Edu-BUS Wi-Fi: An On-Board Wi-Fi Educational System Using a Raspberry Pi

Shamar Ward(✉) and Mechelle Gittens

University of the West Indies Cave Hill Campus, Cave Hill, Barbados
shamar.ward@mycavehill.uwi.edu,
mechelle.gittens@cavehill.uwi.edu

Abstract. Onboard transit systems are commonly used for commutes to and from home, work and other destinations. However, this is generally not productive time with commuters often gazing out of the window. Other systems have proposed a solution to this problem however, they are offline and would require each device on each vehicle to be updated manually. In this paper, we present an onboard educational Wi-Fi system that offers educational content which can be updated remotely as well as free internet access to particular commuters. Also, specific users can be allowed to browse the internet by connecting wirelessly while onboard. Additionally, Edu-Bus can be used as an educational tool, requiring persons to watch an educational video or read some educational content before gaining access to Edu-Bus services such as free Wi-Fi. Our system uses a Raspberry Pi which is converted into a wireless hotspot, as well as an optional USB or Wi-Fi modem to connect to the 3G/4G provider. Additionally, Edu-Bus can be powered directly from an adapter using the vehicle's cigarette lighter outlet or temporarily from its internal battery. We implemented this work on-board a fleet of four (4) buses at the University of The West Indies Cave Hill Campus. We analyzed student connections to the system generated by user activities. Our approach to this problem not only makes Edu-Bus extendable but also introduces the ability to deploy similar systems on varying vehicle types and modes of transportation.

Keywords: Raspberry pi · On-Board Wi-Fi · University campus · Educational content

1 Introduction

Many commuters use transit systems to travel to and from their destinations. For some, their trip may be lengthy in distance. However, others may have shorter trips but due to traffic congestion, they may have a long transit time to their destination. As discussed in the work done by Rahane et al. in [1] commuters may become bored during these journeys. To address this Rahane et al. in [1] proposed an entertainment service system that users could connect to and view content during their journey. Additionally, Rahane et al. in [1] also offered a recommendation system to persons who connected to their system. While the work done by Rahane et al. in [1] may address the issues faced by bored passengers, its major flaw is that the system is completely offline and only maintains a network in the bus. Therefore, the system has to be manually updated,

Y. Yin et al. (Eds.): MobiCASE 2019, LNICST 290, pp. 68–82, 2019.
https://doi.org/10.1007/978-3-030-28468-8_6

which does not cater to wide-scale implementation on a fleet of busses. Additionally, Rahane et al. in [1] introduced additional components such as a router to facilitate users connecting to the system. This incurs an additional cost when trying to develop a low-cost system. In this paper, we implement a similar concept to the one introduced by Rahane et al. in [1]. However, our system only uses the Raspberry Pi which reduces the cost of the system as well as it allows content to be uploaded remotely using the optional 3G/4G modem or via Wi-Fi when it becomes available. Most importantly, our system allows selected users to connect to the internet directly. Users can view educational content on select web pages. The Edu-BUS Wi-Fi system records the MAC address of each connecting user. This allows administrators to monitor the connection rate and time for each vehicle. Additionally, administrators can restrict persons from performing certain activities if necessary, based on their device's MAC address. The system is also capable of being implemented in any vehicle that has a functional cigarette lighter outlet but can also operate as a self-powered device for a limited time based on the internal battery storage capacity. Our approach lends to extensibility and the ability to offer various content types on various modes of transportation.

2 Related Works

In this section, we will discuss similar systems to our work. We will break down these works into sections which will focus on the core features of our work which are: onboard systems, Raspberry Pi systems, and Wi-Fi Captive portal systems.

2.1 On-Board Systems

We formerly mentioned work by Rahane et al. in [1], which developed a system that was capable of delivering entertainment content to passengers. They used a Raspberry Pi, router and a USB Wi-Fi dongle to provide content using a server which was installed on the Raspberry Pi, a block diagram illustrating their work is shown in Fig. 1.

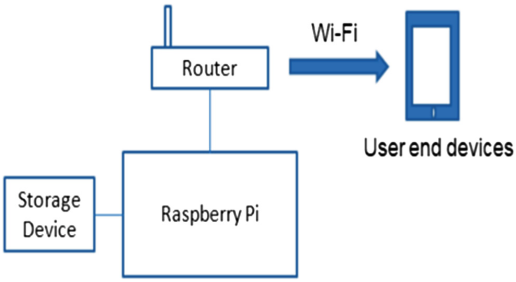

Fig. 1. Raspberry Pi block diagram [1].

