NWFBi21WSF522Central→Wah Fu (South)→Central,110381111

NWFBi24XiFWSF12Wah Fu (South)→Central (Exchange Square)→441X→Central (Exchange Square)→110391111

Figure 26. Response when rno is not specified. All routes of Citybus and NWFB are returned.

Figure 27. Response with rno = 5B. Only route 5B is returned as this is exact match.

Figure 28. Response with rno = 78. All routes starting with "78" are returned.
2. $l$ – language

Controlling the language of the response. 0 for Traditional Chinese, 1 for English and 2 for Simplified Chinese.

**API 2 – Retrieving variants of a route**

Requests are pointed to [https://mobile.nwstbus.com.hk/api6/getvariantlist.php](https://mobile.nwstbus.com.hk/api6/getvariantlist.php) for this API. It is responsible for showing the different versions of routings for a bus route in the specified bound. In this step, only one parameter, $id$, is needed. This parameter comes from the result of API 1, shown in Figure 29. Parameter $l$ can be omitted, and Traditional Chinese results will be returned by default.

![Figure 29. The part where parameter id comes from.](image)

Figure 30 below shows the a search for variants for route 788 bound for Central.

![Figure 30. Showing the 6 variations of 788 Central bound.](image)

**API 3 – Retrieving bus stops list for a specified route and direction**

After having the results from API 2, the stops list for a specific variant of a route can be retrieved. Again, only one essential parameter, $info$, is needed. Request is sent to [https://mobile.nwstbus.com.hk/api6/ppstoplist.php](https://mobile.nwstbus.com.hk/api6/ppstoplist.php). $info$ also comes from the response of API 2. However, it is different from going from API 1 to API 2. Direct copying from the response does not work. The formatting has to be changed.
For the content of parameter *info*, it always starts with “0|*|”. The following part derives from result of API 2. Following is a valid sample of *info*:

“0|*|CTB||788-MAF-1||1||13||10133||O”. Compare with result from API 2 in Figure 31:

![Figure 31. Compare parameter *info* with result from API 2.](image)

After putting them together, it is clear that the asterisks (***)) have to be changed into || as delimiter. In the whole string *info*, each part between delimiter || have their own meanings. The slot containing “CTB” specifies the operating company. Slot for “788-MAF-1” corresponds to something called “RDV” in the Citybus/NWFB system and RDV possibly means Route-Destination-Variant. Every route has at least one RDV of their own. The next two integers represent the sequence number of the first and last stop to show. For example, having 1 and 10 will show ten stops, numbers 1 to 10; having 2 and 6 will only show stops number 2 to 6, five stops in total. The second last one is like an ID for a specific routing. It is different for every route, every variant. The very last character specifies Inbound (I) or Outbound (O).

With a valid *info* string, request is sent to the server and stop list will return as response, shown in Figure 32.
Figure 32. Stops list for 788 Central bound, normal routing.

The result might look complicated, but it is not. Each line corresponds to a stop. Again, each line has || as delimiter. The fourth slot is the stop ID, unique for each bus stop, with the fifth slot differentiating which subgroup this route belongs to in that bus stop. The sixth and seventh are the coordinates (latitude and longitude) of the stop. Other information does not have much use in querying ETA.

API 4 – Retrieving the actual ETA

With the stop sequence number and stop ID from API 3, all the necessary information for querying the actual ETA data have been collected. Then we can get the real ETA from the server. To get the time data, a few parameters have to be added to the request sending to https://mobile.nwstbus.com.hk/api6/getnextbus2.php. They are:

1. service_no – route number
2. stopseq – stop sequence number
3. stopid – stop ID
4. rdv – route variant
5. bound – direction
Utilising Smartphone Application Data

Final Report

Figure 33. Extract of stop list of 788 Central bound.

Continue with the route 788 example above. Say we would like to query ETA at Pottinger Street. In this case, the parameters would be as follows, with reference to Figure 33 above:

1. service_no = 788
2. stopseq = 10
3. stopid = 1034
4. rdv = 788-MAF-1
5. bound = 0

Figure 34. Query result of 788 Central bound at Pottinger Street.