The server delivered pages with static content, allowing users to select various media to view. Rahane et al. in [1] mentioned that this method reduced the data cost for users since the content was stored locally. From this work, we gathered that the Raspberry Pi could be used to serve web pages, as well as it could be used as a recommender system to suggest media content to users. However, this work has some limitations such as the ability to deploy new content remotely. This would have a serious impact if being deployed on a fleet of vehicles. Additionally, this system does not log or report user activity and does not allow access control of individual users. It is also unclear how the equipment is powered and the ease of deployment of such a system on multiple vehicles. We, therefore, performed a search for an easy deployment method of connecting such a system to a vehicle. We discovered the work done by Rathod et al. in [2] where they developed a vehicle tracking and air pollutant monitoring system. In their work, they were concerned about the air pollution poorly maintained vehicles would cause. Therefore, they developed a system which would monitor the air pollution emitted by the vehicle using an MQ-7 gas sensor. Based on the values which were generated by the sensor the system would notify the driver by sending an SMS message using a GSM module. The SMS received by the driver contained the status and emission level of their vehicle. Additionally, they used a 12 V supply from the vehicle and implemented a 5 V step-down module that would allow the ATmega328 microcontroller and sensor to receive power. From this work, we gathered the ability to use a 12 V supply from the vehicle. However, we also noted we must use a step-down module to allow the Raspberry Pi module to operate, since it needs a 5 V supply (Sect. 2.2).

We also found work by Shinde et al. [3] where they developed a vehicle monitoring system using a Raspberry Pi. The Raspberry Pi engaged a GSM, GPRS module SIM900A to transmit information to the server. They also added temperature and gas sensors to the Raspberry Pi to ensure students' safety on their journey. From this work, we learned how to connect the Raspberry Pi to the internet to transmit and receive information from a server using the SIM900A module. However, passengers on the vehicle who were interested in viewing this information would have to use their phones GPRS data, which is inefficient [1]. Our system allows passengers to view information whilst on-board using their cell phone. Our system eliminates the need to install additional software to use such features while aboard the vehicle.

2.2 Raspberry Pi Systems

The Raspberry Pi was originally designed for educational purposes [1]. However, the Raspberry Pi is used in a wide range of applications ranging from home automation to educational projects. Guravaiah and colleagues [4] propose an algorithm called River Formation Dynamics based the Multi-hop Routing Protocol for Vehicles (RFDMRPV) that addressed problems such as vehicle theft within a locality. To test this algorithm, they implemented it using open source platform systems such as the Raspberry Pi, Arduino and XBee which uses the ZigBee protocol. In their work, they described the details of the Raspberry Pi 3 B model which are shown in Fig. 2.

CPU	1.2GHz 64-bit Quad Core ARMv8 CPU
GPU	Video Core IV 3D Graphics Core
RAM	1GB LPDDR2 (900 MHz)
GPIO	40-pin header, Populated
Networking	802.11n Wireless LAN and 10/100 Wired LAN
Bluetooth	Bluetooth Low Energy (BLE), Bluetooth 4.0
Storage	Micro SD Card Slot
Ports	HDMI, 3.5mm analogue audio-video jack, 4 x USB2.0, Ethernet, Camera Serial Interface (CSI), Display Serial Interface (DSI)
Power	Micro USB. Requires 5V, 2.5Amp

Fig. 2. Details on Raspberry Pi 3 B device [4].

We learned that the Raspberry Pi requires 5 V power from a micro USB adapter. Additionally, we also discovered that the Raspberry Pi 3 B model has Wi-Fi capabilities that can be used to connect to wireless routers or as a Wi-Fi hotspot. This work by Rahane and colleagues is similar to the work done in [5] by Ward and Gittens where they developed a system using repurposed cell phones to monitor Wi-Fi signals.

Bhardwaj and associates [6] developed a system called Wi-Pi that used the Raspberry Pi and an additional Wi-Fi dongle to monitor Wi-Fi in an enterprise environment. They deployed various Raspberry Pis to store Wi-Fi performance. When a connection to the Wi-Pi server was established, the Raspberry Pis transmitted the performance data to the main Wi-Pi server. From this work, we learned how to use the Raspberry Pi to connect to access points. However, as previously identified, the Raspberry Pi 3 B model has built-in Wi-Fi, therefore, this is not necessary.

In addition to using a Wi-Fi dongle with the Raspberry Pi to enable network services, we can use a GSM 3G/4G modem to connect the Raspberry Pi to the cellular network as shown in [7]. Vujović in [7] proposed a system that provides a sensor node that is accessible all over the world using either a GSM/GPRS shield or a USB 3G/4G modem. This system would be useful in situations where sensors are deployed in dangerous or hazardous areas. This work demonstrates how to integrate the Raspberry Pi with the cellular GSM/GPRS network by using a GSM/GPRS shield or by connecting a USB GSM 3G/4G modem. Since the Raspberry Pi does not contain self-powering capabilities, we must consider instances where the vehicle is turned off. At this time, the vehicle may still have occupants that would rely on the Edu-BUS.

We therefore, need to identify a method to keep the Raspberry Pi powered even with additional attachments. Sakai and Sugano [8] developed a system that can track humans. Use cases for this work include small children and dementia patients. Sakai and Sugano used a Raspberry Pi, GPS module, battery, 3G modem and a Wi-Fi adapter. They used a battery to keep the Raspberry Pi powered. This allowed the device to be deployed wirelessly in a stuffed animal 30 cm in size. From this work we learned how to connect the Raspberry Pi to a battery that allowed it to be powered without a wired connected power source.

2.3 Wi-Fi Captive Portal Systems

Dabrowski et al. in their work in [9] introduced the concept of captive portals. In their work, they mentioned that captive portals are used worldwide in restaurants, airports and train stations, which offer Wi-Fi Hotspots. Captive portals allow administrators to set a display page that requires some action from the user before they are allowed Wi-Fi access. Dabrowski et al. [9], created a proof of concept to raise awareness that personal privacy could be at risk when using public Wi-Fi with captive portals implemented. To test this, they deployed a virtual machine that had a USB Wi-Fi adapter (TP-LINK TL-WN722 N) configured in access point mode. From this work we learned that captive portals could be used to control access to free Wi-Fi access points. Additionally, we recognized that USB Wi-Fi dongles, particularly the (TP-LINK TL-WN722N) could be configured to not only connect to Wi-Fi access points but could also be configured as an AP. Gatehouse [10] implemented a Wi-Fi network and captive portal gauging the multiple meanings of free and public Wi-Fi.

The interface requires users to navigate the interface which should prompt reflection. From this work, we gathered a captive portal could be used for more than just accepting user credentials but could also be used to display various types of information before users are allowed to connect.

2.4 Related Works Summary

From our survey of related works, we noted it was possible to use a Raspberry Pi with a router to provide Wi-Fi access to resources on the Raspberry Pi. However, we noted that the Raspberry Pi 3 B has onboard Wi-Fi and therefore is suited to replace the external router mentioned in the work by Rahane et al. in [1]. We also found that it is possible to use a captive portal to capture information from the user before they are provided with free Wi-Fi service. Additionally, from the work done by Rathod et al. in [2] we noted that a 5-V step-down module can be used in a vehicle system to connect microcontrollers. We, therefore, can use this approach to connect the Raspberry Pi to a vehicle which has a 12-V supply. Having identified similar systems, devices and approaches employed we will now define the Edu-Bus Wi-Fi system.

3 Edu-BUS Wi-Fi Implementation

In this section we will discuss the implementation of Edu-Bus Wi-Fi which seeks to address our research question: *Can a low-cost system be deployed onboard multiple vehicles to offer Free Wi-Fi to passengers and allow administrators to monitor and control user activities and authentication remotely.*

To implement the free Edu-BUS Wi-Fi we used the following:

- Raspberry Pi
- Battery
- USB 3G/4G modem

We will now discuss how we configure the devices mentioned previously to offer the Free Edu-Bus Wi-Fi which is illustrated in Fig. 3.

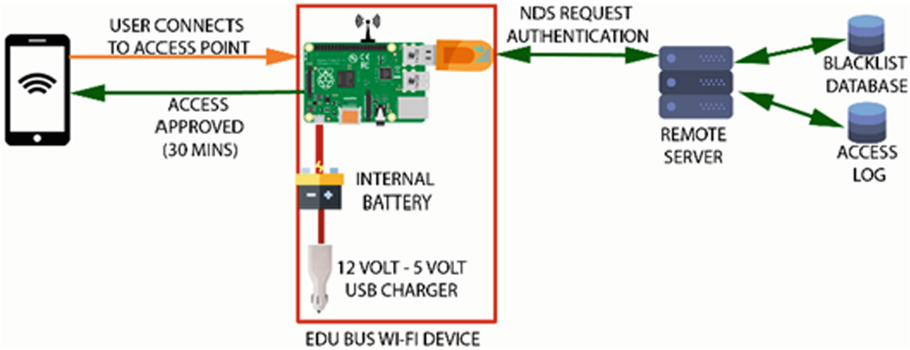

Fig. 3. Free Wi-Fi Edu-Bus

3.1 Wireless Configuration

In this section, we discuss how we configured the Raspberry Pi to offer Wi-Fi access to persons on-board. We recall the work done by Rahane et al. in [1] used a Raspberry Pi with a USB Wi-Fi dongle and a router to enable users to connect to their devices wirelessly while onboard. However, we also recall in the work done by Kataoka and Kumar in [6] mentioned the Raspberry Pi has built-in Wi-Fi capabilities. We also recognized in the work done by Dabrowski et al. in [9] some USB adapters can be converted into AP mode that allows users to connect their devices. We, therefore, investigated the possibility of using the Raspberry Pi as an access point which would remove the need to use a wireless router done in the work by Rahane et al. in [1].