Figure 34 shows the query result with the above parameters. There are a lot of information in the response. The numbers in red boxes are the ETAs. Blue boxes contain destination of that departure, as the buses might only operate until some stop before the terminus. Green boxes contain the distance of the bus from the queried stop. These can be extracted using || as delimiter again.

Until this stage, the working methods to retrieve actual ETA from KMB/LWB and Citybus/NWFB servers have been found. The showcase application can then be built.
3 Application

In this section, the showcase application will be explained.

3.1 Overview

The showcase application for this project is an Android app. Android app is chosen because Android apps are easy to write, and are usable on a lot of devices. The app is called “ETA Combined” and is built with Android SDK API 28 (Android 9.0 Pie) as it is the latest and supports latest functions. The app runs on Android API level 23 (Android 6.0 Marshmallow) or above.

Figure 35. ETA Combined running on an Android emulator with Android 6.0
3.2 Contents of the Application

In this section, the components of the app will be explained. All the demonstration below are on Android emulator running Android 9.0 Pie.

3.2.1 Select Route Page

This is the startup page of the application. When it is first loaded (e.g. starting the app freshly), route database in JSON format is downloaded from a server (currently the FYP website server for this project). Routes are then listed in ascending order, in a string-sorting manner. Company logos are also shown as there are bus routes having same route number but operated by different companies. Figure 36 shows a snapshot of the select route page. Users press on the desired route to navigate to next page.

![Select route page of ETA Combined.](image)

Figure 36. Select route page of ETA Combined.
Users can choose language of data displayed on this page, using the menu on the Action Bar at the top. Traditional Chinese and English are supported. Figure 37 shows the menu for choosing language. Traditional Chinese is default language.

![Figure 37. Language menu.](image)

After choosing language, a Toast message will be shown, indicating the new language chosen. Figure 38 shows the message.

![Figure 38. Toast message shown after choosing language.](image)
3.2.2 Select Destination Page

After selecting the route, users are brought to a new page for selecting destination. The destinations list is obtained from bus companies’ servers. Therefore, if there are any changes in the termini, this app can always get the latest destinations of a route. Figure 39 is an example, showing the two destinations, Grand Promenade in Sai Wan Ho and Stanley Fort/Ma Hang, for NWFB route 14. Users can click on one of the destinations to navigate to next step.

![Select destination page for NWFB route 14.](image)

*Figure 39. Select destination page for NWFB route 14.*
3.2.3 Select Variant Page

If there exist two or more variants of the direction which the user chose, this page will be shown, letting user to choose the variant. The variant list is also retrieved online so that it is always up to date. Figure 40 shows the variants for NWFB 14 Stanley bound. Users click on one of the variants to see bus stops. If there is only one variant, i.e. no special routing, this page is skipped.

Figure 40. Example of variants of NWFB 14, Stanley bound
3.2.4 Select Stop Page

After choosing variant, or it was skipped, users are presented with the list of bus stops retrieved from bus companies’ servers. Sequence numbers are shown for users to find a stop easier. Figure 41 is the stops list of NWFB 14 for Stanley Plaza, via Stanley Village and Stanley Fort. Users can click on the stop to view ETA at that stop.

![Figure 41. Stops list for NWFB 14 via Stanley Village and Stanley Fort to Stanley Plaza.](image)
3.2.5 Show ETA page

When everything have been selected, users can see when do the buses come at the stop they chose. ETAs are listed in chronological order. Figure 42 shows ETA for NWFB 14 for Stanley Plaza, at Shan Tsui Court stop.

![ETA Screen Screenshot](image)

*Figure 42. ETA of NWFB 14 at Shan Tsui Court*

If the route is a jointly operated one, ETAs retrieved from the two companies are combined and sorted in chronological order. Figure 43 shows Victoria Park stop of route 307, jointly operated by KMB and CTB, bounds for Tai Po Centre. It is clear shown that the ETAs are merged into a sorted list, which is one of the main target of this project.
Figure 43. ETA page for jointly operated route 307 towards Tai Po.

ETAs are combined and sorted.