Kim and Lee [11] used open source software to deploy APs. They identified HOSTAPD and OpenWRT as software that is useful to convert the Raspberry Pi into an AP. We, therefore, investigated the HOSTAPD and OpenWRT and recognized OpenWRT requires the Raspberry Pi to run on specific software where as HOSTAPD can be installed on the native Raspbian platform designed for the Raspberry Pi. This would allow us to install other native features to that operating system while having the Raspberry Pi configured as an AP. Additionally, Kim and Lee [11] identified DHCPD to allocate IP addresses to users after they connect to the AP. Having installed HOSTAPD and used DNSMASQ, we have a functioning wireless AP that users can connect to using Wi-Fi enabled devices. We must now identify compatible software with the Raspbian operating system that would enable us to offer captive portal features as discussed previously in Sect. 2.3.

3.2 Captive Portal

In this section, we discuss how we integrate a captive portal with our Raspbian operating system and are already installed HOSTAPD and DNSMASQ open source software. We, therefore, performed a search to identify compatible captive portal software for Raspbian. We identified Nodogsplash which is suggested in the following tutorials [12, 13]. We also discovered Kupiki Hotspot [14], which was an implementation of CoovaChilli and Freeradius described in the tutorial found on [15]. Other possible solutions were proposed such as Tornado but were not widely used and implemented. We selected Nodogsplash since it was well documented and there are various tutorials which use this as their solution for a Captive Portal. Additionally, NoDogSplash is well documented in addition to the tutorials its documentation can be found on [16]. We discuss NoDogSplash and its features that we could use in our system.

Nodogsplash (NDS): In addition to the captive portal feature, Nodogsplash(NDS) offers other features which could be used in our solution by restricting and applying access control measures on users these include:

- Blocking all outgoing packets
- Performing Packet filtering
- Forwarding External Authentication Methods (FAS)

Blocks All Outgoing Packets: NDS intercepts all outgoing packets which have a destination of port 80. NDS will display its default page when a user attempts to access a webpage. In some cases, devices may be equipped with Captive Portal Detection (CPD) and will automatically display the NDS splash page. This function can be used to validate users before they are allowed to access the internet and other services provided by Edu-BUS Wi-Fi.

Performs Packet Filtering: By inserting rules into the iptables, NDS can filter incoming packets that contain certain marks and forwards matching ones. Packets coming through the router is one of the following types:

- **Blocked** – The MAC address of the transmitting device is in the BlockedMACList or the MAC address is not in the AllowedMACList or TrustedMACList. These packets are dropped [16].
- **Trusted** – The MAC address of the transmitting device exist in the TrustedMACList. These packets by default are automatically routed to their destination port [16].
- **Authenticated** – The IP address and MAC address has been authenticated using the NDS process and has not yet expired. These packets are routed to their destination [16].
- **Preauthenticated** - Packets which have a port destination or addresses not allowed by the NDS configuration are dropped except packets with a destination of Port 80. Packets addressed to Port 80 are redirected to Port 2050. NDS has an libhttpd-based web server that is listening on Port 2050 and displays the NDS splash page. The user is then authenticated by performing some defined activity.

Facilitates External Authentication Methods: We discussed the packet filtering capabilities of NDS that enables us to authenticate users. However, NDS has the ability to use an external method for validating users this could be done on an external server. NDS offers four methods of authentication using its Forwarding Authentication Service (FAS) these are:

- **Fasport** – The port number of the NDS is changed to a Port that another application is listening.
- **Fasremoteip** – NDS navigates to this address for authentication.
- **Faspath** – NDS will navigate to this path for authentication.
- **Fas_secure_enable** – When set to 1 the client token which is used for authentication is held and FAS has to request a token using NDSCTL. However, if this value is set to 0, NDS will provide the client token in clear text to FAS along with authentication and redir.

NDS Configuration: We selected the Fasremote method for authentication as this would allow us to remotely control the authentication of users. However, this requires us to have an internet connection on our Raspberry Pi to connect to our server since NDS FAS will navigate to http://[fasremoteip]:[fasport]/faspath?authaction=http://[-gatewayaddress]:[gatewayport]/nodogsplash_auth/?clientip=[clientip]&gatewayname=[gatewayname]&tok=[token]&redir=[requested_url] passing the above parameters. We, therefore, need to have an established internet connection on the Raspberry Pi to facilitate the connection.

3.3 Enabling Internet Connection

Since the system is being deployed on a moving vehicle the method of connecting to the internet needs to consider this factor. We recall in the work done by Vujović in [7], they proposed the use of a GSM/GPRS shield or a USB 3G/4G modem to connect Raspberry Pis to the internet using the cellular network. We selected the USB 3G/4G method as it was the most available method to us. We selected the Huawei E3372h-510 Unlocked 150 Mbps 4G LTE USB Stick shown in Fig. 4 after performing a search to identify modems which were both compatible with our 4G LTE bands and with the Raspberry Pi. The Huawei E3372h-510 adapter auto-configures to enable internet connection on the Raspberry Pi.

Fig. 4. Huawei E3372h-510

Additionally, we tested an alternative method that could be used in the event that a USB 3G/4G adapter is not available. This method was similar to the method employed by Rahane et al. in [1]. We attached a USB Wireless adapter to the Raspberry Pi similar to Rahane et al. in [1] however, the router which we connected to was a Huawei E5573C modem with 3G/4G connectivity the device is shown in Fig. 5.

Fig. 5. Huawei E5573C Modem

This required us to create a shell script which would ensure the Raspberry Pi remained connected to the Wi-Fi. However, we recognized this approach had a major flaw. When the vehicle is switched off for an extended period and the backup battery has no remaining charge the device will switch off completely. When the vehicle is switched back on the device will not automatically turn back on it will require the power button to be pressed. Whereas, with the USB 3G/4G approach as soon as the Raspberry Pi receives power again the USB 3G/4G will switch on and auto reconfigure itself to provide internet connectivity. This capability is important to avoid the need for human interaction. Additionally, to avoid human interaction with the Edu-Wi-Fi Bus system we need to develop a method of implementation which does not require an action from the vehicle operator after installation.

3.4 Vehicle Installation

We recall in the work done by Rathod et al. in [2] the authors used the 12 V power supply from the vehicle to power their device. Additionally, they employed the use of a 5 V step-down module since their microcontroller required a 5 V power supply. We also recall in the work done by Guravaiah, Thivyavignesh and Velusamy in [4] the Raspberry Pi also requires a 5 V power supply. We, therefore, investigated how we could power the Raspberry Pi from a vehicle employing a step-down module. We discovered the cigarette lighter port which is available in most vehicles and is used to charge phones and other devices. The USB adapter converts the 12 V power supply outputted from the cigarette lighter and converts it to a 5 V output that provides a USB male jack. By employing this installation method, it makes it easier to move the Edu-Bus Wi-Fi System to other vehicles if necessary. Figure 10 shows the USB 5 V adapter which is used to charge the Raspberry Pi. However, we recall when the vehicle is switched off no power is supplied from the cigarette lighter. Therefore, we have to implement a redundancy in the event that users wish to use the Edu-Bus Wi-Fi system for a limited time while the vehicle is switched off. We, therefore, recall the work done by Sakai and Sugano in [8] were they used a battery to power the Raspberry Pi which had various other modules connected to it. We, therefore, connected the Raspberry Pi to a battery and then connected the battery to the 5 V outlet in the USB adapter as shown in Fig. 3. This method would, therefore, allow us to connect the Edu-Bus Wi-Fi box to any vehicle which contained a functioning cigarette lighter and the system would remain functional for a limited time when the vehicle is switched off.

3.5 User Access Control Management

We now have an access point with NDS enabled and when a user connects to the access point, they will be presented with the NDS splash page shown in Fig. 6. When the image displayed in the center of the page is clicked the user is given access to free Wi-Fi. In answering our research question, we must have the ability to monitor and control users access to the free Wi-Fi resource.

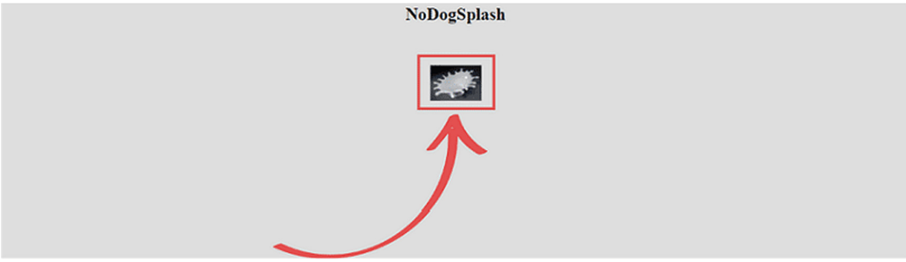

Fig. 6. NDS Page shown when a user connects to Edu Wi-Fi

We, therefore, need to implement a method that would allow us to configure the NDS splash page remotely. We recall NDS allows for external authentication (FAS). We also note using FAS we can configure NDS to redirect the splash page to a remote URL and port and pass it parameters. This would enable us to remotely control and update what is displayed to the user before they are authenticated.