As seen in Figure 43, the app also supports showing the congestion information mentioned in ETA data (see the first entry). At this page, user can retrieve updated ETA information by clicking the refresh button. When new data has been loaded, a short toast message, “Data loaded” is shown on the screen, as shown in Figure 43 on the previous page. However, the refresh button only send new request for ETA to the bus companies. It does not guarantee that the data returned is different from before, as bus companies update their ETA data every 1-2 minutes.
3.3 Flowchart of the Application

This is a flowchart of ETA Combined. This app serves as a simple tool for users to check ETA easily, especially for jointly operated routes.

![Flowchart of ETA Combined](image)

*Figure 44. Flowchart of ETA Combined.*
4 Future Improvements

As the current version of the application is mainly for showcasing the research results of this project, there are still many rooms for improvement before releasing to public. Some noticed problems and proposed improvements for ETA Combined is discussed in this section.

4.1 Adding new functionalities to ETA Combined

1. Supports route search
   
   Currently, all the supported routes are listed out on the home page. If the user wants to find large route numbers located near the end of the list, the user need to scroll a lot. This is not very user-friendly. Searching function will be implemented later to improve user experience.

2. Include all of the routes from KMB, NWFB and CTB
   
   The number of routes the app currently support is controlled by a file on the server, and the data input is by hand. More routes will be supported after implementing searching function.

3. Include data for New Lantao Bus (NLB)
   
   If the method of retrieving ETA data from NLB servers is found, NLB routes will be added to the database of the app.

4. Support location search
   
   It is better if the app can determine user location and (1) filter out nearby routes and (2) jump to nearest stop in the stops list page.

4.2 Building ETA Combined for iOS

The current version of ETA Combined is written in Java, which is the native coding language on Android. If the app has an iOS version in the future, more people will be benefitted. Converting the app into cross-platform frameworks like React Native is under consideration as this can reduce the development time by eliminating the need to maintain two separate code bases.
5 Difficulties and Challenges

Some difficulties and challenges found during the project is discussed in the section.

5.1 Difficulties

The part of tracking ETA requests and responses cycle for KMB was relatively easy as the APIs are easily obtained and they used JSON format which is convenient to manipulate. However servers of Citybus/NWFB uses different format of response. When writing the app, suitable data structures are needed to store both sets of data.

Also, during the app development process, many Exceptions as run-time errors were found. They were completely new to us. Trial and error in coding was also very frustrating.

5.2 Challenges

After converting APK into source codes, the original naming of variables were changed, a lot of time was used to study the whole program to understand what do the variables store and their usage. That was a very complicated task for the brain, as the variables and functions names were mostly meaningless.

New techniques were needed when writing the app. As a beginner in app development, another great amount of time was used to understand the Android development environment.
6 Conclusion

This project focuses at how to extract and make use of non-public, non-open data from smartphone applications, using ETA data from different apps from bus companies as an example. The project tries to gather the ETA data together in one place in order to benefit user in their daily lives. In this report, the general background was first discussed. Then the two main techniques, Network Tracking and Reverse Engineering were introduced. Then the process of finding way to retrieve ETA and related data for bus companies’ servers was shown. The results were tested. In the end, an Android app to integrate ETA data from the two major bus operator groups was built to facilitate passengers. With this project, passengers have the chance to forget about the hard times when they have to switch between applications to view bus ETA. However this project is small-scaled and it only focus on gathering and integrating data. Further speaking, we hope that more and more applications or companies will open their data to the public so that software developers can make good use of the public data and create great applications that make everyone’s life better.
Acknowledgments

I would like to express my very great appreciation to my supervisor, Dr T.W. Chim, for his kindness and sufficient help whenever I need. He gave lots of valuable and constructive suggestions during the planning stage of the project. During the development of project, he also gave timely help which assisted me a lot.

Also, I would like to thank my friends who helped to test the Android app. As an iPhone user, I could not always test out my app. Since the app focuses on bus arrival time which requires great accuracy, many testing were needed to check if the codes are pointing to correct results. Without their reports of bugs, I believe that the app will still be buggy.

Finally, I wish to thank my parents for their support throughout my study.
References