Authenticate Users Remotely. We recall the FAS functionality of NDS allows NDS to be configured to direct users to a remote host for it to provide authentication. We, therefore, created a page using PHP and javascript on our remote server which NDS FAS would be forwarded to. NDS FAS has the ability to pass the mac-address of the user's device who is seeking to access the Edu Wi-Fi system. We recorded the mac-address of each user to perform authentication and monitoring. Before giving access to the user their mac-address is checked against the blacklist database. If their mac-address is found they are denied access.

Remotely Monitor Users Activity. We mentioned previously we attain the mac-addresses from NDS FAS as users attempt to connect. These mac-address are stored in our database along with the Gateway mac-address which is the mac-address of the Raspberry Pi they are connecting too which is the unique mac address of each Raspberry Pi on each vehicle as well as the time of the activity.

Having designed a system which answers our research question. We will now test our system design and implementation by applying it to a case study which has the requirements of our research question.

4 CASE STUDY – Co-Pilot Pass

Our case study is motivated by the 2018 Internet Society Chapterthon which focused on promoting the safe usage of the internet of Things (IoT) devices. The University of the West Indies Cave Hill Campus (UWICHC) offers a shuttle service to students that allow them to travel off and on campus. Our proposed approach was to use the Edu-Bus Wi-Fi system to educate students who boarded any four (4) of the UWICHC shuttles. To do this we provided free Wi-Fi access to students aboard however, before connecting they were presented with a captive portal that required them to watch an educational video based on safe usage of IoT devices. Additionally, we needed to record the number of unique users who would have watched the IoT videos to gauge the effectiveness of the campaign. We will now discuss how we used the Edu-Bus Wi-Fi system to implement the Co-Pilot Pass project.

4.1 Co-Pilot Pass Requirements

We will now state the requirements of the Co-Pilot Pass project and identify how the Edu-Bus Wi-Fi system was used to address these requirements. Below we list the requirements:

- Authenticate users remotely after watching educational videos.
- Display, edit and update educational video paths remotely.
- Record the number of unique users who watched the IoT videos.

Authenticate Users Remotely After Watching Educational Videos. We recall the FAS functionality of NDS allows NDS to be configured to direct users to a remote host for it to provide authentication. We, therefore, created a page using PHP and javascript on our remote server which NDS FAS would be forwarded to which is shown in Fig. 7. We placed an HTML 5 tag that linked to a video locally on the Raspberry Pi which maintained the benefit mentioned by Rahane et al. in [1] since the locally hosted video would not incur a cost. We then used JavaScript to listen for a video complete event. After this event was received, we used JavaScript to hide the video tag and display a button shown in Fig. 8. This button contains the link which is required for NDS to authenticate the user by placing the URL and Port to NDS mentioned in Sect. 3.3 and by passing the user token and redir parameter.

Display, Edit and Update Educational Video Paths Remotely. We stated in Sect. 4.1 that we insert a video link into the page that allows for the loading of the video locally hosted on the Raspberry Pi. However, we should be able to display multiple videos and update these videos remotely. We, therefore, implemented a bash script which contained an RSYNC function called by a Cron job which would sync the local folder with the remote folder on our server at 12:15 AM daily. This time was

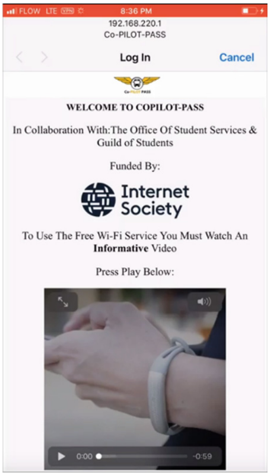

Fig. 7. Edu Bus Wi-Fi displaying a video to users before they connect.

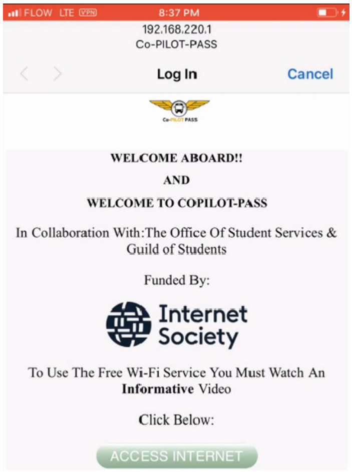

Fig. 8. Edu Bus Wi-Fi displaying a button after users watched the video.

chosen based on operation times of shuttle and based on user connections by time analysis discussed in the section below. Additionally, we wished to randomize the videos being displayed we, therefore, included a JavaScript function that would allow us to check the contents of the remote directory through PHP and identify the videos that could be displayed. Additionally, we implemented a feature which allowed the script to randomize the selection of a video from the video folder in the event only one video should be displayed. However, our script accepted parameters from FAS and could, therefore, be used to further customize which video and how it is displayed.

Record the Number of Unique Users Who Watched the IoT Videos. We recall from Sect. 3.5 we recorded the mac-address of each user which connected to the Edu Bus Wi-Fi. This activity was also recorded with other parameters passed by NDS FAS which include the device gateway (Raspberry Pi mac-address). From this information, we can, therefore, ascertain the number of users which connected and viewed the videos which were 829 unique users and 2724 sessions. Figure 9 shows the number of sessions which occurred over the 3-week testing period per bus. We notice bus 1 sessions are much lower than the other busses having only 122 sessions during the test period. This occurred bus was having mechanical issues during that time and was removed from operation from time to time.

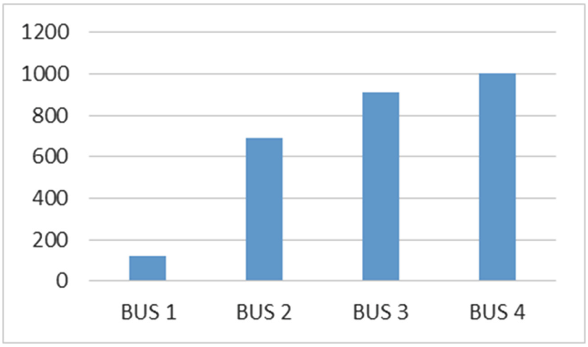

Fig. 9. A graph illustrating the number of connections and video views per bus

While we achieved the requirements of the Co-Pilot project by attaining the number of unique users and number of sessions were persons viewed the education videos we were also able to provide additional information which could be used to target users on particular days ant particular times. In Fig. 10 we show the user connections and sessions by day of the week and Fig. 11 by hour of the day.

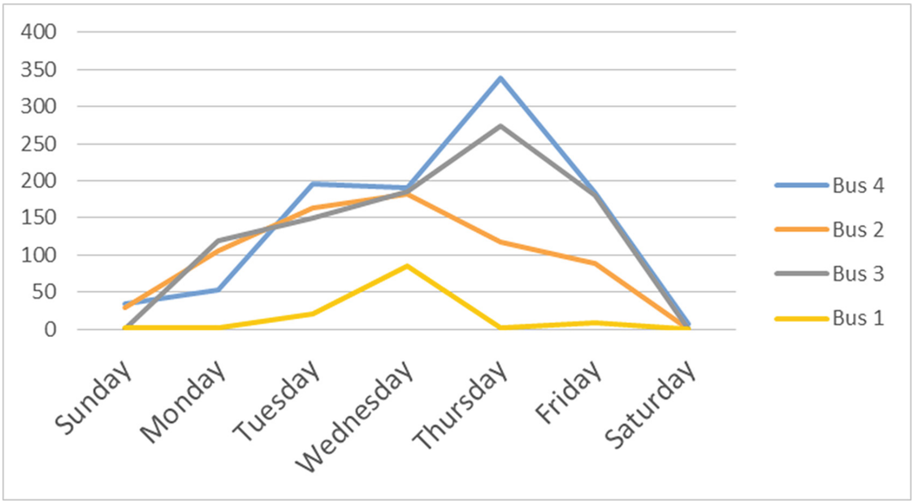

Fig. 10. Sessions by day of the week

The day of the week view of the data showed users on Sunday and Saturday had very few sessions. This is consistent with the operation days of the shuttle which is Monday to Friday. However, from time to time on Sundays the shuttles are mobilized to ferry students to various events we can, therefore, see a slight number of users connected on Sunday on Bus 2 and Bus 4 in relation to Saturday. Additionally, the same could be said about hours of the day since the shuttle operates from 6 AM until 11:00 PM. We can also see most connections occurred during the morning between the hours of 7 PM–9 PM.

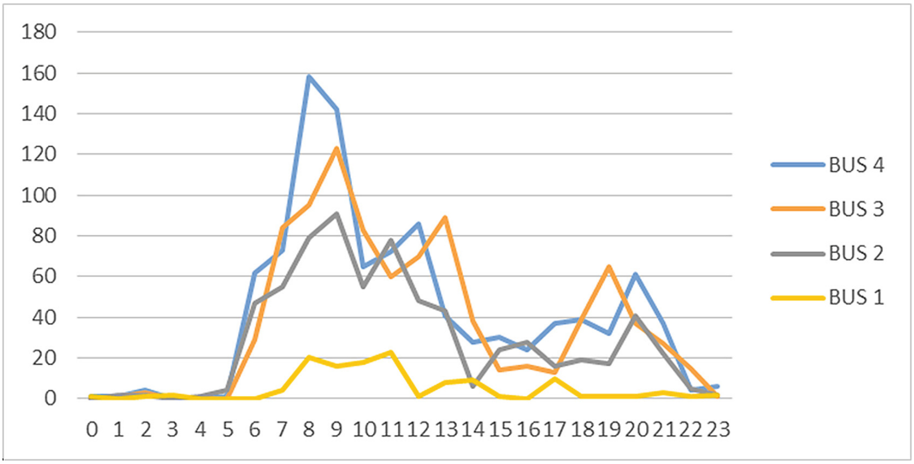

Fig. 11. Session per hour

5 Conclusions and Future Work

In this paper, we developed a system capable of providing internet enabled Wi-Fi onboard a vehicle using a Raspberry Pi and a USB 3G/4G modem. Additionally, we integrated NDS and used its FAS capabilities to perform validation and authentication of users seeking to use the onboard Wi-Fi. Our system is also capable of displaying content during the authentication process which must be viewed by a user before access is given. To reduce the data cost content is stored locally on the Raspberry Pi. However, the content is automatically synchronized with a remote server and therefore allows content to be uploaded remotely. Our system can be used on-board any vehicle which has a functioning cigarette lighter. It can also function when the vehicle is completely turned off for a limited period of time based on the battery capacity. We tested our system on-board four (4) UWICHC shuttles which has a ridership of 20,000 students per semester. We collected the connection time mac-address and default gateway mac-address of each user. We tested the system for three weeks and recorded 829 unique users and a total of 2724 sessions. We also presented information which could be used to determine the peak times when users connect to Wi-Fi onboard the vehicles and could be used to display special videos during that time.

References

1. Rahane, W.P., Anand Kabra, S., Ravindra Kore, P., Dipakkumar Pawar, A., Vijaykumar Gugale, A.: Onboard Entertainment Services using Raspberry Pi with Recommendation Engine (2018)
2. Rathod, M., Gite, R., Pawar, A., Singh, S., Kelkar, P.: An air pollutant vehicle tracker system using gas sensor and GPS, City (2017)

3. Shinde, P.A., Mane, Y.B., Tarange, P.H.: Real time vehicle monitoring and tracking system based on embedded Linux board and android application (2015)
4. Guravaiah, K., Thivyavignesh, R.G., Velusamy, R.L.: Vehicle monitoring using internet of things. In: Proceedings of the Proceedings of the 1st International Conference on Internet of Things and Machine Learning. ACM, Liverpool (2017)
5. Ward, S.A., Gittens, M.: Monitoring and analyzing wi-fi availability and performance on a university campus using recycled cell phones to aid students in selecting study areas. In: Proceedings of the Proceedings of the 2016 ACM on SIGUCCS Annual Conference, Denver, Colorado, USA, 2016). ACM
6. Bhardwaj, D., Kataoka, K., Kumar, V.: Wi-Pi: distributed wi-fi performance assessment using raspberry pi. In: Proceedings of the Proceedings of the AINTEC 2014 on Asian Internet Engineering Conference. ACM, Chiang Mai (2014)
7. Vujović, V., Maksimović, M., Perišić, B., Milošević, G.: A proposition of low cost Sensor Web implementation based on GSM/GPRS services. City (2015)
8. Sakai, M., Sugano, M.: Human tracking system embedded in stuffed animal. City (2016)
9. Dabrowski, A., Merzdovnik, G., Kommenda, N., Weippl, E.: Browser History Stealing with Captive Wi-Fi Portals. City (2016)
10. Gatehouse, C.: FreeAsInWifi (2015)
11. Junho, K., Sungwon, L.: Constructing Infrastructure wireless network using open source. City (2016)
12. Gus Setting up a Raspberry Pi Captive Portal. PiMyLifeUp, City (2018)
13. Guy, C.S.: Building a Raspberry Pi Captive Portal Wi-Fi Hotspot Cyber Security Guy, City
14. Server, R.P.H.: Kupiki-Hotspot-Script
15. Server, R.P.H.: Raspberry Pi, CoovaChilli and Freeradius for a Wifi Hotspot with captive portal. Raspberry Pi Home Server, City (2016)
16. Nodogsplash Nodogsplash. City (2018)